On Black-Box Complexity and Adaptive, Universal Composability of Cryptographic Tasks

Dana Dachman-Soled

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2011

ABSTRACT

On Black-Box Complexity and Adaptive, Universal Composability of Cryptographic Tasks

Dana Dachman-Soled

Two main goals of modern cryptography are to identify the minimal assumptions necessary to construct secure cryptographic primitives as well as to construct secure protocols in strong and realistic adversarial models. In this thesis, we address both of these fundamental questions.

In the first part of this thesis, we present results on the black-box complexity of two basic cryptographic primitives: non-malleable encryption and optimally-fair coin tossing. Black-box reductions are reductions in which both the underlying primitive as well as the adversary are accessed only in an input-output (or black-box) manner. Most known cryptographic reductions are black-box. Moreover, black-box reductions are typically more efficient than non-black-box reductions. Thus, the black-box complexity of cryptographic primitives is a meaningful and important area of study which allows us to gain insight into the primitive.

We study the black box complexity of non-malleable encryption and optimally-fair coin tossing, showing a positive result for the former and a negative one for the latter. Non-malleable encryption is a strong security notion for public-key encryption, guaranteeing that it is impossible to "maul" a ciphertext of a message m into a ciphertext of a related message. This security guarantee is essential for many applications such as auctions. We show how to transform, in a black-box manner, any public-key encryption scheme satisfying a weak form of security, semantic security, to a scheme satisfying non-malleability. Coin tossing is perhaps the most basic cryptographic primitive, allowing two distrustful parties to flip a coin whose outcome is 0 or 1 with probability $1/2$. A fair coin tossing protocol is one in which the outputted bit is unbiased, even in the case where one of the parties may abort early. However, in the setting where parties may abort early, there is always a strategy for one of the parties to impose bias of $\Omega(1/r)$ in an r-round protocol. Thus, achieving bias of $O(1/r)$

in r rounds is optimal, and it was recently shown that optimally-fair coin tossing can be achieved via a black-box reduction to oblivious transfer. We show that it cannot be achieved via a black-box reduction to one-way function, unless the number of rounds is at least $\Omega(n/\log n)$, where n is the input/output length of the one-way-function.

In the second part of this thesis, we present protocols for multiparty computation (MPC) in the Universal Composability (UC) model that are secure against malicious, adaptive adversaries. In the standard model, security is only guaranteed in a stand-alone setting; however, nothing is guaranteed when multiple protocols are arbitrarily composed. In contrast, the UC model, introduced by (Canetti, 2000), considers the execution of an unbounded number of concurrent protocols, in an arbitrary, and adversarially controlled network environment. Another drawback of the standard model is that the adversary must decide which parties to corrupt before the execution of the protocol commences. A more realistic model allows the adversary to adaptively choose which parties to corrupt based on its evolving view during the protocol. In our work we consider the the adaptive UC model, which combines these two security requirements by allowing both arbitrary composition of protocols and adaptive corruption of parties.

In our first result, we introduce an improved, efficient construction of non-committing encryption (NCE) with optimal round complexity, from a weaker primitive we introduce called *trapdoor-simulatable public key encryption (PKE)*. NCE is a basic primitive necessary to construct protocols secure under adaptive corruptions and in particular, is used to construct oblivious transfer (OT) protocols secure against semi-honest, adaptive adversaries. Additionally, we show how to realize trapdoor-simulatable PKE from hardness of factoring Blum integers, thus achieving the first construction of NCE from hardness of factoring. In our second result, we present a compiler for transforming an OT protocol secure against a semi-honest, adaptive adversary into one that is secure against a malicious, adaptive adversary. Our compiler achieves security in the UC model, assuming access to an ideal commitment functionality, and improves over previous work achieving the same security guarantee in two ways: it uses black-box access to the underlying protocol and achieves a constant multiplicative overhead in the round complexity. Combining our two results with the work of (Ishai et al., 2008), we obtain the first black-box construction of UC and adaptively secure MPC from trapdoor-simulatable PKE and the ideal commitment functionality.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I am indebted to my advisor, Tal Malkin, for introducing me to research in cryptography and for her guidance, encouragement and friendship throughout these years. Tal's love for research is infectious and her insights and unique perspectives have been invaluable. I am grateful to Rocco Servedio, who co-advised me during the first few years of my graduate studies. Rocco has patiently taught me much about the right way to do research and how to do it with integrity. Moreover, I would like to thank Rocco for all of his dedication, guidance and advice. It has been a privilege to have had Tal and Rocco as my advisors and mentors.

I would like to thank Yevgeniy Dodis for introducing me to the field of cryptography while I was an undergraduate. His energy and enthusiasm played a large part in directing my path towards research in this field. I have been very fortunate to have had several individuals play an informal mentoring role throughout my studies. In particular, I would like to thank Hugo Krawczyk, Tal Rabin and Hoeteck Wee for their personal attention, guidance and advice.

I would like to thank Yehuda Lindell for hosting me during Summer 2009 at Bar Ilan and would like to thank Tal Rabin and the IBM crypto group for hosting me during Summer 2010. I am grateful to the co-authors of the works included in this thesis: Seung Geol Choi, Yehuda Lindell, Mohammad Mahmoody, Tal Malkin and Hoeteck Wee. As well I am grateful to my other co-authors and collaborators: Eric Blais, Ran Canetti, Rosario Gennaro, Anatoly Frenkel, Hugo Krawczyk, Homin Lee, Mariana Raykova, Rocco Servedio, Vugranam Sreedhar, Vinod Vaikuntanathan, Muthu Venkitasubramaniam, Andrew Wan and Moti Yung. Thank you also to Rosario Gennaro, Tal Malkin, Rocco Servedio, Hoeteck Wee and Moti Yung for serving on my thesis committee.

Finally, I would like to thank my family. Thank you to my parents Moshe and Yael Glasner for their love and patience and for always supporting and encouraging me. Thank you to my husband, Josh, for all his love and support. Thank you to my little Huda for always cheering me up at the end

of a long day.

This research was supported in part by an FFSEAS Presidential Fellowship.

To: Josh, without whom this work would never have been possible.

Chapter 1

Introduction

> Is it possible for two parties who have never met to exchange messages over a public channel while an eavesdropper learns nothing?

This intriguing question was first posed over 30 years ago by Diffie and Hellman. The task seems impossible, and in an information-theoretic sense it is: A computationally unbounded eavesdropper can learn just as much about the exchanged messages as the designated parties. However, if we limit the computational powers of the parties and make strong computational hardness assumptions such as ”factoring is hard” or ”discrete log is hard” then, surprisingly, the above task and a myriad of other interesting tasks *can* be achieved.

Thus, the field of cryptography, as we know it today, relies on computational assumptions. Whether an assumption is credible or not may be difficult to determine and may change over time. Therefore, a main goal of the theoretical foundations of cryptography is to determine the minimal assumptions necessary for a given cryptographic task.

> Is it possible for mutually distrustful parties to jointly compute a function of their inputs while revealing nothing more than the output?

Whether or not a cryptographic task is achievable depends on how we model adversaries and how we define security. For example, depending on our adversarial model and security definition, it is either possible to achieve the task above under appropriate hardness assumptions, or it is unconditionally

impossible to achieve. More specifically, in a model where the protocol runs in a standalone setting, we can achieve the above task for any functionality, based on various computational hardness assumptions. However, if we require the protocol to be secure in a setting where concurrent executions are allowed, then it becomes impossible to compute most functionalities securely without trusted setup. Thus, a second main goal of the theoretical foundations of cryptography is to construct protocols that are secure in the most realistic models and against the strongest adversaries possible.

This thesis addresses these two broad goals by focusing on the following two topics: Investigating the black-box complexity of basic cryptographic primitives and constructing protocols that are secure in the UC-model under adaptive corruptions. Studying black-box complexity is a first step in determining minimal assumptions necessary for cryptographic tasks; the adaptive UC-model is one of the strongest and most realistic models that have been proposed thus far for proving security of cryptographic protocols.

1.1 Black-Box Complexity.

The study of the black-box complexity of a cryptographic primitive $\mathcal{P}$ involves determining the relationship between primitive $\mathcal{P}$ and some other primitive $\mathcal{Q}$ by showing either a black-box reduction from $\mathcal{P}$ to $\mathcal{Q}$, or ruling out such a reduction. Black-box reductions are reductions in which both the underlying primitive as well as the adversary are accessed only in an input-output (or black-box) manner. The vast majority of known cryptographic reductions are black-box (with a few notable exceptions [Feige *et al.*, 1999; Dolev *et al.*, 2000; Barak, 2001; Barak, 2002]). Moreover, black-box reductions are typically more efficient than non-black-box reductions since, in most cases (with the exception of the work of [Barak, 2001; Barak, 2002]) non-black-box reductions involve generic zero-knowledge (ZK) proofs. Such proofs tend to be inefficient for two reasons: First, generic ZK proofs usually require Karp reductions to an NP-complete problem such as Hamiltonian cycle, and although guaranteed to be polynomial-time, these reductions can be highly inefficient. Second, if the statement of the proof involves the underlying cryptographic primitive, as is the case in non-black-box usage, the size of ZK proof will depend on the complexity of the underlying primitive. For example, consider proving in ZK that two ciphertexts C_1, C_2 both encrypt the same message m (we note that a very similar example will come

up in the context of constructing non-malleable encryption schemes). The number of commitments required in the ZK proof will be at least the size of the verification circuit, which in turn is at least as large as the encryption circuit. For security parameter k, a typical encryption circuit has size k^3 ([Ishai *et al.*, 2006]) and so for a typical setting of $k = 1024$ or $k = 2048$ the proof will need to contain over 1 billion commitments. Clearly, this overhead is unsustainable in many practical applications.

One of the main goals of modern cryptography has been to identify the minimal assumptions necessary to construct secure cryptographic primitives. Understanding the black-box complexity of cryptographic primitives is a meaningful step towards reaching this goal. In the positive case, in which it is shown that there exists a black-box reduction from primitive $\mathcal{Q}$ to primitive $\mathcal{P}$, we may conclude that the minimal assumptions necessary for realizing $\mathcal{P}$ are also sufficient to achieve $\mathcal{Q}$. Searching specifically for *black-box* reductions is important since black-box constructions lead to simpler and more efficient protocols that avoid the inefficiency of generic zero knowledge proofs. In the negative case, in which it is proven that there exists no black-box reduction from primitive $\mathcal{Q}$ to primitive $\mathcal{P}$, we provide evidence that the minimal assumptions necessary for realizing $\mathcal{Q}$ are stronger than the minimal assumptions necessary for realizing $\mathcal{P}$. Indeed, the first work in this area by Impagliazzo and Rudich ([Impagliazzo and Rudich, 1989]) proved that there is no black-box reduction of key agreement (KA) to one-way function (OWF). Because of this result, it is widely accepted by the community that achieving KA requires stronger hardness assumptions than OWF, even though the result itself only rules out black-box reductions.

In this work we investigate the black box complexity of two basic cryptographic primitives, non-malleable encryption and optimally-fair coin tossing, showing a positive result for the former and a negative one for the latter. Non-malleable encryption is a strong security notion for public-key encryption, guaranteeing that it is impossible to "maul" a ciphertext of a message m into a ciphertext of a related message. This security guarantee is essential for many applications such as auctions. We show how to transform, in a black-box manner, any public-key encryption scheme satisfying a weak form of security, semantic security, to a scheme satisfying non-malleability. Coin tossing is perhaps the most basic cryptographic primitive, allowing two distrustful parties to flip a coin whose outcome is 0 or 1 with probability $1/2$. A fair coin tossing protocol is one in which the outputted bit is unbiased, even in the case where one of the parties may abort early. However, in

the setting where parties may abort early, it's been long known that there is always a strategy for one of the parties to impose bias of $\Omega(1/r)$ in an r-round protocol. Thus, achieving bias of $O(1/r)$ in r rounds is optimal, and it was recently shown that optimally-fair coin tossing can be achieved via a black-box reduction to oblivious transfer. We show that it cannot be achieved via a black-box reduction to one-way function, unless the number of rounds is at least $\Omega(n/\log n)$, where n is the input/output length of the one-way-function.

1.2 Adaptive UC-security.

In an overlapping theme, this work explores strong adversarial models and presents efficient, provably secure protocols in these models. In the standard model, security is only guaranteed in a stand-alone setting; however, nothing is guaranteed when multiple protocols are arbitrarily composed. Thus, in today's world of multiprogramming, proving security of protocols in the stand-alone setting does not give us meaningful real-world guarantees. In contrast, the UC (Univeral Composability) model, introduced by Canetti [Canetti, 2000], considers the execution of an unbounded number of concurrent protocols, in an arbitrary, and adversarially controlled network environment. Another drawback of the standard model is that the adversary must decide which parties to corrupt before the execution of the protocol commences. A more realistic model allows the adversary to adaptively choose which parties to corrupt based on its evolving view during the protocol. In our work we consider the adaptive UC model, which combines these two security requirements by allowing both arbitrary composition of protocols and adaptive corruption of parties.

We introduce an improved, more efficient construction of non-committing encryption (NCE). NCE is a basic primitive that is necessary for proving security under adaptive corruptions. Intuitively, NCE is a semantically secure public key encryption scheme with the added property that a simulator playing the part of both the sender and receiver can generate encryptions that can be opened later to either 0 or 1. NCE is necessary in the adaptive model even to realize the simplest functionality of private bit transfer over public channels. In our work we present a construction of NCE from a new, weaker primitive we introduce called *trapdoor-simulatable public-key encryption (PKE)*. Additionally, we show how to realize trapdoor-simulatable PKE from hardness of factoring Blum integers, thus achieving the first construction of NCE from hardness of factoring. In addition

to the private bit transfer functionality, NCE can be used to construct semi-honest oblivious transfer (OT), as shown by [Canetti *et al.*, 2002]. Building on this construction of semi-honest OT from NCE, we next present a compiler for transforming an OT protocol secure against a semi-honest, adaptive adversary into one that is secure against a malicious, adaptive adversary. Our compiler achieves security in the UC model, assuming access to an ideal commitment functionality, and improves over previous work achieving the same security guarantee in two ways: it uses black-box access to the underlying protocol and achieves a constant multiplicative overhead in the round complexity. Combining our two results with the work of [Ishai *et al.*, 2008], we obtain the first black-box construction of UC and adaptively secure MPC from trapdoor-simulatable PKE and the ideal commitment functionality.

Part I

Black-Box Complexity

Much work in modern cryptography has been dedicated to understanding the relationships between different primitives and notions of security. For example, [Yao, 1982; Goldwasser and Micali, 1984; Rompel, 1990; Håstad *et al.*, 1999; Goldreich *et al.*, 1986; Luby and Rackoff, 1988; Impagliazzo and Luby, 1989; Naor and Yung, 1989; Naor, 1991] have shown that private key encryption, pseudorandom generators, pseudorandom functions and permutations, bit commitment, and digital signatures exist if and only if one-way functions exist. On the other hand, some cryptographic primitives such as public key encryption, oblivious transfer, and key agreement are not known to be equivalent to one way functions. Thus, it is natural to ask whether the existence of one-way functions implies these primitives. However, it seems unclear how to formalize such a question; since it is widely believed that both one-way functions and public key encryption exist, this would imply in a trivial logical sense that the existence of one-way functions implies the existence of public key encryption. Thus, we can only hope to rule out restricted types of constructions that are commonly used to prove implications in cryptography. Impagliazzo and Rudich [Impagliazzo and Rudich, 1989] were the first to develop a technique to rule out the existence of an important class of reductions between primitives known as black-box reductions. Intuitively, this is a reduction where the primitive is treated as an oracle or a "black-box". There are actually several flavors of black-box reductions (fully black-box, semi black-box and weakly black-box [Reingold *et al.*, 2004]). In our work, we only deal with fully black-box reductions, and so we will focus on this notion here.

Informally, a fully black-box reduction from a primitive $\mathcal{Q}$ to a primitive $\mathcal{P}$ is a pair of *oracle* ppt Turing machines $(G, \mathcal{S})$ such that the following two properties hold:

Correctness: For every implementation f of primitive $\mathcal{P}$, $g = G^f$ implements $\mathcal{Q}$.
Security: For every implementation f of primitive $\mathcal{P}$, and every adversary $\mathcal{A}$, if $\mathcal{A}$ breaks G^f (as an implementation of $\mathcal{Q}$) then $\mathcal{S}^{\mathcal{A},f}$ breaks f. (Thus, if f is "secure", then so is G^f.)

Subsequently to the initial groundbreaking work of [Impagliazzo and Rudich, 1989], there has been substantial work on proving the impossibility of black-box reductions between other fundamental primitives. Such proofs are frequently called "black-box separations" since they show that one primitive is more powerful than the other with respect to the class of black-box reductions. Some examples of these works are: A separation between one-way permutations and collision-resistant hash functions [Simon, 1998], a separation between public key encryption (PKE) and

oblivious transfer (OT) [Gertner *et al.*, 2000], a separation between trapdoor predicates and trapdoor functions [Gertner *et al.*, 2001], a partial separation between semantically-secure encryption and CCA-secure encryption [Gertner *et al.*, 2007], as well as many other works.

Another related area of work has been to study the best possible efficiency of black-box reductions and to show that any black-box construction of some primitive $\mathcal{Q}$ from another primitive $\mathcal{P}$ must make at least q queries to primitive $\mathcal{P}$. Such questions have been studied in the works of [König *et al.*, 1999; Gennaro *et al.*, 2005; Barak and Mahmoody-Ghidary, 2007]. There are other notions of efficiency that may be of interest as well such as round-efficiency and communication complexity as studied in the works of [Haitner *et al.*, 2007; Haitner *et al.*, 2008].

A compelling question that arises from the above discussion is the following: are arbitrary reductions more powerful than black-box reductions? Intuitively, it may seem that non-black-box access to the code of an adversary does not give any practical advantage over input/output access since we know that "backward engineering" the code of the adversary is a hard problem. For example, if the code of the adversary is "obfuscated" then, by definition, having access to the code provides no advantage over having only input/output access to the adversary. However, in a groundbreaking result, [Barak, 2001] showed that this intuition is false: For specific primitives $\mathcal{P}$ and $\mathcal{Q}$, there exists a reduction from $\mathcal{Q}$ to $\mathcal{P}$ where the simulator accesses the code of the adversary $\mathcal{A}$, while such a reduction provably *does not* exist in the case where the simulator accesses the adversary $\mathcal{A}$ only in a black-box manner. Still, even in the context of the work of [Barak, 2001], the reduction always accesses the underlying primitive $\mathcal{P}$ in a black-box manner and only requires access to the code of the adversary, $\mathcal{A}$. This raises the question of whether reductions of cryptographic task $\mathcal{Q}$ to primitive $\mathcal{P}$ that access the code of $\mathcal{P}$ but not the code of $\mathcal{A}$ are more powerful than reductions that are black-box in both $\mathcal{P}$ and $\mathcal{A}$. In the past several years, there has been significant progress towards answering this question. It has been shown in multiple cases how to achieve fully-black-box constructions where previously all known constructions required non-black-box access to the underlying primitive (c.f. [Ishai *et al.*, 2006; Haitner, 2008; Choi *et al.*, 2008; Choi *et al.*, 2009b; Ishai *et al.*, 2008; Lindell and Pinkas, 2007]), and where previously non-black-box access was believed to be essential. In our work, we continue this line of research by presenting a black-box construction of non-malleable encryption from any semantically-secure encryption. Previously, all known constructions required non-black-box access to the underlying semantically-

secure encryption.

We also study the minimal assumptions necessary for optimally-fair coin-tossing. In the standard notion of security, "fairness" is not guaranteed: Namely, if one of the parties aborts early, perhaps after receiving output, nothing is guaranteed about the correctness of the output of the remaining party. Recently, the problem of fairness in computation has come to the forefront with the exciting works of [Gordon *et al.*, 2008; Gordon and Katz, 2009; Moran *et al.*, 2009; Gordon and Katz, 2010] after almost no progress in this area for over 20 years. Since coin-tossing is a foundational cryptographic primitive with widespread use in multiparty computation protocols, better understanding of the black-box complexity of coin-tossing is necessary to determine the possibilities and limitations of fairness in secure multiparty computation.

A fair two-party coin-tossing protocol is one in which both parties output the same bit that is almost uniformly distributed. Unfortunately, Cleve [Cleve, 1986] showed that for every coin-tossing protocol with r rounds, an efficient fail-stop adversary (adversary that behaves honestly but may choose to abort the protocol at any point) can bias the output by $\Omega(1/r)$. Since this is the best possible, a protocol that limits the bias of any adversary to $O(1/r)$ is called optimally-fair. The only known optimally-fair coin-tossing protocol (introduced recently by Moran et al. [Moran *et al.*, 2009]) relies on the existence of oblivious transfer (OT). However, it is possible to use Blum's protocol [Blum, 1981] to achieve a bias of $O(1/\sqrt{r})$ in r rounds relying only on the assumption that one-way functions (OWF) exist [Cleve, 1986].

In this work we address the question of whether it is possible to achieve a bias of $o(1/\sqrt{r})$ in r rounds relying only on the assumption that OWF exist. We show that it is impossible to achieve coin-tossing with bias $o(1/\sqrt{r})$ via a black-box reduction to OWF for r that is less than $O(n/\log n)$, where n is the input/output length of the OWF. However, the [Moran *et al.*, 2009] protocol based on OT achieves bias of $O(1/r)$ even for r less than $O(n/\log n)$. Thus, our work gives the first evidence that the assumptions needed for optimally-fair coin-tossing are stronger than those needed for unfair coin-tossing.

Chapter 2

Black-Box Construction of a Non-Malleable Encryption Scheme from any Semantically-Secure One

2.1 Introduction

The most basic security guarantee we require of a public key encryption scheme is that of semantic security [Goldwasser and Micali, 1984]: it is infeasible to learn anything about the plaintext from the ciphertext. In many cryptographic applications such as auctions, we would like an encryption scheme that satisfies the stronger guarantee of non-malleability [Dolev *et al.*, 2000], namely that given some ciphertext c, it is also infeasible to generate ciphertexts of some message that is related to the decryption of c. Motivated by the importance of non-malleability, Pass, Shelat and Vaikuntanathan raised the following question [Pass *et al.*, 2006]:

> It is possible to *immunize* any semantically secure encryption scheme against malleability attacks?

Pass et al. gave a beautiful construction of a non-malleable encryption scheme from any semantically secure one (building on [Dolev *et al.*, 2000]), thereby addressing the question in the affirmative. However, the PSV construction – as with previous constructions achieving non-

malleability from general assumptions [Dolev *et al.*, 2000; Sahai, 1999; Lindell, 2006] – suffers from the curse of inefficiency arising from the use of general NP-reductions. In this work, we show that we can in fact immunize any semantically secure encryption schemes against malleability attacks without paying the price of general NP-reductions:

> **Main theorem (informal)** There exists a (fully) black-box construction of a non-malleable encryption scheme from any semantically secure one.

That is, we provide a wrapper program (from programming language lingo) that given any subroutines for computing a semantically secure encryption scheme, computes a non-malleable encryption scheme, with a multiplicative overhead in the running time that is quasi-linear in the security parameter. Before providing further details, let us first provide some background and context for our result.

2.1.1 Relationships amongst Cryptographic Primitives

Much of the modern work in foundations of cryptography rests on general cryptographic assumptions like the existence of one-way functions and trapdoor permutations. General assumptions provide an abstraction of the functionalities and hardness we exploit in specific assumptions such as hardness of factoring and discrete log without referring to any specific underlying algebraic structure. Constructions based on general assumptions may use the primitive guaranteed by the assumption in one of two ways:

Black-box usage: A construction is black-box if it refers only to the input/output behavior of the underlying primitive; we would typically also require that in the proof of security, we can use an adversary breaking the security of the construction as an oracle to break the underlying primitive. (See [Reingold *et al.*, 2004] and references within for more details.). As emphasized earlier, our construction is black-box, using only oracle access to the key generation, encryption and decryption functionality of the underlying encryption scheme.

Non-black-box usage: A construction is non-black-box if it uses the code computing the functionality of the primitive. The PSV construction along with the work it builds on fall into this category: they use an NP reduction applied to the circuit computing the encryption

functionality of the underlying encryption scheme in order to provide a non-interactive zero-knowledge proof of consistency.

Motivated by the fact that the vast majority of constructions in cryptography are black-box, a rich and fruitful body of work initiated in [Impagliazzo and Rudich, 1989] seeks to understand the power and limitations of black-box constructions in cryptography, resulting in a fairly complete picture of the relations amongst most cryptographic primitives with respect to black-box constructions (we summarize several of the known relations pertaining to encryption in Figure 2.1). More recent work has turned to tasks for which the only constructions we have are non-black-box, yet the existence of a black-box construction is not ruled out. Two notable examples are general secure multi-party computation against a dishonest majority and encryption schemes secure against adaptive chosen-ciphertext (CCA2) attacks[1] (c.f. [Goldreich *et al.*, 1988; Dolev *et al.*, 2000]).

The general question of whether we can securely realize these tasks via black-box access to a general primitive is not merely of theoretical interest. A practical reason is related to efficiency, as non-black-box constructions tend to be less efficient due to the use of general NP reductions to order to prove statements in zero knowledge; this impacts both computational complexity as well as communication complexity (which we interpret broadly to mean message lengths for protocols and key size and ciphertext size for encryption schemes). Moreover, if resolved in the affirmative, we expect the solution to provide new insights and techniques for circumventing the use of NP reductions and zero knowledge in the known constructions. Finally, given that there has been no formal model that captures non-black-box constructions in a satisfactory manner, the pursuit of a positive result becomes all the more interesting.

Indeed, Ishai et al. [Ishai *et al.*, 2006] recently provided an affirmative answer for secure multi-party computation by exhibiting black-box constructions from some low-level primitive. Their techniques have since been used to yield secure multi-party computation via black-box access to an oblivious transfer protocol for semi-honest parties, which is complete (and thus necessary) for secure multi-party computation [Haitner, 2008]. This leaves the following open problem:

Is it possible to realize CCA2-secure encryption via black-box access to a low-level

[1]These are encryption schemes that remain semantically secure even under a CCA2 attack, wherein the adversary is allowed to query the decryption oracle except on the given challenge. A CCA1 attack is one wherein the adversary is allowed to query the decryption oracle before (but not after) seeing the challenge.

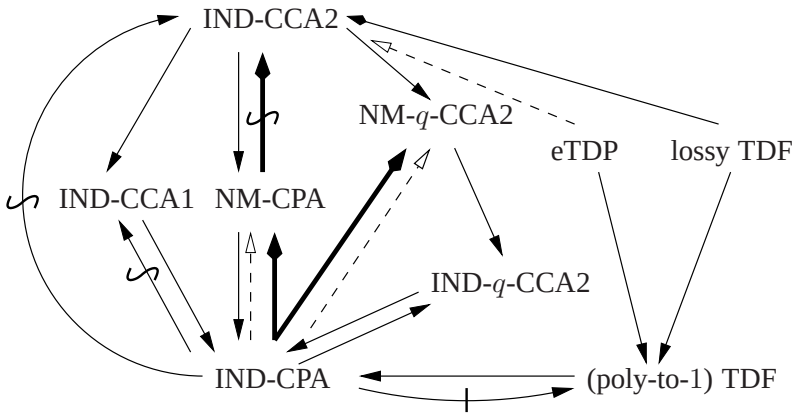
primitive, e.g. enhanced trapdoor permutations or homomorphic encryption schemes?

Previous work pertaining to this problem is limited to non-black-box constructions of CCA2-secure encryption from enhanced trapdoor permutations [Dolev *et al.*, 2000; Sahai, 1999; Lindell, 2006]; nothing is known assuming homomorphic encryption schemes. In work concurrent with ours, Peikert and Waters [Peikert and Waters, 2008] made substantial progress towards the open problem – they constructed CCA2-secure encryption schemes via black-box access to a new primitive they introduced called lossy trapdoor functions, and in addition, gave constructions of this primitive from number-theoretic and worst-case lattice assumptions. Unfortunately, they do not provide a black-box construction of CCA2-secure encryption from enhanced trapdoor permutations.

Our work may also be viewed as a step towards closing this remaining gap (and a small step in the more general research agenda of understanding the power of black-box constructions). Specifically, the security guarantee provided by non-malleability lies between semantic security and CCA2 security, and we show how to derive non-malleability in a black-box manner from the minimal assumption possible, i.e., semantic security. In the process, we show how to enforce consistency of ciphertexts in a black-box manner. This issue arises in black-box constructions of both CCA2-secure and non-malleable encryptions. However, our consistency checks only satisfy a weaker notion of non-adaptive soundness, which is sufficient for non-malleability but not for CCA2-security (c.f. [Pass *et al.*, 2006]). As a special case of our result, we obtain a black-box construction of non-malleable encryptions from any (poly-to-1) trapdoor function. Our results are incomparable with those of Peikert and Waters since we start from weaker assumptions but derive a weaker security guarantee.

Related positive results.

A different line of work focuses on (very) efficient constructions of CCA2-secure encryptions under specific number-theoretic assumptions [Cramer and Shoup, 1998; Cramer and Shoup, 2004; Canetti *et al.*, 2004]. Apart from those based on identity-based encryption, these constructions together with previous ones based on general assumptions can be described under the following framework (c.f. [Blum *et al.*, 1988; Naor and Yung, 1990; Rackoff and Simon, 1991; Elkind and Sahai, 2002]). Start with some cryptographic hardness assumption that allows us to build a semantically secure



Solid lines indicate black box constructions, and dotted lines indicate non-black-box constructions. The separations are with respect to black-box reductions, or black box shielding reductions. Our contributions are indicated with the thick arrows.

Figure 2.1: Relations among generic encryption primitives

encryption scheme, and then prove/verify that several ciphertexts satisfy certain relations in one of two ways:

- exploiting algebraic relations from the underlying assumption to deduce additional structure in the encryption scheme (e.g. homomorphic, reusing randomness) [Cramer and Shoup, 1998; Cramer and Shoup, 2004];
- apply a general NP reduction to prove in non-interactive zero knowledge (NIZK) statements that relate to the primitive [Dolev *et al.*, 2000; Sahai, 1999; Lindell, 2006].

None of the previous approaches seems to yield black-box constructions under general assumptions. Indeed, our work (also [Peikert and Waters, 2008]) does not use the above framework.

2.1.2 Our Results

As mentioned earlier, we exhibit a black-box construction of a non-malleable encryption scheme from any semantically secure one, the main novelty being that our construction is black-box. While this is interesting in and of itself, our construction also compares favorably with previous work in several regards:

- *Improved parameters.* We improve on the computational complexity of previous constructions based on general assumptions. In particular, we do not have to do an NP-reduction in either encryption or decryption, although we do have to pay the price of the running time of Berlekamp-Welch for decryption. The running time incurs a multiplicative overhead that is quasi-linear in the security parameter, over the running time of the underlying CPA secure scheme. Moreover, the sizes of public keys and ciphertext are independent of the computational complexity of the underlying scheme.

- *Conceptual simplicity/clarity.* Our scheme (and the analysis) is arguably much simpler than many of the previous constructions, and like [Pass *et al.*, 2006], entirely self-contained (apart from the Berlekamp-Welch algorithm). We do not need to appeal to notions of zero-knowledge, nor do we touch upon subtle technicalities like adaptive vs non-adaptive NIZK. Our construction may be covered in an introductory graduate course on cryptography without requiring zero knowledge as a pre-requisite.

- *Ease of implementation.* Our scheme is easy to describe and can be easily implemented in a modular fashion.

We may also derive from our construction additional positive and negative results.

Bounded CCA2 non-malleability.

Cramer et al. [Cramer *et al.*, 2007] introduced the bounded CCA2 attack, a relaxation of the CCA2 attack wherein the adversary is only allowed make an a-priori bounded number of queries q to the decryption oracle, where q is fixed prior to choosing the parameters of the encryption scheme. In addition, starting from any semantically secure encryption, they obtained[2]:

- an encryption scheme that is semantically secure under a bounded-CCA2 attack via a black-box construction, wherein the size of the public key and ciphertext are quadratic in q; and

- an encryption scheme that is non-malleable under a bounded-CCA2 attack via a non-black-box construction, wherein the size of the public key and ciphertext are linear in q.

[2]While semantic security and non-malleability are equivalent under a CCA2 attack [Dolev *et al.*, 2000], they are not equivalent under a bounded-CCA2 attack, as shown in [Cramer *et al.*, 2007].

Combining their approach for the latter construction with our main result, we obtain an encryption scheme that is non-malleable under a bounded-CCA2 attack via a black-box construction, wherein the size of the public key and ciphertext are linear in q.

Separation between CCA2 security and non-malleability.

Our main construction has the additional property that the decryption algorithm does not query the encryption functionality of the underlying scheme. Gertner, Malkin and Myers [Gertner *et al.*, 2007] referred to such constructions as shielding and they showed that there is no shielding black-box construction of CCA1-secure encryption schemes from semantically secure encryption. Combined with the fact that any shielding construction when composed with our construction is again shielding, this immediately yields the following:

> **Corollary (informal)** There exists no shielding black-box construction of CCA1-secure encryption schemes from non-malleable encryption schemes.

Note that a CCA2-secure encryption scheme is trivially also CCA1-secure, so this also implies a separation between non-malleability and CCA2-security for shielding black-box constructions.

Our techniques.

At a high level, we follow the cut-and-choose approach for consistency checks from [Pass *et al.*, 2006], wherein the randomness used for cut-and-choose is specified in the secret key. A crucial component of our construction is a message encoding scheme with certain locally testable and self-correcting properties, based on the fact that low-degree polynomials are simultaneously good error-correcting codes and a secret-sharing scheme; this has been exploited in the early work on secure multi-party computation with malicious adversaries [Ben-Or *et al.*, 1988]. We think this technique may be useful in eliminating general NP-reductions in other constructions in cryptography (outside of public-key encryption).

Towards CCA2 Security?

The main obstacle towards achieving full CCA2 security from either semantically secure encryptions or enhanced trapdoor permutations using our approach (and also the [Pass *et al.*, 2006] approach) lies in guaranteeing soundness of the consistency checks against an adversary that can adaptively determine its queries depending on the outcome of previous consistency checks. It seems conceivable that using a non-shielding construction that uses re-encryption may help overcome this obstacle.

2.1.3 Overview of our Construction

Recall the DDN [Dolev *et al.*, 2000] and PSV [Pass *et al.*, 2006] constructions: to encrypt a message, one (a) generates k encryptions of the same message under independent keys, (b) gives a non-interactive zero-knowledge proof that all resulting ciphertexts are encryptions of the same message, and (c) signs the entire bundle with a one-time signature. It is in step (b) that we use a general NP-reduction, which in return makes the construction non-black-box. In the proof of security, we exploit that fact that for a well-formed ciphertext, we can recover the message if we know the secret key for any of the k encryptions.

How do we guarantee that a tuple of k ciphertexts are encryptions of the same plaintext without using a zero-knowledge proof and without revealing any information about the underlying plaintext? Naively, one would like to use a cut-and-choose approach (as has been previously used in [Lindell and Pinkas, 2007] to eliminate zero-knowledge proofs in the context of secure two-party computation), namely decrypt and verify that some constant fraction, say $k/2$ of the ciphertexts are indeed consistent. There are two issues with this approach:

- First, if only a constant number of ciphertexts are inconsistent, then we are unlikely to detect the inconsistency. To circumvent this problem, we could decrypt by outputting the majority of the remaining $k/2$ ciphertexts.
- The second issue is more fundamental: decrypting any of the ciphertexts will immediately reveal the underlying message, whereas it is crucial that we can enforce consistency while learning nothing about the underlying message.

We circumvent both issues by using a more sophisticated encoding of the message m based on low-degree polynomials instead of merely making k copies of the message as in the above schemes. Specifically, we pick a random degree k polynomial p such that $p(0) = m$ and we construct a $k \times 10k$ matrix such that the i'th column of the matrix comprises entirely of the value $p(i)$. To verify consistency, we will decrypt a random subset of k columns, and check that all the entries in each of these columns are the same.

- The issue that only a tiny number of ciphertexts are inconsistent is handled using the error-correcting properties of low-degree polynomials; specifically, each row of a valid encoding is a codeword for the Reed-Solomon code (and we output $\perp$ if it's far from any codeword).

- Low-degree polynomials are also good secret-sharing schemes, and learning a random subset of k columns in a valid encoding reveals nothing about the underlying message m. Encoding m using a secret-sharing scheme appears in the earlier work of Cramer et al. [Cramer *et al.*, 2007], but they do not consider redundancy or error-correction.

As before, we encrypt all the entries of the matrix using independent keys and then sign the entire bundle with a one-time signature. It is important that the encoding also provides a robustness guarantee similar to that of repeating the message k times: we are able to recover the message for a valid encryption if we can decrypt *any* row in the matrix. Indeed, this is essentially our entire scheme with two technical caveats:

- As with previous schemes, we will associate one pair of public/secret key pairs with each entry of the matrix, and we will select the public key for encryption based on the verification key of the one-time signature scheme.

- To enforce consistency, we will need a codeword check in addition to the column check outlined above. The reason for this is fairly subtle and we will highlight the issue in the formal exposition of our construction.

Decreasing ciphertext size. To encrypt an n-bit message with security parameter k, our construction yields $O(k^2)$ encryptions of n-bit messages in the underlying scheme. It is easy to

see that this may be reduced to $O(k \log^2 k)$ encryptions by reducing the number of columns to $O(\log^2 k)$.

2.1.4 Follow-up work

The encoding scheme introduced here has been used in a number of follow-up works: a black-box construction of set intersection protocols from homomorphic encryption [Dachman-Soled *et al.*, 2009], and a black-box construction of a CCA2-secure encryption scheme for strings starting from one for bits [Myers and Shelat, 2009].

2.2 Preliminaries & Definitions

Notation.

We adopt the notation used in [Pass *et al.*, 2006]. We use $[n]$ to denote $\{1, 2, \ldots, n\}$. If $\mathcal{A}$ is a probabilistic polynomial time (hereafter, ppt) algorithm that runs on input x, $\mathcal{A}(x)$ denotes the random variable according to the distribution of the output of $\mathcal{A}$ on input x. We denote by $\mathcal{A}(x; r)$ the output of $\mathcal{A}$ on input x and random coins r. Computational indistinguishability between two distributions A and B is denoted by $A \stackrel{c}{\approx} B$ and statistical indistinguishability by $A \stackrel{s}{\approx} B$.

2.2.1 Semantically Secure Encryption

Definition 2.2.1 (Encryption Scheme) *A triple* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is an encryption scheme, if* Gen *and* Enc *are ppt algorithms and* Dec *is a deterministic polynomial-time algorithm which satisfies the following property:*

> Correctness. *There exists a negligible function* $\mu(\cdot)$ *such that for all sufficiently large* k*, we have that with probability* $1 - \mu(k)$ *over* $(\mathrm{PK}, \mathrm{SK}) \leftarrow \mathsf{Gen}(1^k)$*: for all* m*,* $\Pr[\mathsf{Dec}_{\mathrm{SK}}(\mathsf{Enc}_{\mathrm{PK}}(m)) = m] = 1$.

Definition 2.2.2 (Semantic Security) *Let* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme and let the random variable* $\mathsf{IND}_b(\Pi, \mathcal{A}, k)$*, where* $b \in \{0, 1\}$*,* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *are ppt algorithms and* $k \in \mathbb{N}$*, denote the result of the following probabilistic experiment:*

$\mathsf{IND}_b(\Pi, \mathcal{A}, k)$:

$(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^k)$

$(m_0, m_1, \text{STATE}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\text{PK})$ *s.t.* $|m_0| = |m_1|$

$y \leftarrow \mathsf{Enc}_{\text{PK}}(m_b)$

$D \leftarrow \mathcal{A}_2(y, \text{STATE}_{\mathcal{A}})$

Output D

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* indistinguishable under a chosen-plaintext (CPA) attack*, or* semantically secure*, if for any ppt algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *the following two ensembles are computationally indistinguishable:*

$$\left\{\mathsf{IND}_0(\Pi, \mathcal{A}, k)\right\}_{k\in\mathbb{N}} \overset{c}{\approx} \left\{\mathsf{IND}_1(\Pi, \mathcal{A}, k)\right\}_{k\in\mathbb{N}}$$

It follows from a straight-forward hybrid argument that semantic security implies indistinguishability of multiple encryptions under independently chosen keys:

Proposition 2.2.3 *Let* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be a semantically secure encryption scheme and let the random variable* $\mathsf{mIND}_b(\Pi, \mathcal{A}, k, \ell)$*, where* $b \in \{0, 1\}$*,* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *are ppt algorithms and* $k \in \mathbb{N}$*, denote the result of the following probabilistic experiment:*

$\mathsf{mIND}_b(\Pi, \mathcal{A}, k, \ell)$:

For $i = 1, \ldots, \ell$*:* $(\text{PK}_i, \text{SK}_i) \leftarrow \mathsf{Gen}(1^k)$

$(\langle m_0^1, \ldots, m_0^\ell\rangle, \langle m_1^1, \ldots, m_1^\ell\rangle, \text{STATE}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\langle \text{PK}_1, \ldots, \text{PK}_\ell\rangle)$

s.t. $|m_0^1| = |m_1^1| = \cdots = |m_0^\ell| = |m_1^\ell|$

For $i = 1, \ldots, \ell$*:* $y_i \leftarrow \mathsf{Enc}_{\text{PK}_i}(m_b^i)$

$D \leftarrow \mathcal{A}_2(y_1, \ldots, y_\ell, \text{STATE}_{\mathcal{A}})$

Output D

then for any ppt algorithms $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *and for any polynomial* $p(k)$ *the following two ensembles are computationally indistinguishable:*

$$\left\{\mathsf{mIND}_0(\Pi, \mathcal{A}, k, p(k))\right\}_{k\in N} \overset{c}{\approx} \left\{\mathsf{mIND}_1(\Pi, \mathcal{A}, k, p(k))\right\}_{k\in N}$$

2.2.2 Non-malleable Encryption

Definition 2.2.4 (Non-malleable Encryption [Pass *et al.*, 2006]) *Let* $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme and let the random variable* $\mathsf{NME}_b(\Pi, \mathcal{A}, k, \ell)$ *where* $b \in \{0,1\}$, $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *are ppt algorithms and* $k, \ell \in \mathbb{N}$ *denote the result of the following probabilistic experiment:*

$\mathsf{NME}_b(\Pi, \mathcal{A}, k, \ell)$:

$(\text{PK}, \text{SK}) \leftarrow \mathsf{Gen}(1^k)$

$(m_0, m_1, \text{STATE}_{\mathcal{A}}) \leftarrow \mathcal{A}_1(\text{PK})$ *s.t.* $|m_0| = |m_1|$

$y \leftarrow \mathsf{Enc}_{\text{PK}}(m_b)$

$(\psi_1, \ldots, \psi_\ell) \leftarrow \mathcal{A}_2(y, \text{STATE}_{\mathcal{A}})$

Output $(d_1, \ldots, d_\ell)$ *where* $d_i = \begin{cases} \perp & \text{if } \psi_i = y \\ \mathsf{Dec}_{\text{SK}}(\psi_i) & \text{otherwise} \end{cases}$

$(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is* non-malleable under a chosen plaintext (CPA) attack *if for any ppt algorithms* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *and for any polynomial* $p(k)$*, the following two ensembles are computationally indistinguishable:*

$$\left\{\mathsf{NME}_0(\Pi, \mathcal{A}, k, p(k))\right\}_{k\in\mathbb{N}} \overset{c}{\approx} \left\{\mathsf{NME}_1(\Pi, \mathcal{A}, k, p(k))\right\}_{k\in\mathbb{N}}$$

It was shown in [Pass *et al.*, 2006] that an encryption that is non-malleable (under Definition 2.2.4) remains non-malleable even if the adversary $\mathcal{A}_2$ receives several encryptions under many different public keys (the formal experiment is the analogue of mIND for non-malleability).

2.2.3 (Strong) One-Time Signature Schemes

Informally, a (strong) one-time signature scheme $(\mathsf{GenSig}, \mathsf{Sign}, \mathsf{VerSig})$ is an existentially unforgeable signature scheme, with the restriction that the signer signs at most one message with any key. This means that an efficient adversary, upon seeing a signature on a message m of his choice, cannot generate a valid signature on a different message, or a different valid signature on the same message m. Such schemes can be constructed in a black-box way from one-way functions [Rompel, 1990; Lamport, 1979], and thus from any semantically secure encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ using black-box access only to Gen.

2.3 Construction

Given an encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, we construct a new encryption scheme $\Pi = (\mathsf{NMGen}^{\mathsf{Gen}}, \mathsf{NMEnc}^{\mathsf{Gen},\mathsf{Enc}}, \mathsf{NMDec}^{\mathsf{Gen},\mathsf{Dec}})$, summarized in Figure 4.2, and described as follows.

Polynomial encoding. We identify $\{0,1\}^n$ with the field $\mathrm{GF}(2^n)$. To encode a message $m \in \{0,1\}^n$, we pick a random degree k polynomial p over $\mathrm{GF}(2^n)$ such that $p(0) = m$ and construct a $k \times 10k$ matrix such that the i'th column of the matrix comprise entirely of the value $s_i = p(i)$ (where $0, 1, \ldots, 10k$ are the lexicographically first $10k + 1$ elements in $\mathrm{GF}(2^n)$ according to some canonical encoding). Note that $(s_1, \ldots, s_{10k})$ is both a $(k+1)$-out-of-$10k$ secret-sharing of m using Shamir's secret-sharing scheme and a codeword of the Reed-Solomon code $\mathcal{W}$, where

$$\mathcal{W} = \{\ (p(1), \ldots, p(10k) \mid p \text{ is a degree } k \text{ polynomial }\}.$$

Note that $\mathcal{W}$ is a code over the alphabet $\{0,1\}^n$ with minimum relative distance 0.9, which means we may efficiently correct up to 0.45 fraction errors using the Berlekamp-Welch algorithm.

Encryption. The public key for Π comprises $20k^2$ public keys from $\mathcal{E}$ indexed by a triplet $(i, j, b) \in [k] \times [10k] \times \{0,1\}$; there are two keys corresponding to each entry of a $k \times 10k$ matrix. To encrypt a message m, we (a) compute $(s_1, \ldots, s_{10_k})$ as in the above-mentioned polynomial encoding, (b) generate $(\mathrm{SKSIG}, \mathrm{VKSIG})$ for a one-time signature, (c) compute a $k \times 10k$ matrix $\vec{c} = (c_{i,j})$ of ciphertexts where $c_{i,j} = \mathsf{Enc}_{\mathrm{PK}_{i,j}^{v_i}}(s_j)$, and (d) sign $\vec{c}$ using SKSIG.

$$\begin{pmatrix} \mathsf{Enc}_{\mathrm{PK}_{1,1}^{v_1}}(s_1) & \mathsf{Enc}_{\mathrm{PK}_{1,2}^{v_1}}(s_2) & \cdots & \mathsf{Enc}_{\mathrm{PK}_{1,10k}^{v_1}}(s_{10k}) \\ \mathsf{Enc}_{\mathrm{PK}_{2,1}^{v_2}}(s_1) & \mathsf{Enc}_{\mathrm{PK}_{2,2}^{v_2}}(s_2) & \cdots & \mathsf{Enc}_{\mathrm{PK}_{2,10k}^{v_2}}(s_{10k}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathsf{Enc}_{\mathrm{PK}_{k,1}^{v_k}}(s_1) & \mathsf{Enc}_{\mathrm{PK}_{k,2}^{v_k}}(s_2) & \cdots & \mathsf{Enc}_{\mathrm{PK}_{k,10k}^{v_k}}(s_{10k}) \end{pmatrix}$$

Consistency Checks. A valid ciphertext in Π satisfies two properties: (1) the first row is an encryption of a codeword in $\mathcal{W}$ and (2) every column comprises k encryptions of the same plaintext. We want to design consistency checks that reject ciphertexts that are "far" from being valid ciphertexts under Π. For simplicity, we will describe the consistency checks as applied to the underlying matrix of plaintexts. The checks depend on a random subset S of k columns chosen during key generation.

COLUMN CHECK (`column-check`): We check that each of the k columns in S comprises entirely of the same value.

CODEWORD CHECK (`codeword-check`): We find a codeword w that agrees with the first row of the matrix in at least $9k$ positions; the check fails if no such w exists. Then we check that the first row of the matrix agrees with w at the k positions indexed by S.

The codeword check ensures that with high probability, the first row of the matrix agrees with w in at least $10k - o(k)$ positions. We explain its significance after describing the alternative decryption algorithm in the analysis.

Decryption. To decrypt, we (a) verify the signature and run both consistency checks, and (b) if all three checks accept, decode the codeword w and output the result, otherwise output $\perp$. Note that to decrypt we only need the $20k$ secret keys corresponding to the first row of the matrix and $2k$ secret keys corresponding to each of the k columns in S.

Note that the decryption algorithm may be stream-lined, for instance, by running the codeword check only if the column check succeeds. We choose to present the algorithm as is in order to keep the analysis simple; in particular, we will run both consistency checks independent of the outcome of the other.

2.4 Analysis

Having presented our construction, we now formally state and prove our main result:

Theorem 2.4.1 (Main Theorem, restated).
Assume there exists an encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *that is semantically secure under a CPA attack. Then there exists an encryption scheme* $\Pi = (\mathsf{NMGen}^{\mathsf{Gen}}, \mathsf{NMEnc}^{\mathsf{Gen},\mathsf{Enc}}, \mathsf{NMDec}^{\mathsf{Gen},\mathsf{Dec}})$ *that is non-malleable under a CPA attack.*

We establish the theorem (as in [Dolev *et al.*, 2000; Pass *et al.*, 2006], etc) via a series of hybrid arguments and deduce indistinguishability of the intermediate hybrid experiments from the semantic security of the underlying scheme $\mathcal{E}$ under some set of public keys Γ. To do so, we will need to implement an alternative decryption algorithm NMDec^* that is used in the intermediate experiments to simulate the actual decryption algorithm NMDec in the non-malleability experiment. We need NMDec^* to achieve two conflicting requirements:

- NMDec^* and NMDec must agree on essentially all inputs, including possibly malformed ciphertexts;
- We can implement NMDec^* without having to know the secret keys corresponding to the public keys in Γ.

Of course, designing NMDec^* is difficult precisely because NMDec uses the secret keys corresponding to the public keys in Γ.

Here is a high-level (but extremely inaccurate) description of how NMDec^* works: Γ is the set of public keys corresponding to the first row of the $k \times 10k$ matrix. To implement NMDec^*, we will decrypt the i'th row of the matrix of ciphertexts, for some $i > 1$, which the column check (if successful) guarantees to agree with the first row in most positions; error correction takes care of the tiny fraction of disagreements.

2.4.1 Alternative Decryption Algorithm NMDec^*

Let $\text{VKSIG}^* = (v_1^*, \ldots, v_k^*)$ denote the verification key in the challenge ciphertext given to the adversary in the non-malleability experiment, and let $\text{VKSIG} = (v_1, \ldots, v_k)$ denote the verification key in (one of) the ciphertext(s) generated by the adversary. First, we modify the signature check to also output $\perp$ if there is a forgery, namely $\text{VKSIG} = \text{VKSIG}^*$. Next, we modify the consistency checks (again, as applied to the underlying matrix of plaintexts) as follows:

COLUMN CHECK (`column-check`*): This is exactly as before, we check that the each of the k columns in S comprises entirely of the same value.

CODEWORD CHECK (`codeword-check`*): Let i be the smallest value such that $v_i \neq v_i^*$ (which exists since VKSIG $\neq$ VKSIG*). We find a codeword w that agrees with the i'th row of the matrix in at least $8k$ positions (note agreement threshold smaller than before); the check fails if no such w exists. Then we check that the first row of the matrix agrees with w at the k positions indexed by S.

To decrypt, run the modified signature and consistency checks, and if all three checks accept, decode the codeword w and output the result, otherwise output $\perp$. To implement the modified consistency checks and decryption algorithm, we only need the $10k$ secret keys indexed by $\overline{\text{VKSIG}^*}$ for each row of the matrix, and as before, the $2k$ secret keys corresponding to each of the k columns in S.

Remark on the Codeword Check. At first, the codeword check may seem superfluous. Suppose we omit the codeword check, and as before, define w to be a codeword that agrees with the first row in $9k$ positions and with the i'th row in $8k$ positions in the respective decryption algorithms; the gap is necessary to take into account inconsistencies not detected by the column check. Now, consider a malformed ciphertext ψ for Π where in the underlying matrix of plaintexts, each row is the same corrupted codeword that agrees with a valid codeword in exactly $8.5k$ positions. Without the codeword checks, ψ will be an invalid ciphertext according to NMDec and a valid ciphertext according to NMDec^* and can be used to distinguish the intermediate hybrid distributions in the analysis; with the codeword checks, ψ is an invalid ciphertext according to both. It is also easy to construct a problematic malformed ciphertext for the case where both agreement thresholds are set to the same value (say $9k$).

2.4.2 A Promise Problem

Recall the guarantees we would like from NMDec and NMDec^*:

- On input a ciphertext that is an encryption of a message m under Π, both NMDec and NMDec^* will output m with probability 1.

- On input a ciphertext that is "close" to an encryption of a message m under Π, both NMDec and NMDec^* will output m with the same probability (the exact probability is immaterial) and $\perp$ otherwise.
- On input a ciphertext that is "far" from any encryption, then both NMDec and NMDec^* output $\perp$ with high probability.

To quantify and establish these guarantees, we consider the following promise problem (Π_Y, Π_N) that again refers to the underlying matrix of plaintexts. An instance is a matrix of k by $10k$ values in $\{0,1\}^n \cup \perp$.

Π_Y (YES instances) — for some $w \in \mathcal{W}$, every row equals w.

Π_N (NO instances) — either there exist two rows that are 0.1-far (i.e. disagree in at least k positions), or the first row is 0.1-far from every codeword in $\mathcal{W}$ (i.e. disagree with every codeword in at least k positions).

Valid encryptions correspond to the YES instances, while NO instances will correspond to "far" ciphertexts. To analyze the success probability of an adversary, we examine each ciphertext ψ it outputs with some underlying matrix $\mathbf{M}$ of plaintexts (which may be a YES or a NO instance or neither) and show that both NMDec and NMDec^* agree on ψ with high probability. To facilitate the analysis, we consider two cases:

- If $\mathbf{M} \in \Pi_N$, then it fails the column/codeword checks in both decryption algorithms with high probability, in which case both decryption algorithms output $\perp$. Specifically, if there are two rows that are 0.1-far, then column check rejects $\mathbf{M}$ with probability $1 - 0.9^k$. On the other hand, if the first row is 0.1-far from every codeword, then the codeword check in NMDec rejects $\mathbf{M}$ with probability 1 and that in NMDec^* rejects $\mathbf{M}$ with probability at least $1 - 0.9^k$; that is, with probability $1 - 0.9^k$, both codeword checks in NMDec and NMDec^* rejects $\mathbf{M}$.
- If $\mathbf{M} \notin \Pi_N$, then both decryption algorithms always output the same answer for all choices of the set S, provided there is no forgery. Fix $\mathbf{M} \notin \Pi_N$ and a set S. The first row is 0.9-close

to codeword $w \in \mathcal{W}$ and we know in addition that every other row is 0.9-close to the first row and thus 0.8-close to w. Therefore, we will recover the same codeword w and message m whether we decode the first row within distance 0.1, or any other row within distance 0.2. This means that the codeword checks in both decryption algorithms compare the first row with the same codeword w. As such, both decryption algorithms output $\perp$ with exactly the same probability, and whenever they do not output $\perp$, they output the same message m.

2.4.3 Proof of Main Theorem

In the hybrid argument, we consider the following variants of NME_b as applied to Π, where VKSIG^* denotes the verification key in the ciphertext $y = \mathsf{NMEnc}_{\text{PK}}(m_b)$:

Experiment $\mathsf{NME}_b^{(1)}$

— $\mathsf{NME}_b^{(1)}$ proceeds exactly like NME_b, except we replace `sig-check` in NMDec with `sig-check`*:

(`sig-check`*) Verify the signature with $\mathsf{VerSig}_{\text{VKSIG}}[\mathbf{c}, \sigma]$. Output $\perp$ if the signature fails to verify or if $\text{VKSIG} = \text{VKSIG}^*$.

Experiment $\mathsf{NME}_b^{(2)}$

— $\mathsf{NME}_b^{(2)}$ proceeds exactly like NME_b except we replace NMDec with NMDec*:

$\mathsf{NMDec}^*_{\text{SK}}([\mathbf{c}, \text{VKSIG}, \sigma])$:

1. (`sig-check`*) Verify the signature with $\mathsf{VerSig}_{\text{VKSIG}}[\mathbf{c}, \sigma]$. Output $\perp$ if the signature fails to verify or if $\text{VKSIG} = \text{VKSIG}^*$.
2. Let $\mathbf{c} = (c_{i,j})$ and $\text{VKSIG} = (v_1, \ldots, v_k)$. Let i be the smallest value such that $v_i \neq v_i^*$. Compute $s_j = \mathsf{Dec}_{\text{SK}_{i,j}^{v_i}}(c_{i,j})$, $j = 1, \ldots, 10k$ and $w = (w_1, \ldots, w_{10k}) \in \mathcal{W}$ that agrees with $(s_1, \ldots, s_{10k})$ in at least $8k$ positions. If no such codeword exists, output $\perp$.

3. (`column-check`*) For all $j \in S$, check that $\mathsf{Dec}_{\mathrm{SK}_{1,j}^{v_1}}(c_{1,j}) = \mathsf{Dec}_{\mathrm{SK}_{2,j}^{v_2}}(c_{2,j}) = \cdots = \mathsf{Dec}_{\mathrm{SK}_{k,j}^{v_k}}(c_{k,j})$.
4. (`codeword-check`*) For all $j \in S$, check that $\mathsf{Dec}_{\mathrm{SK}_{1,j}^{v_1}}(c_{1,j}) = w_j$.

If all three checks accept, output the message m corresponding to the codeword w; else, output $\perp$.

Claim 2.4.2 *For $b \in \{0,1\}$, we have* $\left\{\mathsf{NME}_b(\Pi, \mathcal{A}, k, p(k))\right\} \stackrel{c}{\approx} \left\{\mathsf{NME}_b^{(1)}(\Pi, \mathcal{A}, k, p(k))\right\}$

Proof: This follows readily from the security of the signature scheme. ■

Claim 2.4.3 *For $b \in \{0,1\}$, we have* $\left\{\mathsf{NME}_b^{(1)}(\Pi, \mathcal{A}, k, p(k))\right\} \stackrel{s}{\approx} \left\{\mathsf{NME}_b^{(2)}(\Pi, \mathcal{A}, k, p(k))\right\}$

Proof: We will show that both distributions are statistically close for all possible coin tosses in both experiments (specifically, those of $\mathsf{NMGen}, \mathcal{A}$ and NMEnc) except for the choice of S in NMGen. Once we fix all the coin tosses apart from the choice of S, the output $(\psi_1, \ldots, \psi_{p(k)})$ of $\mathcal{A}_2$ are completely determined and identical in both experiments. We claim that with probability $1 - 2p(k) \cdot 0.9^k = 1 - \mathrm{neg}(k)$ over the choice of S, the decryptions of $(\psi_1, \ldots, \psi_{p(k)})$ agree in both experiments. This follows from the analysis of the promise problem in Section 2.4.2. ■

Claim 2.4.4 *For every ppt machine $\mathcal{A}$, there exists a ppt machine $\mathcal{B}$ such that for $b \in \{0,1\}$,*

$$\left\{\mathsf{NME}_b^{(2)}(\Pi, \mathcal{A}, k, p(k))\right\} \equiv \left\{\mathsf{mIND}_b(\mathcal{E}, \mathcal{B}, k, 9k^2)\right\}$$

Proof: The machine $\mathcal{B}$ is constructed as follows: $\mathcal{B}$ participates in the experiment mIND_b (the "outside") while internally simulating $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ in the experiment $\mathsf{NME}_b^{(2)}$.

- (pre-processing) Pick a random subset $S = \{u_1, \ldots, u_j\}$ of $[10k]$ and run $\mathsf{GenSig}(1^k)$ to generate $(\mathrm{SKSIG}^*, \mathrm{VKSIG}^*)$ and set $(v_1^*, \ldots, v_k^*) = \mathrm{VKSIG}^*$. Let ϕ be a bijection identifying $\{(i,j) \mid i \in [k], j \in [10k] \setminus S\}$ with $[9k^2]$.

- (key generation) $\mathcal{B}$ receives $\langle \mathrm{PK}_1, \ldots, \mathrm{PK}_{9k^2} \rangle$ from the outside and simulates NMGen as follows: for all $i \in [k], j \in [10k], \beta \in \{0,1\}$,

$$(\mathrm{PK}_{i,j}^{\beta}, \mathrm{SK}_{i,j}^{\beta}) = \begin{cases} (\mathrm{PK}_{\phi(i,j)}, \perp) & \text{if } \beta = v_i^* \text{ and } j \notin S \\ \mathsf{Gen}(1^k) & \text{otherwise} \end{cases}$$

- (message selection) Let (m_0, m_1) be the pair of messages $\mathcal{A}_1$ returns. $\mathcal{B}$ then chooses k random values $(\gamma_{u_1}, \ldots, \gamma_{u_k}) \in \{0,1\}^n$ and computes two degree k polynomials p_0, p_1 where p_β interpolates the $k+1$ points $(0, m_\beta), (u_1, \gamma_{u_1}), \ldots, (u_k, \gamma_{u_k})$ for $\beta \in \{0,1\}$. $\mathcal{B}$ sets $m_\beta^{\phi(i,j)} = p_\beta(j)$, for $i \in [k], j \in [10k] \setminus S$ and forwards $(\langle m_0^1, \ldots, m_0^{9k^2} \rangle, \langle m_1^1, \ldots, m_1^{9k^2} \rangle)$ to the outside.

- (ciphertext generation) $\mathcal{B}$ receives $\langle y_1, \ldots, y_{9k^2} \rangle$ from the outside (according to the distribution $\mathsf{Enc}_{\mathrm{PK}_1}(m_b^1), \ldots, \mathsf{Enc}_{\mathrm{PK}_{9k^2}}(m_b^{9k^2})$) and generates a ciphertext $[\mathbf{c}, \mathrm{VKSIG}^*, \sigma]$ as follows:

$$c_{i,j} = \begin{cases} y_{\phi(i,j)} & \text{if } j \notin S \\ \mathsf{Enc}_{\mathrm{PK}_{i,j}^{v_i^*}}(\gamma_j) & \text{otherwise} \end{cases}$$

  $\mathcal{B}$ then computes the signature $\sigma \leftarrow \mathsf{Sign}_{\mathrm{SKSIG}^*}(\mathbf{c})$ and forwards $[\mathbf{c}, \mathrm{VKSIG}^*, \sigma]$ to $\mathcal{A}_2$. It is straight-forward to verify that $[\mathbf{c}, \mathrm{VKSIG}^*, \sigma]$ is indeed a random encryption of m_b under Π.

- (decryption) Upon receiving a sequence of ciphertexts $(\psi_1, \ldots, \psi_{p(k)})$ from $\mathcal{A}_2$, $\mathcal{B}$ decrypts these ciphertexts using NMDec^* as in $\mathsf{NME}_b^{(2)}$. Note that to simulate NMDec^*, it suffices for $\mathcal{B}$ to possess the secret keys $\{\mathrm{SK}_{i,j}^{\beta} \mid \beta = 1 - v_i^* \text{ or } j \in S\}$, which $\mathcal{B}$ generated by itself.

■

Combining the three claims, we conclude that for every ppt adversary $\mathcal{A}$, there is a ppt adversary $\mathcal{B}$ such that for $b \in \{0,1\}$,

$$\begin{aligned} \left\{\mathsf{NME}_b(\Pi, \mathcal{A}, k, p(k))\right\} &\overset{c}{\approx} \left\{\mathsf{NME}_b^{(1)}(\Pi, \mathcal{A}, k, p(k))\right\} \\ &\overset{s}{\approx} \left\{\mathsf{NME}_b^{(2)}(\Pi, \mathcal{A}, k, p(k))\right\} \equiv \left\{\mathsf{mIND}_b(\mathcal{E}, \mathcal{B}, k, 9k^2)\right\} \end{aligned}$$

By Prop 2.2.3, $\mathsf{mIND}_0(\mathcal{E}, \mathcal{B}, k, 9k^2) \stackrel{c}{\approx} \mathsf{mIND}_1(\mathcal{E}, \mathcal{B}, k, 9k^2)$, which concludes the proof of Theorem 2.4.1.

2.4.4 Achieving Bounded-CCA2 Non-Malleability

We sketch how our scheme may be modified to achieve non-malleability under a bounded-CCA2 attack. Here, we allow the adversary to query Dec at most q times in the non-malleability experiment (but it must not query Dec on y). The modification is the straight-forward analogue of the [Cramer *et al.*, 2007] modification of the [Pass *et al.*, 2006] scheme: we increase the number of columns in the matrix from $10k$ to $80(k+q)$, and the degree of the polynomial p and the size of S from k to $8(k+q)$, and propagate the changes accordingly. The analysis is basically as before, except for the following claim (where $\mathsf{NME}-\mathsf{q}-\mathsf{CCA}_b^{(1)}, \mathsf{NME}-\mathsf{q}-\mathsf{CCA}_b^{(2)}$ are the respective analogues of $\mathsf{NME}_b^{(1)}, \mathsf{NME}_b^{(1)}$):

Claim 2.4.5 *For $b \in \{0,1\}$, we have*

$$\left\{\mathsf{NME}-\mathsf{q}-\mathsf{CCA}_b^{(1)}(\Pi, \mathcal{A}, k, p(k))\right\} \stackrel{s}{\approx} \left\{\mathsf{NME}-\mathsf{q}-\mathsf{CCA}_b^{(2)}(\Pi, \mathcal{A}, k, p(k))\right\}$$

Proof:[sketch] As before, we will show that both distributions are statistically close for all possible coin tosses in both experiments (specifically, those of NMGen, $\mathcal{A}$ and NMEnc) except for the choice of S in NMGen. However, we cannot immediately deduce that the output of $\mathcal{A}_2$ are completely determined and identical in both experiments, since they depend on the adaptively chosen queries to NMDec, and the answers depend on S. Instead, we will consider all 2^q possible computation paths of $\mathcal{A}$ which are determined based on the q query/answer pairs from NMDec. For each query, we consider the underlying matrix of plaintexts $\mathbf{M}$:

- If $\mathbf{M} \in \Pi_N$, then we assume NMDec returns $\perp$.
- If $\mathbf{M} \notin \Pi_N$, then we consider two branches depending on the two possible outcomes of the consistency checks.

We claim that with probability $1 - 2^q \cdot p(k) \cdot 0.9^{8(k+q)} > 1 - \mathrm{neg}(k)$ over the choice of S, the decryptions of $(\psi_1, \ldots, \psi_{p(k)})$ agree in both experiments in all 2^q computation paths. ■

Remark on achieving (full) CCA2 security. It should be clear from the preceding analysis that the barrier to obtaining full CCA2 security lies in handling queries outside Π_N. Specifically, with even just a (full) CCA1 attack, an adversary could query NMDec on a series of adaptively chosen ciphertexts corresponding to matrices outside Π_N to learn the set S upon which it could readily break the security of our construction.

$\mathsf{NMGen}(1^k)$:

1. For $i \in [k], j \in [10k], b \in \{0,1\}$, run $\mathsf{Gen}(1^k)$ to generate key-pairs $(\text{PK}^b_{i,j}, \text{SK}^b_{i,j})$.
2. Pick a random subset $S \subset [10k]$ of size k.

Set $\text{PK} = \left\{(\text{PK}^0_{i,j}, \text{PK}^1_{i,j}) \mid i \in [k], j \in [10k]\right\}$ and $\text{SK} = \left\{S, (\text{SK}^0_{i,j}, \text{SK}^1_{i,j}) \mid i \in [k], j \in [10k]\right\}$.

$\mathsf{NMEnc}_{\text{PK}}(m)$:

1. Pick random $\alpha_1, \ldots, \alpha_k \in \text{GF}(2^n)$ and set $s_j = p(j), j \in [10k]$ where $p(x) = m_0 + \alpha_1 x + \ldots + \alpha_k x^k$.
2. Run $\mathsf{GenSig}(1^k)$ to generate $(\text{SKSIG}, \text{VKSIG})$. Let $(v_1, \ldots, v_k)$ be the binary representation of VKSIG.
3. Compute the ciphertext $c_{i,j} \leftarrow \mathsf{Enc}_{\text{PK}^{v_i}_{i,j}}(s_j)$, for $i \in [k], j \in [10k]$.
4. Compute the signature $\sigma \leftarrow \mathsf{Sign}_{\text{SKSIG}}(\vec{c})$ where $\vec{c} = (c_{i,j})$.

Output the tuple $[\vec{c}, \text{VKSIG}, \sigma]$.

$\mathsf{NMDec}_{\text{SK}}([\vec{c}, \text{VKSIG}, \sigma])$:

1. (`sig-check`) Verify the signature with $\mathsf{VerSig}_{\text{VKSIG}}[\vec{c}, \sigma]$.
2. Let $\vec{c} = (c_{i,j})$ and $\text{VKSIG} = (v_1, \ldots, v_k)$. Compute $s_j = \mathsf{Dec}_{\text{SK}^{v_1}_{1,j}}(c_{1,j})$, $j = 1, \ldots, 10k$ and the codeword $w = (w_1, \ldots, w_{10k}) \in \mathcal{W}$ that agrees with $(s_1, \ldots, s_{10k})$ in at least $9k$ positions. If no such codeword exists, output $\perp$.
3. (`column-check`) For all $j \in S$, check that $\mathsf{Dec}_{\text{SK}^{v_1}_{1,j}}(c_{1,j}) = \mathsf{Dec}_{\text{SK}^{v_2}_{2,j}}(c_{2,j}) = \cdots = \mathsf{Dec}_{\text{SK}^{v_k}_{k,j}}(c_{k,j})$.
4. (`codeword-check`) For all $j \in S$, check that $s_j = w_j$.

If all three checks accept, output the message m corresponding to the codeword w; else, output $\perp$.

Figure 2.2: THE NON-MALLEABLE ENCRYPTION SCHEME Π

Chapter 3

On the Black-Box Complexity of Optimally-Fair Coin Tossing

3.1 Introduction

We study the fundamental problem of (two-party) coin tossing, where two mutually distrustful parties wish to generate a common random bit. Ideally, this bit should be almost completely unbiased (namely be equal to 1 with probability that is at most negligibly far from $1/2$). Furthermore, by the definition of a secure coin tossing protocol, if the two parties follow the protocol then they must both output the same random bit. Unfortunately, however, as shown in a classic result by Cleve [Cleve, 1986], if one of the parties may deviate from the protocol (even if the deviation is only "fail-stop" meaning that the adversary merely aborts early), then secure coin tossing cannot be achieved. In fact, Cleve proved that for any coin tossing protocol running for r rounds there exists an efficient fail-stop adversary that can bias the resulting bit by at least $\Omega(1/r)$.

On the positive side, an early result by Blum [Blum, 1981] uses one-way functions to construct a coin tossing protocol in a weaker model, where an unbiased output is achieved if both parties complete the protocol, but if a malicious party aborts early, the honest party does not output any bit. This protocol was used by Cleve [Cleve, 1986] to construct a coin tossing protocol that runs for r rounds and for which no efficient adversary can bias the output bit by more than $O(1/\sqrt{r})$ assuming

that one-way functions exist.[1]

This gap between the lower and upper bounds in [Cleve, 1986] remained open for more than two decades. Recently, it was closed by Moran et al. [Moran *et al.*, 2009], who constructed a protocol for coin tossing that matches the lower-bound of [Cleve, 1986]. Specifically, they constructed an $O(r)$-round protocol with the property that no adversary can bias the output by more than $O(1/r)$. Thus, they demonstrated that the $\Omega(1/r)$ lower-bound is tight. We call such a protocol *optimally-fair* because no protocol can achieve lower bias.

Interestingly, the protocol of [Moran *et al.*, 2009] uses general secure computation and thus requires the assumption that oblivious transfer exists (or any assumption implying it, like enhanced trapdoor permutations). We note that by combining the works of [Kilian, 1988; Haitner *et al.*, 2011; Pass and Wee, 2009], it is possible to construct constant-round protocols for secure computation of any functionality in a black-box manner from semi-honest oblivious transfer. Thus, the result of [Moran *et al.*, 2009] implies a black-box construction of optimally-fair coin tossing from oblivious transfer. In contrast, the coin tossing protocol of Blum [Blum, 1981] and the protocol of [Cleve, 1986] achieving bias of $O(1/\sqrt{r})$ can be constructed in a black-box manner from any one-way function. This disparity was observed by [Moran *et al.*, 2009] who state: *"A challenging problem is to either achieve the optimal bias based on seemingly weaker assumptions (e.g., one-way functions), or to demonstrate that oblivious transfer is in fact essential."*

In this paper we take a step toward answering this question, and show that one-way functions are not sufficient for achieving optimally-fair coin tossing via *black-box reductions* when the number of rounds r is $o(n/\log n)$ for security parameter n (i.e., the input/output length of the one-way function). We note that the protocols mentioned above of [Cleve, 1986; Moran *et al.*, 2009] are indeed black-box

Theorem 3.1.1 (Main Theorem, Informal) *Let Π be a black-box construction for two-party optimally-fair coin tossing based on one-way functions with input and output length n. Then the number of rounds r of interaction in Π is at least $r = \Omega(n/\log n)$.*

Thus, we rule out black-box constructions of optimally-fair coin tossing secure against semi-

[1] Essentially, this protocol works by running Blum's protocol r times sequentially and outputting the bit that appeared in most executions. (If one of the parties halts prematurely, then the other party takes locally chosen uniformly distributed bits as the output bits for the remaining Blum executions.)

honest, fail-stop adversaries (with $o(n/\log n)$ rounds) from one-way functions. In contrast, the positive result of [Moran *et al.*, 2009] is a black-box construction of optimally-fair coin tossing secure against malicious adversaries from oblivious transfer.

In fact, we prove something even stronger:

- **Stronger primitives.** The same result holds even if the primitive used in the construction is an exponentially-hard one-way function or an exponentially hard collision resistant hash function $h\colon \{0,1\}^n \mapsto \{0,1\}^{\theta(n)}$ (or in fact any primitive which can be derived in a black-box manner from a random oracle). The result holds also for more structured primitives such as one-way permutation. The latter extension is based on the simple observation that a random function and a random permutation can not be distinguished with "few" queries asked by the construction.

- **Optimality of the bias.** The same result holds even when Π achieves any $o(1/\sqrt{r})$ bias (not only for optimally-fair protocols with a bias of $O(1/r)$).

Our main technical lemma in order to prove Theorem 3.1.1 is to show how to remove random oracles from certain secure protocols in the random oracle models which we believe to be of independent interest.

Lemma 3.1.2 (Simulation Lemma, Informal) *Let Π be a two-party protocol in the random oracle model in which the parties query a (random) oracle of input/output length n, ask a total of $m = \mathrm{poly}(n)$ queries and communicate for $o(n/\log n)$ rounds. Then there are two protocols: Π_E (the extended protocol) and Π_T (the threshold-simulation protocol) such that the following holds.* **(a)** *In Π_E the parties act as Π but the ask up to $2^{o(n)}$ extra queries from the oracle.* **(b)** *Π_T is performed in the plain model without the random oracle.* **(c)** *The joint views of the parties in Π_E and Π_T are λ-close for an arbitrary parameter $\lambda = 1/\mathrm{poly}(n)$.*

The high level structure of the proof of Theorem 3.1.1 is to use the simulation lemma and the result of [Cleve and Impagliazzo, 1993] which breaks any coin-tossing protocol in the plain model with "few" rounds. See Section 3.1.1 for more details.

We also observe that our simulation lemma can be used to derive impossibility results in the context of secure two-party computation of non-trivial functions. Kushilevitz [Kushilevitz, 1992]

classified the finite functions that have perfectly secure two-party protocols against semi-honest adversaries and called them "decomposable functions". Maji, Prabhakaran and Rosulek [Maji *et al.*, 2009] extended this result to the regime of statistical security and showed that only decomposable functions can have (randomized) two-party protocols which are statistically secure against semi-honest parties. The latter result together with our simulation lemma imply that if a function is not decomposable, it can not have a black-box secure protocol based on one-way function (or based on the other primitives mentioned above) with $o(n/\log n)$ rounds of communication. The steps of the proof of this result are very similar to the case of coin-tossing described in Theorem 3.1.1. See Section 3.1.1 for more details.

Discussion and Implications. Our lower-bound proves that either there is no black-box construction of optimally-fair coin tossing from any of the primitives mentioned in Theorem 3.4.1, or if there is any such construction it will suffer from an almost linear $\widetilde{\Omega}(n)$ lower-bound on its round-complexity (which arguably is the main efficiency measure) depending on the security parameter of the primitive used. We leave open the question of whether there exists an $\widetilde{\Omega}(n)$-round optimally fair coin-tossing protocol from one-way function (or any of the other primitives mentioned in Theorem 3.4.1). However, such a construction, where the number of rounds, and thus the bias, must depend on the security parameter, seems counter-intuitive and is unlike the known coin-tossing protocols of [Cleve, 1986; Moran *et al.*, 2009] (yet see the comparison below with statistically hiding commitments which do have constructions with the number of rounds depending on the security parameter).

In particular, our negative result implies that the use of oblivious transfer (as an assumption stronger than one-way function) in the construction of [Moran *et al.*, 2009], achieving $O(1/r)$ bias for any r, is *inherent*. Moreover, the construction of [Cleve, 1986], using commitments (that can be constructed in a black-box way from one-way functions) and achieving $O(1/\sqrt{r})$ bias for any r, is actually *optimal* (as Theorem 3.4.1 holds for any $o(1/\sqrt{r})$ bias).

It is also interesting to contrast our lower bound with the original impossibility result of Cleve [Cleve, 1986]. One way to view the result of [Cleve, 1986] is as a proof that in order to achieve $O(1/r)$ bias any protocol must have at least $\Omega(r)$ rounds of interaction. Our lower bound then says that it is only possible to achieve $O(1/r)$ bias with r rounds when relying on one-way functions

(or any of the primitives mentioned in Theorem 3.4.1) for $r = \Omega(n/\log n)$ which is very large. In particular, it is not possible to construct a protocol (using a black-box reduction) whose round efficiency depends *only* on the desired bias and is independent of the security parameter n used to determine the input length to the one-way function. This has the ramification that increasing the security parameter in order to obtain a stronger guarantee of invertibility of the one-way function (to get a more secure protocol) has an effect also on the round complexity of the protocol.

Our work also leaves open the question of whether there exists an optimally fair coin-tossing protocol based on semantically-secure public-key encryption (PKE) for $r = o(n/\log n)$ number of rounds. This is an interesting question since there is no black-box reduction from oblivious transfer to public key encryption [Gertner *et al.*, 2000] or from public key encryption to one-way function [Impagliazzo and Rudich, 1989]. This tells us that, with respect to black-box reductions, public key encryption is a strictly stronger primitive than one-way function but a strictly weaker primitive than oblivious transfer. Thus, we may ask whether the [Moran *et al.*, 2009] positive result relies on the full power of multi-party computation (which is implied by oblivious transfer but not by public key encryption) or whether it merely exploits the ability to do key agreement (which is implied by public key encryption but not by one-way function).

Black-Box Separations. One of the main goals of modern cryptography has been to identify the minimal assumptions necessary to construct secure cryptographic primitives. For example, [Yao, 1982; Goldwasser and Micali, 1984; Rompel, 1990; Håstad *et al.*, 1999; Goldreich *et al.*, 1986; Luby and Rackoff, 1988; Impagliazzo and Luby, 1989; Naor and Yung, 1989; Naor, 1991] have shown that private key encryption, pseudorandom generators, pseudorandom functions and permutations, bit commitment, and digital signatures exist if and only if one-way functions exist. On the other hand, some cryptographic primitives such as public key encryption, oblivious transfer, and key agreement are not known to be equivalent to one way functions. Thus, it is natural to ask whether the existence of one-way functions implies these primitives. However, it seems unclear how to formalize such a question; since it is widely believed that both one-way functions and public key encryption exist, this would imply in a trivial logical sense that the existence of one-way functions implies the existence of public key encryption. Thus, we can only hope to rule out restricted types of constructions that are commonly used to prove implications in cryptography.

Impagliazzo and Rudich [Impagliazzo and Rudich, 1989] were the first to develop a technique to rule out the existence of an important class of reductions between primitives known as black-box reductions. Intuitively, this is a reduction where the primitive is treated as an oracle or a "black-box". There are actually several flavors of black-box reductions (fully black-box, semi black-box and weakly black-box [Reingold *et al.*, 2004]). In our work, we only deal with fully black-box reduction, and so we will focus on this notion here. Informally, a fully black-box reduction from a primitive $\mathcal{Q}$ to a primitive $\mathcal{P}$ is a pair of *oracle* PPT Turing machines $(G, \mathcal{S})$ such that the following two properties hold:

Correctness: For every implementation f of primitive $\mathcal{P}$, $g = G^f$ implements $\mathcal{Q}$.
Security: For every implementation f of primitive $\mathcal{P}$, and every adversary $\mathcal{A}$, if $\mathcal{A}$ breaks G^f (as an implementation of $\mathcal{Q}$) then $\mathcal{S}^{\mathcal{A},f}$ breaks f. (Thus, if f is "secure", then so is G^f.)

We remark that an *implementation* of a primitive is any specific scheme that meets the requirements of that primitive (e.g., an implementation of a public-key encryption scheme provides samplability of key pairs, encryption with the public-key, and decryption with the private key). Correctness thus states that when G is given oracle access to any valid implementation of $\mathcal{P}$, the result is a valid implementation of $\mathcal{Q}$. Furthermore, security states that any adversary breaking G^f yields an adversary breaking f. The reduction here is *fully* black-box in the sense that the adversary $\mathcal{S}$ breaking f uses $\mathcal{A}$ in a black-box manner.

Comparison to Similar Lower-Bounds on the Round-Complexity. The only similar lower-bound on the round-complexity of black-box constructions that we are aware of is the result of Haitner, Hoch, Reingold, and Segev [Haitner *et al.*, 2007] which deals with the round-efficiency of statistically hiding commitment schemes. Interestingly, our lower-bound is exactly the same as that of [Haitner *et al.*, 2007] which also is based on the security parameter of the one-way function used in the construction . It seems that the techniques used in [Haitner *et al.*, 2007] and our techniques explained below are quite different. This raises the question of whether there are more connections between the two results. For instance, is it possible to simplify any of these arguments using ideas from the other work? More importantly, this suggests the intriguing possibility that perhaps a positive solution for optimally-fair coin tossing from one-way functions can be achieved

with $O(n/\log n)$ rounds, using the techniques which are used in constructing the positive results of $O(n/\log n)$-round statistically hiding commitments [Naor *et al.*, 1998; Haitner and Reingold, 2007; Haitner *et al.*, 2009]. We note, however, that there is a black-box construction of collision-resistant hash functions (CRHF) in the random oracle model. This implies that our black-box separation extends also to constructions of optimally-fair coin tossing from CRHF. In contrast, since there exists an $O(1)$-round protocol for statistically-hiding commitment from CRHF, the [Haitner *et al.*, 2007] result clearly does not extend to constructions from CRHF.

3.1.1 Our Technique

We recall a result of Cleve and Impagliazzo [Cleve and Impagliazzo, 1993] which shows that for any coin tossing protocol with r rounds, there exists a *computationally unbounded* adversary who can achieve bias of at least $\Omega(1/\sqrt{r})$ (see Appendix A.1). Moreover, this adversary follows the protocol as specified, except that it may abort prematurely; as such the adversary is fail-stop. We show that a black-box construction of an $o(n/\log n)$-round coin tossing from own-way functions with input/output length n (or in fact any primitive which is implied by a random-function in a black-box way) will essentially suffer from the same attack of [Cleve and Impagliazzo, 1993] and thus cannot guarantee any bias below $\Omega(1/\sqrt{r})$ through a black-box proof of security.

We start by assuming that there is a black-box construction Π of optimally-fair coin tossing from one-way function with $r = o(n/\log n)$ rounds. A random function is one-way with overwhelming probability, so informally speaking, if we feed the construction Π with a random function it should still be an optimally-fair coin tossing protocol. In fact, something stronger happens when a construction based on one-way function is fed with a random function: Such a construction will now be secure even against computationally *unbounded* adversaries who are allowed to ask $2^{o(n)}$ oracle queries to the random oracle. The reason for this is that if there were such an adversary, then the security reduction will imply that there is an adversary inverting a random function with $2^{o(n)}$ number of queries (see the proof of Theorem 3.4.1 for more details) which is not possible. We will take advantage of this stronger property to derive the contradiction by presenting a $2^{o(n)}$-query attack whenever the round complexity is $o(n/\log n)$. The idea of feeding a black-box construction with a random-function and enhancing its security, and then deriving contradiction by a simple counting argument (rather than refuting the relativizing reductions [Impagliazzo and

Rudich, 1989]—which is a much harder task) is also employed in previous works such as [Gennaro *et al.*, 2005; Barak and Mahmoody-Ghidary, 2007].

Our main technical step will be to show that the round-complexity of $o(n/\log n)$ for the black-box construction of coin tossing implies the existence of a $2^{o(n)}$-query adversary who is able to bias the output bit by $\omega(1/r)$. In fact we show how to achieve bias $\Omega(1/\sqrt{r}) = \omega(1/r)$. The existence of such an attack implies the result because by the security reduction the ability to bias the protocol yields an adversary inverting the one-way function. Our $2^{o(n)}$-query attacker runs the protocol (of the corresponding party) honestly except that it gathers more information about the random oracle along the execution of the protocol by asking $\mathrm{poly}(n,r)^r$ (which is $2^{o(n)}$ for $r = o(n/\log n)$) more queries and achieves bias of $\Omega(1/\sqrt{r})$ by deciding to stop at some point during the protocol.

We shall emphasize that the reason that we can *not* directly use the attack of [Cleve and Impagliazzo, 1993] in the presence of a random oracle is that, even conditioned on the transcript of the interaction, the random oracle builds *dependencies* between the views of Alice and Bob. However the attack of [Cleve and Impagliazzo, 1993] essentially uses the fact that conditioned on the transcript the views of Alice and Bob are independent in a plain protocol (where no random oracle is used). Thus we need to find a way to "kill" this dependency to be able to use their attack.

Our $2^{o(n)}$-query attacker uses special properties of an attack given by Barak and Mahmoody [Barak and Mahmoody, 2009] to break any key-agreement protocol with an optimal number of queries to the random oracle. The attacker of [Barak and Mahmoody, 2009]—which here we call the "independence learning algorithm", or the simply the learning algorithm for short—gets as input a threshold parameter ϵ which controls its efficiency and accuracy at the same time. Roughly speaking if Alice and Bob ask m oracle queries in their execution, it will lead to $O(m/\epsilon)$ queries asked by the learner and the error of $m\epsilon$. This learning algorithm can be described more naturally as an *online* algorithm which learns certain oracle queries during the interaction between Alice and Bob (despite the fact that passive adversaries can always wait till the end of the interaction). Our attacker uses this learning algorithm internally and feeds it with *different* values for the threshold parameter ϵ for each round; the parameter ϵ taken grows exponentially with the round numbers. Due to the heavy use of the threshold parameter of the learning algorithm in our attack, we call it the "threshold attacker" TA. Note that since the learning algorithm only requires the knowledge of the public transcripts, both Alice and Bob can run the learning algorithm in any two-party protocol

(e.g., a coin tossing protocol rather than a key-agreement protocol). Thus our threshold attacker TA, which is in fact executed by either Alice or Bob, can also run the learning algorithm during the coin tossing protocol.

The Threshold Attacker—More Details. For an arbitrary two-party protocol Π in the random oracle model (or any other oracle model) we can think of "curious" parties who run the protocol honestly but will ask more oracle queries along their execution of the protocol[2]. We use the terminology of [Goyal *et al.*, 2010] and call such a game a *curious extension* of the original protocol Π. To get the threshold attacker, Alice or Bob (whoever is performing the attack) will need to play a curious extension of the original protocol by asking up to $2^{o(n)}$ oracle queries. Here we will only deal with an extension based on the learning algorithm of [Barak and Mahmoody, 2009]. That is, the attacking party runs the learning algorithm along the honest execution of the original coin-tossing protocol and decides to abort prematurely. We let the parties take turn in simulating the learning algorithm in the following way: Whenever Alice (or Bob) is sending a message w_i, they attach to it the set of query/answer pairs that the learning algorithm would learn after w_i is sent across the channel. For brevity we call this specific curious extension in which both Alice and Bob run the learning algorithm along the original game (and attach the learner's view of each round to their messages) simply "the extended execution" of the original protocol (without referring to the learning algorithm explicitly). We show how our threshold attacker can perform their attack in the extended execution.

We prove that the extended protocol has the interesting property that now Alice and Bob can in fact "simulate" the random oracle on their own (using their private randomness) in a way that their views are statistically close to those in the execution of the original extended game in the random oracle model. To perform the simulation, Alice and Bob will answer their queries to the random oracle using fresh randomness unless they have asked this query at some point before (and thus chose the answer already) or that they are told by the other party what the answer to this query should be (through the extra messages simulating the learner's view).

To prove that the above simple simulation is indeed a statistically-close simulation of the

[2]This is slightly different from the semi-honest parties who run the protocol honestly without asking more oracle queries and only later analyze their view of the interaction.

extension game we need to show that (unless with small probability) there is no inconsistencies between the oracle answers chosen by Alice and Bob for their oracle queries. Here we crucially use the fact that the learning algorithm provides enough information along the game so that Alice and Bob will always choose consistent oracle answers for their queries. Suppose that Alice is sending a message w_i and is also attaching a list of $k \approx m/\epsilon_i$ simulated learning queries to the message w_i where ϵ_i is the learner's threshold used in round i by Alice and m is the total number of queries in the original protocol. For any query q among these k queries which are being asked by Alice from the random oracle (and thus being simulated) for the first time, we want that q is *not* among Bob's "private" queries which was simulated at some point before (yet is not announced through the learner's simulation). The learner's algorithm has the property that if Bob uses threshold ϵ_{i-1} to simulate the learner in the previous round $i-1$ then any such query q has chance of at most ϵ_{i-1} to be a "private" query of Bob. Therefore, by a union bound, the probability that any of these k queries cause an inconsistency is at most $\approx k\epsilon_{i-1} = m\epsilon_{i-1}/\epsilon_i$. By taking $\epsilon_{i-1} \ll \epsilon_i/m$, we can control the probability of such event to be arbitrary small. This clarifies why we end up using exponentially smaller thresholds for smaller rounds.

Finally, since we could simulate the extended execution through a plain protocol, we can use the *inefficient* attack of [Cleve and Impagliazzo, 1993], which can be applied to any plain protocol and apply it to the simulation of the extension game. Since the extended execution and its simulation are statistically close experiments, we conclude that almost the same bias would be achieved by the attacker in the extension execution with only $2^{o(n)}$ queries and so we are done.

A Parallel Work. The threshold simulation technique was discovered independently in a parallel work by Maji and Prabhakaran [Maji and Prabhakaran, 2010] in the context of using random oracle for the aim of achieving statistically secure protocols.

3.2 Definitions and Useful Lemmas

Definition 3.2.1 (coin tossing from one-way function) *For (interactive) oracle algorithms* A, B *we call* $\Pi = \langle \mathsf{A}, \mathsf{B} \rangle$ *a black-box construction of coin tossing with bias at most* δ *based on exponentially-hard one-way functions with security parameter* n*, if the following properties hold:*

- A *and* B *have their own private randomness* $R_{\mathsf{A}}, R_{\mathsf{B}}$*. They take as input* 1^n *and run in time* $\mathrm{poly}(n)$ *and interact for* $r(n) = \mathrm{poly}(n)$ *number of rounds.*

- **Completeness:** *For any function* $f\colon \{0,1\}^n \mapsto \{0,1\}^n$*, when* A *and* B *are given oracle access to* f*, then at the end of the protocol* A*'s output* a *and* B*'s output* b *are such that* $a = b$ *and* b *is considered the output of the protocol. Also if during the protocol* A *(resp.,* B*) receives the special message* $\perp$ *(denoting that the other party has stopped playing in the protocol) then* A *(resp.,* B*) outputs a bit* a *(resp* b*) on their own which is considered as the output of the protocol.*

- **Security (against bias** δ**):** *There is an oracle algorithm* S *running in time* $2^{o(n)}$ *with the following property. For any* $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ *given as oracle, if* $\widehat{\mathsf{A}}$ *(resp.,* $\widehat{\mathsf{B}}$*) is a malicious interactive algorithm interacting with* B *(resp.,* A*) which makes the output bit* b *to be* $\delta(n)$*-biased, then* $S^{f,\widehat{\mathsf{A}}}$ *(given oracle access to* f *and* $\widehat{\mathsf{A}}$*) breaks the security of* f *(as an exponentially-hard one-way function).*

We denote by $(a|b) \leftarrow \langle \widehat{\mathsf{A}}, \mathsf{B} \rangle$ *(resp.* $(a|b) \leftarrow \langle \mathsf{A}, \widehat{\mathsf{B}} \rangle$*) the joint output of* $\widehat{\mathsf{A}}$ *and* B *(resp.* A *and* $\widehat{\mathsf{B}}$*) generated by an interaction of* $\widehat{\mathsf{A}}$ *and* B *(resp.* A *and* $\widehat{\mathsf{B}}$*).*

The proof of the following two lemmas can be verified by inspection.

Lemma 3.2.2 (Inverting Random Functions) *Let* $\mathcal{A}$ *be a* computationally unbounded *oracle algorithm given oracle access to a random function* $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ *(the randomness of* f *is chosen after* $\mathcal{A}$ *is fixed). Then if* $\mathcal{A}$ *asks at most* $2^{\alpha n}$ *queries from* f*, the probability that* $\mathcal{A}$ *can successfully invert a given input* $y = f(U_n)$ *(to any preimage of* y*) is at most* $2 \cdot 2^{(\alpha-1)n} + 2^{-n}$ *which is negligible for any constant* $\alpha < 1$*.*

Lemma 3.2.3 (Inverting Random Functions with a Fixed Subdomain) *Let* $S \subset \{0,1\}^n$ *be of size* $|S| \leq 2^{\beta n}$ *for* $\beta < 1$*, and let* $f_S\colon S \mapsto \{0,1\}^n$ *be a fixed function. Let* F *be the set of all functions* $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ *which are equal to* f_S *over* S*. Now, let* $\mathcal{A}$ *be a* computationally unbounded *oracle algorithm which can depend on* f_S *and is given oracle access to a random function* $f \leftarrow_R F$ *(the randomness of* f *is chosen after* $\mathcal{A}$ *is fixed). Then if* $\mathcal{A}$ *asks at most* $2^{\alpha n}$ *queries from* f*, the probability that* $\mathcal{A}$ *can successfully invert a given input* $y = f(U_n)$ *(to any*

preimage of y) is at most $2 \cdot (2^{(\alpha-1)n} + 2^{(\beta-1)n}) + 2^{-n}$ which is negligible for any constants $\alpha < 1$ and $\beta < 1$.

3.3 Simulation Lemma

In this section, we present a general lemma that holds for any two-party protocol in the random oracle model. This lemma will be useful for proving our result on coin tossing, but also has applications to general two-party computation as we describe below.

Lemma 3.3.1 (Simulation Lemma) *Let Π be a two-party protocol between Alice and Bob in the random oracle model where they ask at most m oracle queries and interact for r rounds. Then there exist protocols Π_T and Π_E called the λ-threshold simulation and λ-extended execution of Π such that the views of Alice and Bob (as a jointly distributed random variable) in Π_T and Π_E are λ-close. Moreover, the following properties hold:*

- *Π_T makes no oracle queries.*
- *For $\lambda = 1/\operatorname{poly}(n)$, $r = o(n/\log n)$ and $m = \operatorname{poly}(n)$, Π_E makes at most $2^{o(n)}$ queries.*
- *Let $W^{\Pi} = [w_1^{\Pi}, \ldots, w_i^{\Pi}]$ be the sequence of messages sent between Alice and Bob so far in an execution of protocol Π relative to oracle f with random tapes $R_{\mathsf{A}}, R_{\mathsf{B}}$ respectively. For $\lambda = 1/\operatorname{poly}(n)$, $r = o(n/\log n)$ and $m = \operatorname{poly}(n)$, both Alice and/or Bob can make at most $2^{o(n)}$ queries and produce the transcript $W^{\Pi_E} = [w_1^{\Pi_E}, \ldots, w_i^{\Pi_E}]$ that is generated by an execution of the protocol Π_E relative to oracle f with random tapes $R_{\mathsf{A}}, R_{\mathsf{B}}$.*

The above lemma implies the following corollary:

Corollary 3.3.2 *Let $p = 1/\operatorname{poly}(n)$ and let $\mathcal{Q}$ be some two-party cryptographic task such that for every implementation Π_{plain} in the plain model with $r = o(n/\log n)$ rounds, there is a computationally-unbounded, semi-honest, fail-stop adversary which breaks the security of Π_{plain} with probability p. Let Π be a black-box construction of $\mathcal{Q}$ with r rounds based on exponentially-hard one-way functions with security parameter n (i.e. the input/output length of f). Then $r = \Omega(n/\log n)$.*

In other words, Corollary 3.3.2 implies that unless there exists an implementation of $\mathcal{Q}$ in the plain model that ensures security and *fairness* for semi-honest parties, there is no black-box construction of $\mathcal{Q}$ from one-way functions (with input/outpus length n) that with $o(n/\log n)$ rounds.

The corollary follows from Lemma 3.3.1 due to the following: Assume such a construction Π exists with $r = o(n/\log n)$ rounds. Now consider Π_T, the λ-threshold simulation of Π. Since Π_T also has $r = o(n/\log n)$ rounds and does not make calls to the oracle, we have by hypothesis that there is an unbounded attacker $\widehat{\mathsf{A}}$ (resp. $\widehat{\mathsf{B}}$) which breaks the security of Π_T with probability $p = 1/\operatorname{poly}(n)$. Now, for $\lambda \leq p/2 = 1/\operatorname{poly}(n)$, we have that the views of Alice and Bob (as a jointly distributed random variable) in Π_T and in the λ-extended exection, Π_E, are λ-close. Moreover, given the transcript generated by Π, Alice (resp. Bob) can make at most $2^{o(n)}$ queries and produce the corresponding transcript of Π_E. Thus, there is a threshold attacker TA which plays the part of Alice (resp. Bob) in Π, makes at most $2^{o(n)}$ queries to compute the messages of Π_E, runs $\widehat{\mathsf{A}}$ (resp. $\widehat{\mathsf{B}}$) internally while simulating the view of $\widehat{\mathsf{A}}$ (resp. $\widehat{\mathsf{B}}$) using the λ-close view produced by Π_E and finally outputs whatever $\widehat{\mathsf{A}}$ (resp. $\widehat{\mathsf{B}}$) outputs. So TA breaks the security of Π_E (and thus of Π) with probability $p/2$, where the probability is computed over the randomness of f. Having the threshold attacker TA the proof can be concluded as follows:

(a) Since the attacker TA breaks security with probability $p/2 = 1/\operatorname{poly}(n)$, by an averaging argument, for at least $p/4$ fraction of the functions $f\colon \{0,1\}^n \mapsto \{0,1\}^n$, the attacker TA^f breaks security with probability $p/4$. We call such function f, a good function. **(b)** Using the security reduction S, for all good functions f, S^{f,TA^f} inverts $y = f(U_n)$ with probability at least $2^{-o(n)}$. **(c)** We can combine the algorithms S and TA to get a single oracle algorithm T^f which inverts $f(U_n)$ with probability $2^{-o(n)}$ when f is a good function by asking only $2^{o(n)}$ queries to f. Which means that in this case T asks only $2^{o(n)}$ oracle queries and inverts a *random* f with probability at least $p/4 \cdot 2^{-o(n)} = 2^{-o(n)}$ (because f is a good function with probability at least $p/4$). The latter contradicts Lemma 3.2.2.

Before we prove Lemma 3.3.1, we review relevant previous work.

The Independence Learner of [Barak and Mahmoody, 2009]. Here we describe the properties of the attacker of Barak and Mahmoody [Barak and Mahmoody, 2009] presented in the context of breaking any key agreement protocol with optimal number of queries to the random oracle. Since

the main property of the learning algorithm is that conditioned on the learner's information Alice and Bob's views are almost independent, we call this attack the independence learning algorithm.

Lemma 3.3.3 (The Independence Learner of [Barak and Mahmoody, 2009]) *Let Σ be any two-party protocol in the random oracle model (with arbitrary number of rounds) between Alice and Bob in which Alice and Bob ask at most m queries from the random oracle H. Then there is a universal constant c and a (computationally unbounded)* independence learning *algorithm which is given a parameter ϵ (called the* threshold*) as input and has the following properties. For brevity we denote the independence learning algorithm by Eve.*

- *Eve only has access the public messages sent between Alice and Bob and can ask queries from the random oracle.*
- (cm/ϵ)**-Efficiency:** *Eve is deterministic and, over the randomness of the oracle and Alice and Bob's private randomness, the expected number of Eve queries from the oracle H is at most cm/ϵ.*
- *Eve asks its queries* along *the game. Namely, although Eve can wait till the end and then ask all of her queries, her description defines which queries to be asked right after each message is sent across the public channel. So the learning algorithm is divided into the same number of rounds as the protocol.*
- $(c\sqrt{m\epsilon})$**-Security:** *Let $W = [w_1, \ldots, w_i]$ be the sequence of messages sent between Alice and Bob so far, and let I be the list of oracle query/answer pairs that Eve has asked till the* end *of the i'th round, and let $\mathbf{AB} = (\mathbf{A}, \mathbf{B})$ be the* joint *distribution over the views of Alice and Bob* only *conditioned on (W, I). By $\mathbf{A}$ and $\mathbf{B}$ we refer to the projections of $\mathbf{AB}$ over its first or second components (referring to the view of either Alice or Bob only) as random variables. For a specific view $\mathsf{A} \leftarrow \mathbf{A}$ for Alice, by $Q(\mathsf{A})$ we refer to the set of oracle queries that A contains. We also use the notation $Q(I)$ to refer to the queries denoted in I.*

  With probability at least $1 - c\sqrt{m\epsilon}$ over the randomness of Alice, Bob, and the random oracle H the following holds at all *moments during the protocol when Eve is done with her learning phase in that round: There are* independent *distributions $\widehat{\mathbf{A}}, \widehat{\mathbf{B}}$ such that:*

1. *The statistical distance between* $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ *and* $\mathbf{AB}$ *is at most* $\Delta(\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}, \mathbf{AB}) \leq c\sqrt{m\epsilon}$.
2. *For every oracle query* $q \notin Q(I)$*, it holds that* $\Pr[q \in Q(\widehat{\mathbf{A}}) \cup Q(\widehat{\mathbf{B}})] \leq \epsilon$.

- **Robustness.** *The learning algorithm is robust to the input parameter* ϵ *in the following sense. If the parameter* ϵ *changes in the interval* $\epsilon \in [\epsilon_1, \epsilon_2]$ *arbitrarily during the learner's execution (even inside a learning phase of a specific round), it still preserves* $O(cm/\epsilon_1)$*-efficiency and* $(c\sqrt{m\epsilon_2})$*-security.*

Lemma 3.3.3 is implicit in [Barak and Mahmoody, 2009], and we show how to derive it from the explicit results of [Barak and Mahmoody, 2009] in Appendix A.2.

Given a protocol Π, we now describe the λ-extended execution, Π_E, and the λ-threshold simulation, Π_T, of Π that were mentioned in Lemma 3.3.1.

Definition 3.3.4 (Extended Execution) *Let* Π *be a two-party protocol between Alice and Bob in the random oracle model where they ask at most* m *oracle queries and interact for* r *rounds. The extended execution* Π_E *of* Π *gets as input a parameter* λ *and simulates the original protocol* Π *in the random oracle model as follows.*

- *Let* $\epsilon_r = \frac{1}{m} \cdot \left(\frac{\lambda}{9rc}\right)^2$ *and for* $j \in \{r, r-1, \ldots, 2\}$ *define* $\varepsilon_{j-1} = \varepsilon_j \cdot \frac{\lambda^2}{90r^2m}$*. Note that if* r, λ, m *are* $\leq \mathrm{poly}(n)$*, then* $\epsilon_r = 1/\mathrm{poly}(n)$ *and* $\epsilon_1 = \mathrm{poly}(n)^{-r}$.
- *Now imagine an Eve who runs the independence learner of Lemma 3.3.3 and uses* ϵ_i *as its learning parameter in the learning phase after the* i*'th round.*
- *In round* i*, the party who is sending the message* w_i*, also runs the* i*'ih round of the learning phase of Eve and attaches to* w_i *the list of all the query/answer pairs that are the result of this learning algorithm. Note that since Eve's algorithm is only depending on the messages being sent and her previous knowledge about the oracle, the parties are able to do this job.*

Definition 3.3.5 (Threshold Simulation) *Let* Π *be a two-party protocol between Alice and Bob in the random oracle model where they ask at most* m *oracle queries and interact for* r *rounds. A threshold simulation* Π_T *of* Π *gets as input a parameter* λ *and simulates the original protocol* Π plainly *as follows.*

- *The parameters ϵ_i for $i \in [r]$ are defined similar to the extended execution.*
- *In the i'th round the party who sends the i'th message tries to simulate the i'th round of the extended execution but* without *using a random oracle. The way the simulation is done is as follows: To compute the message w_i, suppose q is a query to be asked from the oracle. Now if q is in the set of queries learned by Eve so far or if q was asked previously by the same party, the same answer will be returned which was used before. But, if the query q is new, a fresh random answer will be used. The same is also done to answer any query that the learning algorithm Eve tries to learn.*

The following lemma explains why a threshold simulation is indeed a good simulation of the extended execution.

Lemma 3.3.6 (Properties of the Threshold Simulation) *Let Π be a two-party protocol between Alice and Bob in the random oracle model where they ask at most m oracle queries and let Π_T and Π_E be in order its λ-threshold simulation and λ-extended execution. Then the views of Alice and Bob (as a jointly distributed random variable) in Π_T and Π_E are λ-close.*

Proof: It is easy to see that the extended execution and the threshold simulation will be exactly the same games until the following happens: A party, say Alice sends a message w_i along with the simulation of Eve's i'th round, but one of these queries (which are asked in this round either for her own protocol or to simulate Eve) will hit one of Bob's "private" queries which are *not* announced through Eve's previous simulated query/answers. We show that this "bad" event happens with probability at most λ.

Note that by the robustness of the independence learner Eve and by the choice of the (largest) parameter $\epsilon_r = \frac{1}{m} \cdot \left(\frac{\lambda}{9rc}\right)^2$, Eve's algorithm remains at least $c\sqrt{m\epsilon} = \lambda/(9r)$ secure in round i. So, except with probability at most $r \cdot \lambda/(9r) = \lambda/9$ we can pretend (as a mental experiment) that at all moments the security requirement of the learning algorithm holds with probability 1 rather than $1 - c\sqrt{m\epsilon}$. In the following we show that (up to the bad event mentioned above which happens with probability at most $\lambda/9$) the probability that an "inconsistency" happens in round i is at most $\lambda/(3r)$, and thus we will be done by a union bound. By inconsistency we mean that Alice announces (a different) answer for an oracle query that is privately asked by Bob already (or vice versa).

Suppose Alice is sending the message in the i'th round and suppose no inconsistency has happened so far. Let fix $W = [w_1, \ldots, w_{i-1}]$ to be the sequence of the messages sent till this moment and let I be the union Eve's simulated queries till the end of the $(i-1)$'th round. An inconsistency in round i can happen as follows: one of the queries asked by Alice (either to run her own protocol or to simulate Eve) hits one of Bob's private queries. We bound this probability conditioned on any fixed (W, I) over which the security property of the learner holds (as we said this property will hold with probability at least $1 - \lambda/9$).

As a mental experiment we can continue the game (after fixing (W, I)) by sampling from the random variable $(\mathsf{A}, \mathsf{B}) \leftarrow \mathbf{AB}$ for the views of Alice and Bob so far conditioned on (W, I) and then continue Alice's simulation. Let assume for a moment that we sample $(\mathsf{A}, \mathsf{B}) \leftarrow \widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ rather than from $\mathbf{AB}$. We bound the probability of any inconsistency in the former case to be $2\lambda/(9r)$, and since the distributions $\mathbf{AB}$ and $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ are $\lambda/(9r)$ close, it follows that the probability of any inconsistency in this round is bounded by $2 \cdot \lambda/(9r) + \lambda/(9r) = \lambda/(3r)$ which is small enough for us.

But now we use the security property of the independence learner. Note that when we get the sample $(\mathsf{A}, \mathsf{B}) \leftarrow \widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$, A and B are sampled *independently*. So, we can sample A first, continue Alice's computation, and then sample $\mathsf{B} \leftarrow \widehat{\mathbf{B}}$ at the end (and we will abort if the private queries collide). The number of queries that Alice will ask to run her own protocol is at most m. By the efficiency property of the learning algorithm applied to round i, the number of Eve's simulated queries in this round are, on average, at most cm/ε_i. By a Markov bound, this number is at most $\frac{cm}{\varepsilon_i} \cdot \frac{9r}{\lambda}$ with probability at least $1 - \lambda/(9r)$. So except with probability $\lambda/(9r)$ the total number of queries asked by Alice in this round is at most $m + 9cmr/(\epsilon_j \lambda) < 10cmr/(\epsilon_j \lambda)$. Note that the probability that any of these $10cmr/(\epsilon_j \lambda)$ queries are among the private queries of a sample from $\widehat{\mathbf{B}}$ (sampled as Bob's view) is at most ϵ_{j-1}. So, by a union bound, the probability that at least one of these queries hits $\widehat{\mathbf{B}}$'s private queries is at most $\frac{10cmr}{\epsilon_j \lambda} \cdot \varepsilon_{j-1} = \lambda/(9r)$ and this finishes the proof. ■

So, all left to do is to count how many queries are asked by our λ-extended execution Π_E and show that it is (say on average) at most $2^{o(n)}$. This is indeed the case because of the robustness and the efficiency properties of the learning algorithm. The smallest threshold used in our attack is $\epsilon_1 = \mathrm{poly}(n)^{-r}$ because $\lambda = 1/r$ and $r = \mathrm{poly}(n), m = \mathrm{poly}(n)$. Therefore our attacker asks

at most $O(m/\epsilon_1)$ number of queries on average which for $r = o(n/\log n)$ is at most $O(m/\epsilon_1) = \text{poly}(n)^r = 2^{o(n)}$.

3.4 Proof of the Main Theorem

In this section we first prove our main theorem for the case of exponentially-hard one-way function as the primitive used. Extending the proof to stronger primitives implied by a random oracle is discussed at the end.

Theorem 3.4.1 (Main Theorem, Formal) *Let Π be a black-box construction for two-party coin tossing (between Alice and Bob) with bias at most $o(1/\sqrt{r})$ (where r is the number of rounds in Π) based on exponentially-hard one-way functions with security parameter n (i.e., the input/output length of f). Then $r = \Omega(n/\log n)$.*

Proof: For sake of contradiction let assume that such construction exists with $r = o(n/\log n)$ round complexity. The proof goes through the following steps. We first feed Alice and Bob's protocols in the construction Π with a *random* function $f\colon \{0,1\}^n \mapsto \{0,1\}^n$. We show that in that setting at least one of the parties can ask $n^{O(r)}$ queries to f and bias the output by at least $\Omega(1/\sqrt{r})$ by a fail-stop attack. The probability over which the bias is computed also includes the randomness of f. As in Section 3.3, we call this attacker the threshold attacker, TA. Having the threshold attacker TA the proof can be concluded as follows.

(a) Since the attacker TA achieves bias $\delta = \Omega(1/\sqrt{r})$ and since the bias is always $\delta < 1$, therefore by an averaging argument, for at least $\delta/2$ fraction of the functions $f\colon \{0,1\}^n \mapsto \{0,1\}^n$, the attacker TA^f achieves bias at least $\delta/2 = \Omega(1/\sqrt{r})$. We call such function f, a good function. **(b)** Using the security reduction S, for all good functions f, S^{f,TA^f} inverts $y = f(U_n)$ with probability at least $2^{-o(n)}$. **(c)** We can combine the algorithms S and TA to get a single oracle algorithm T^f which inverts $f(U_n)$ with probability $2^{-o(n)}$ when f is a good function by asking only $2^{o(n)}\,\text{poly}(n)^r$ queries to f. For $r = o(n/\log n)$, it holds that $\text{poly}(n)^r = 2^{o(n)}$, which means that in this case T asks only $2^{o(n)} \cdot 2^{o(n)} = 2^{o(n)}$ oracle queries and inverts a *random* f with probability at least $\frac{\delta}{2} \cdot 2^{-o(n)} = 2^{-o(n)}$ (because f is a good function with probability at least $\delta/2$). The latter contradicts Lemma 3.2.2. ■

In the following we first describe the results that we borrow or derive from previous work needed for our threshold attacker TA, and then will describe and prove the properties of TA.

The Fail Stop Attacker of [Cleve and Impagliazzo, 1993]. Cleve and Impagliazzo [Cleve and Impagliazzo, 1993] showed that when computationally unbounded parties participate in any coin tossing protocol, at least one of them can bias the output bit by following the protocol honestly and aborting at some point based on the information provided to them by their view.

Lemma 3.4.2 (The Attacker of [Cleve and Impagliazzo, 1993]) *Let Σ be any two-party protocol for coin tossing between Alice and Bob with r rounds of interaction. Then either Alice or Bob can bias the output bit by $\Omega(1/\sqrt{r})$ in the fail-stop model through a computationally* unbounded *attack.*

3.4.1 Our Threshold Attacker

In this section we use the attack of Lemma 3.4.2 as well as the results of Section 3.3 to finish the proof of Theorem 3.4.1 by presenting our threshold attacker. We will do so first in a special case where the protocol Π is of a special form which we call *instant*. The case of instant constructions carries the main ideas of the proof. Later we prove Theorem 3.4.1 for constructions which are not necessarily instant.

Definition 3.4.3 (Instant Constructions) *A black-box construction of coin tossing is an* instant *construction if whenever a party aborts the protocol, the other party decides on the output bit* without *asking any additional queries to its oracle.*

We note that the protocol of Cleve [Cleve, 1986] which achieves bias at most $O(1/\sqrt{r})$ based on one-way function is in fact an instant construction.

Given an instant coin-tossing protocol Π, we apply Lemma 3.3.1 to obtain the λ-threshold simulation and λ-extended execution of Π, Π_T, Π_E. Since the threshold simulation, Π_T, is a plain protocol we can apply Lemma 3.4.2 to get an attack of bias $\Omega(1/\sqrt{r})$ by either Alice or Bob. Now if we take the simulation parameter λ to be at most $1/r = o(1/\sqrt{r})$, then the same exact attack will also give a bias of $\Omega(1/\sqrt{r}) - o(1/\sqrt{r}) = \Omega(1/\sqrt{r})$ in the extended execution. Here we crucially rely on the instant property because of the following: As soon as Alice or Bob (who is the attacker)

stops continuing the game, the other party in the threshold simulation will decide on the final output bit by looking at their current view. But this last step will not be statistically close between the extended execution and the threshold execution if in the extended execution the deciding party chooses the output after asking more queries. In other words, if the party who announces the output bit (not the attacker) wants to ask more oracle queries to compute the output bit, there should be some simulated random answers chosen by the corresponding party in the threshold simulation to on behalf of these queries, but that step is not taken care of by Lemma 3.3.6 (because the aborted party is *not* given the learning algorithm's queries for the aborted round). By Lemma 3.3.1, our attacker asks at most $2^{o(n)}$ queries.

Before going over how to handle the non-instant constructions we clarify that extending Theorem 3.4.1 to stronger primitives such as exponentially-hard collision resistant hash function is immediate. All one has to do is to substitute the collision resistant hash functions $h\colon \{0,1\}^n \mapsto \{0,1\}^{n/2}$ used in the construction by a random function $f\colon \{0,1\}^n \mapsto \{0,1\}^{n/2}$ (which is in fact a $2^{\Omega(n)}$-secure hash function). To provide access to a family of hash functions one can use the random oracle over larger domains of input/output length $3n$ and use the first n bits of the input as the index to the hash family and simply throw away the last $\frac{5n}{2}$ bits of the output. The rest of the proof remains the same.

3.4.2 Handling Non-instant Constructions

It is instructing to recall that given a random oracle there is indeed a one-round protocol which is optimally-fair: Alice asks $H(0)$ (assuming that the random oracle is Boolean) and then sends $H(0)$ to Bob which is the final output bit. If Alice aborts and does not send $H(0)$, Bob will go ahead and ask $H(0)$ himself and takes that as the final output bid. It is clear that this trivial protocol is completely fair because $H(0)$ is an unbiased bit. Also note that the proof of the previous section handing the instant constructions works just as well for protocols which use a truly random oracle (rather than a one-way function) as their primitive used. So it should be of no surprise that the proof of the instant case does not immediately generalize to cover all the black-box constructions (the trivial coin-tossing protocol based on random oracle is clearly a non-instant protocol). To handle the non-instant constructions we inherently need to use the fact that the constructions we deal with are optimally-fair protocols given any *one-way* function as the primitive used. In the following we

show how this stronger requirement of the construction gives us what we need in Theorem 3.4.1.

Making constructions almost instant. It is easy to see that any construction for coin tossing can be changed into an equivalent protocol which is "almost" an instant one. Namely, whenever a party A is sending a message m, it can also consider the possibility that the other party B will abort the game right after A sends his message. So, during the computation of m, A can go ahead and ask whatever query from the oracle which is needed to compute the final bit in case B aborts. This way, A will not need to ask any oracle queries in case B aborts in this round. By doing this change (which clearly does not affect the security of the protocol) the construction becomes "almost" instant. The point is that the receiver of the *first* message can not follow the change suggested here because they do not send any message before the first round. Therefore, in the following we only consider constructions which are "almost-instant" (i.e., the only moment that a party might violate the instant property is when the sender of the first message aborts the protocol, and the receiver might still need to ask oracle queries before deciding on the output.)

Handling almost-instant constructions. Suppose Π is an almost-instant construction. Suppose Π_E and Π_T be in order Π's extended execution and the threshold simulation games. The proof of Lemma 3.3.6 shows that if no party aborts the experiments Π_E and Π_T are λ-close. The discussion following the proof of Lemma 3.3.6 shows that if one of the parties runs the same fail-stop attack in Π_E and Π_T the experiments are *still* λ-close conditioned on the assumption that the round in which the abort happens is any round other than the first one. So, all we need to handle is the case in which the sender of the first message (which we assume to be Alice) aborts the game in the first round (after asking some oracle queries). In the following we focus on this specific cease.

Note that when aborted in the first round Bob can *not* simply simulate the extended execution by using fresh randomness to answer his oracle queries in order to decide the output bit. If he does so it might not be consistent with Alice's queries asked before aborting and thus it will not be a good simulation.[3] Despite this issues, if we are somehow magically guaranteed that when aborted in the first round, none of Bob's queries to compute the output bit collides with Alice's queries asked

[3]This will be more clear if one consider the trivial protocol mentioned above which uses a truly random oracle. If Alice aborts whenever $H(0) = 0$, and if Bob uses a fresh random answer whenever he gets aborted by Alice, then the final output will be equal to 1 with probability $3/4$ which is clearly a huge bias!

before, then we can still use fresh random answers to answer Bob's queries to compute the output bit.

Suppose after Alice computes her message but *right before* she sends this message we run the independence learning algorithm with parameter $\lambda/(10m)$. This learning algorithm will announce a set of $O(10m^2/\lambda)$ queries and answers conditioned on which any other query has a chance of at most $\lambda/(10m)$ of being asked by Alice in her computation of the first message. Let the set S be the set of all these $O(10m^2/\lambda)$ queries and let $f(S)$ be their answers. By the security property of the learning algorithm, conditioned on S and $f(S)$, an aborted Bob will not ask any query out of S which collides with Alice's private queries out of S before aborting (unless with probability at most $O(\lambda)$).

The idea is to sample the set S and $f(S)$ once for all, and hardwire them into the random oracle and Alice and Bob's algorithms. This way, simulating Bob's queries with random answers after being aborted will not lead to any inconsistency with Alice's queries unless with probability at most $O(\lambda)$. But if we *fix* the answer of such queries that might hurt the protocol's fairness. At this point we use the fact that the construction is supposed to be fair given any one-way function (and not necessarily a random function). Any random oracle is one-way with overwhelming probability even if we fix a subdomain $S \subseteq \{0,1\}^n$, $|S| \leq \mathrm{poly}(n)$ of its domain and this idea is formalized in Lemma 3.2.3. Namely, if we hardwire the random function over a subdomain $S \subseteq \{0,1\}^n$, $|S| \leq \mathrm{poly}(n)$ we can still use the same exact proof as the case of instant constructions for Theorem 3.4.1 with the only difference that now we will use Lemma 3.2.3 rather than Lemma 3.2.2.

Part II

Universal Composability and Adaptive Security

Secure multi-party computation (MPC) allows several mutually distrustful parties to perform a joint computation without compromising, to the greatest extent possible, the privacy of their inputs or the correctness of the outputs. In the absence of an honest majority, secure MPC protocols can only be realized under computational assumptions. The first MPC protocol achieving security without an honest majority was presented by [Goldreich *et al.*, 1988] and proceeds in the following two-step paradigm: the first step in the construction is to obtain protocols that are secure against semi-honest adversaries, and the second step handles malicious behavior by having the parties prove in zero knowledge that they are adhering to the protocol constructions. This second step requires the code of the underlying primitive with the use of general NP reductions to prove statements in zero knowledge.

In their seminal work, Ishai et al. [Ishai *et al.*, 2006] exhibited MPC protocols that are secure against a static adversary corrupting any number of parties and that rely only on black-box access to a low-level primitive, such as (enhanced) trapdoor permutations or homomorphic encryption schemes. This, along with the follow-up work of Haitner [Haitner, 2008], resolves the theoretical question of the minimal assumptions under which we may obtain black-box constructions of secure MPC protocols against a static adversary. The main technical contribution in both works is to construct a secure protocol for a specific two-party functionality, that of oblivious transfer (OT). The general result then follows from a classic result of Kilian's [Kilian, 1988] showing that any multi-party functionality can be securely computed using black-box access to a secure OT protocol.

A natural and challenging problem in the study of secure multi-party computation is to deal with *adaptive* adversaries – namely adversaries that may choose which parties to corrupt during the course of the computation, without having to trust that honest parties can securely erase any state information. The main tool we have for designing protocols that are secure against adaptive adversaries is that of *non-committing encryption (NCE)*. Non-committing encryption schemes were introduced by Canetti, Feige, Goldreich and Naor [Canetti *et al.*, 1996], who used them to simulate secure channels against an adaptive adversary, thus (by combining with the information theoretically secure protocol of [Ben-Or *et al.*, 1988]) obtaining a protocol secure against an adaptive adversary corrupting up to one third of the parties.[4] Non-committing encryption has also proved a crucial tool

[4]More recent work [Damgård and Ishai, 2005] uses standard pseudorandom generators to achieve security against an adaptive adversary corrupting a minority of the parties.

in the recent constructions of adaptively secure protocols tolerating a dishonest majority [Canetti *et al.*, 2002; Katz and Ostrovsky, 2004].

In our work, we revisit the problem of constructing non-committing encryption schemes, and present a construction of NCE, from a new primitive called *trapdoor simulatable PKE*, which simultaneously improves upon all of the previous constructions in [Canetti *et al.*, 1996; Damgård and Nielsen, 2000]. More specifically, our scheme enjoys optimal round complexity, relies on weaker assumptions, and has better efficiency.

Informally, a simulatable PKE is an encryption scheme with special algorithms for obliviously sampling public keys and random ciphertexts without learning the corresponding secret keys and plaintexts; in addition, both of these oblivious sampling algorithms should be efficiently invertible. We define a weaker assumption, which we refer to as trapdoor simulatable cryptosystems, and prove that it is sufficient for our construction and analysis to go through. Next, we show how to realize tradoor simulatable cryptosystems from from a variant of Rabin's trapdoor permutations (c.f. [Halpern, 1995; Schnorr, 1996; Fischlin and Fischlin, 2002]) based on the hardness of factoring Blum integers. This yields the first factoring-based non-committing encryption scheme.

NCE is used to construct OT secure against semi-honest, adaptive adversaries which, in turn, is used to construct protocols for general MPC secure against adaptive adversaries. Thus, improving the efficiency of NCE is a valuable contribution towards improving the efficiency of adaptively-secure MPC protocols. Another, perhaps more fundamental, cause of inefficiency in MPC protocols is the use of generic zero knowledge proofs to ensure semi-honest behavior. Consequently, an important research goal is to construct adaptively-secure protocols for MPC that do not require generic zero knowledge proofs. Towards this goal, Ishai, Prabhakaran and Sahai [Ishai *et al.*, 2008] recently established an efficient analogue of Kilian's result that holds even in the Universal Composability (UC) model against a malicious, adaptive adversary. Thus, they reduce the problem of finding black-box constructions of MPC protocols in the UC model secure against a malicious, adaptive adversary to finding black-box constructions of OT protocols in the UC model secure against a malicious, adaptive adversary. While there has been fairly extensive work on secure OT protocols against a static malicious adversary (e.g. [Naor and Pinkas, 2001; Kalai, 2005; Peikert *et al.*, 2008]), very few - namely [Beaver, 1998; Canetti *et al.*, 2002; Katz and Ostrovsky, 2004] - provide security against an adaptive adversary; moreover, all of

those which do follow [Goldreich *et al.*, 1988] paradigm and exploit non-black-box access to the underlying primitive.

In our work, following the paradigm introduced by [Ishai *et al.*, 2006; Haitner, 2008], we present a compiler for transforming an OT protocol secure against a semi-honest, adaptive adversary into one that is secure against a malicious, adaptive adversary. Our compiler achieves security in the UC model, assuming access to an ideal commitment functionality, and improves over previous work achieving the same security guarantee in two ways: it uses black-box access to the underlying protocol and achieves a constant multiplicative overhead in the round complexity. Combining our two results with the work of [Ishai *et al.*, 2008], we obtain the first black-box construction of UC and adaptively secure MPC from trapdoor-simulatable PKE and the ideal commitment functionality.

Chapter 4

Improved Non-Committing Encryption with Applications to Adaptively Secure Protocols

4.1 Introduction

Secure multi-party computation (MPC) allows several mutually distrustful parties to perform a joint computation without compromising, to the greatest extent possible, the privacy of their inputs or the correctness of the outputs. An important criterion in evaluating the security guarantee is *how many* parties an adversary is allowed to corrupt and *when* the adversary determines which parties to corrupt. Ideally, we want to achieve the strongest notion of security, namely, against an adversary that corrupts an arbitrary number of parties, and *adaptively* determines who and when to corrupt during the course of the computation (and without assuming erasures). Even though the latter is a very natural and realistic assumption about the adversary, most of the MPC literature only addresses security against a static adversary, namely one that chooses (and fixes) which parties to corrupt before the protocol starts executing. And if indeed such protocols do exist, it is important to answer the following question:

What are the cryptographic assumptions under which we can realize MPC

> protocols secure against a malicious, adaptive adversary that may corrupt a majority of the parties?

Towards answering this question, we revisit the problem of constructing *non-committing encryption schemes*, a cryptographic primitive first introduced by Canetti et al. [Canetti *et al.*, 1996] as a tool for building adaptively secure MPC protocols in the presence of an honest majority. Informally, non-committing encryption schemes are semantically secure, possibly interactive encryption schemes, with the additional property that a simulator can generate special ciphertexts that can be opened to both a 0 and a 1. In a more recent work, Canetti et al. [Canetti *et al.*, 2002] (extending [Beaver, 1998]) showed how to construct adaptively secure oblivious transfer protocols starting from non-committing public-key encryption schemes (i.e. the key generation algorithm must be non-interactive), which may in turn be used to construct MPC protocols secure against a malicious, adaptive adversary that may corrupt an arbitrary number of parties.

Unfortunately, the only known constructions of non-committing public-key encryption schemes (PKEs) are based on the CDH and RSA assumptions [Canetti *et al.*, 1996] and the construction exploits in a very essential way that these assumptions give rise to families of trapdoor permutations with a common domain. If we allow for an interactive key generation phase, Damgård and Nielsen [Damgård and Nielsen, 2000], building on [Beaver, 1997; Canetti *et al.*, 1996], constructed 3-round non-committing encryption schemes based on a more general assumption, that of *simulatable PKEs*, which may in turn be realized from DDH, CDH, RSA and more recently, worst-case lattice assumptions [Gentry *et al.*, 2008] (see figure 4.1).

4.1.1 Our results

First, we present a new construction of non-committing encryption schemes, which simultaneously improves upon all of the previous constructions in [Canetti *et al.*, 1996; Damgård and Nielsen, 2000]:

Optimal Round Complexity. We provide a construction of non-committing PKEs from simulatable cryptosystems. Our construction is surprisingly simple - a twist to the standard cut-and-choose techniques used in [Damgård and Nielsen, 2000; Katz and Ostrovsky, 2004] - and also admits a fairly straight-forward simulation and analysis. In particular, our construction

and the analysis are conceptually and technically simpler than those in [Canetti *et al.*, 1996; Damgård and Nielsen, 2000]; we avoid having to analyze the number of one's in certain Binomial distributions as in [Canetti *et al.*, 1996] and to consider a subtle failure mode as in [Damgård and Nielsen, 2000].

Reducing the assumptions. Informally, a simulatable PKE is an encryption scheme with special algorithms for obliviously sampling public keys and random ciphertexts without learning the corresponding secret keys and plaintexts; in addition, both of these oblivious sampling algorithms should be efficiently invertible.

We define a weaker assumption, which we refer to as trapdoor simulatable cryptosystems, and prove that it is sufficient for our construction and analysis to go through. Roughly speaking, we provide the inverting algorithms in a simulatable cryptosystem with additional trapdoor information (hence the modifier "trapdoor"), which makes it *easier* to design a simulatable cryptosystem.

Improved efficiency. While the main focus of this work is feasibility results (notably, reducing the computational assumptions for both non-committing encryption schemes and adaptively secure MPC), we show how to combine a variant of our basic construction with the use of error-correcting codes to achieve better efficiency. That is, the amortized complexity of encrypting a single bit is $O(1)$ public-key operations on a constant-sized plaintext in the underlying cryptosystem.

Thus, we obtain the following.

Theorem 4.1.1 (informal) *There exists a black-box construction of a non-committing public-key encryption scheme, starting from any trapdoor simulatable cryptosystem.*

Factoring-based constructions.

Next, we derive trapdoor simulatable cryptosystems from a variant of Rabin's trapdoor permutations (c.f. [Halpern, 1995; Schnorr, 1996; Fischlin and Fischlin, 2002]) based on the hardness of factoring Blum integers.

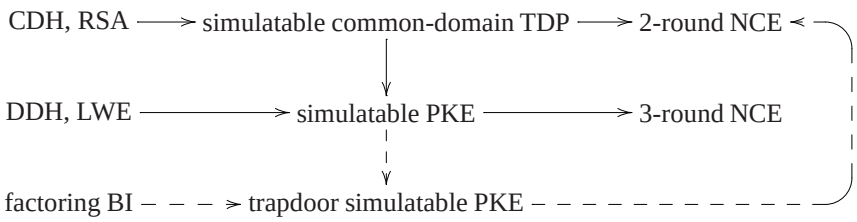


Figure 4.1: Summary of previous results (solid lines) along with our contributions (dashed lines).

Theorem 4.1.2 (informal) *Suppose factoring Blum integers is hard on average. Then, there exists a trapdoor simulatable cryptosystem.*

We stress that we do not know how to construct a simulatable cryptosystem under the same assumptions; specifically, inverting the sampling algorithm for ciphertexts in our construction without the trapdoor (the factorization of the Blum integer modulus) appears to be as hard as factoring Blum integers. This shows that trapdoor simulatable cryptosystems is indeed a meaningful and useful relaxation. In the process, we also obtain the first factoring-based dense cryptosystems.[1] When combined with enhanced trapdoor permutations, this yields the first factoring-based non-interactive proofs of knowledge [De Santis and Persiano, 1992].

Oblivious transfer and MPC.

We consider the applications of our main result to the constructions of adaptively secure oblivious transfer and general MPC protocols in both the stand-alone setting and the UC setting (c.f. [Canetti *et al.*, 2002; Ishai *et al.*, 2008] and Chapter 5).

Theorem 4.1.3 (informal) *There exists a black-box construction of a* 6*-round* 1*-out-of-*ℓ *oblivious transfer protocol for strings in the* $\mathcal{F}_{\text{COM}}$*-hybrid model*[2] *in the UC setting that is secure against a malicious, adaptive adversary, starting from any trapdoor simulatable cryptosystem.*

We add that if the oblivious key generation algorithm in the trapdoor simulatable cryptosystem achieves statistical indistinguishability (which is the case for all of the afore-mentioned constructions), then we obtain an OT protocol that is secure against a computationally unbounded malicious

[1] These are PKE schemes where a random string has a inverse polynomial probability of being a valid public key.

[2] $\mathcal{F}_{\text{COM}}$ is an ideal functionality for commitment.

sender. While our OT protocol is not as efficient as that in the recent work of Garay, Wichs and Zhou [Garay *et al.*, 2009] (we incur an additional multiplicative overhead that is linear in the security parameter), our protocol along with our general framework offers several advantages:

- In addition to relying on the $\mathcal{F}_{\text{COM}}$ functionality and a simulatable PKE (to implement non-committing encryption) as in our work, the [Garay *et al.*, 2009] framework requires a so-called enhanced dual-mode cryptosystem. This is a relatively high-level CRS-based primitive from [Peikert *et al.*, 2008] augmented with two main additional properties: the first has a flavor of oblivious sampling; the second requires that the underlying CRS be a common *random* string (modulo some system parameters) and not just a common reference string. This requirement is inherent to their framework, since this CRS is generated using a coin-tossing protocol. This latter requirement is very restrictive, and the only known construction of an enhanced dual-mode cryptosystem is based on the quadratic residuocity assumption.

- Our protocol immediately handles 1-out-of-ℓ OT, whereas [Garay *et al.*, 2009] only addresses 1-out-of-2 OT, a limitation inherited from [Peikert *et al.*, 2008]. In addition, our protocol has exactly 6 rounds, whereas the [Garay *et al.*, 2009] protocol has 6 rounds in expectation.

Combined with [Canetti *et al.*, 2002; Ishai *et al.*, 2008] and the results of Chapter 5, we obtain the following corollaries:

Corollary 4.1.4 (informal) *Assuming the existence of trapdoor simulatable cryptosystems, there exists adaptively secure multi-party protocols in the stand-alone setting and in the $\mathcal{F}_{\text{COM}}$-hybrid model in the UC setting against a malicious adversary that may adaptively corrupt any number of parties.*

Specifically, we obtain the first adaptively secure multi-party protocols based on hardness of factoring in both the stand-alone setting and the UC setting with a common reference string.

4.1.2 Additional related work

The problem of constructing encryption schemes that are secure against adaptive corruptions was first addressed in the work of Beaver and Haber [Beaver and Haber, 1992]. They considered a

simpler scenario where the honest parties have the ability to securely and completely erase previous states. For instance, an honest sender could erase the randomness used for encryption after sending the ciphertext, so that upon being corrupted, the adversary only gets to see the corresponding plaintext. An intermediate model, wherein we assume secure erasures for either the sender or receiver but not both (or, by limiting the adversary to corrupting at most one of the two parties), has been considered in several other works [Jarecki and Lysyanskaya, 2000; Canetti *et al.*, 2005; Katz and Ostrovsky, 2004].

4.1.3 Follow-up work

Recently, O'Neill et al. [O'Neill *et al.*, 2011] built upon the techniques presented in Section 4.5 of this work to construct a variant of non-committing encryption called *bi-deniable encryption in the multi-distributional model*. In addition to being non-committing, deniable encryption schemes are useful since they enable parties to resist coercion.

Organization.

We present an overview of our constructions in Section 4.2, preliminaries in Section 4.3, the formulation of a trapdoor simulatable PKE in Section 4.4, our factoring-based trapdoor simulatable PKE in Section 4.6, and our non-committing encryption scheme in Section 4.5. In Section 4.7, we show the construction of a 6-round oblivious transfer protocol.

4.2 Overview of our constructions

At a high level, our non-committing PKE is similar to that from previous works [Canetti *et al.*, 1996; Damgård and Nielsen, 2000; Katz and Ostrovsky, 2004]. The receiver generates a collection of public keys in such a way that it only knows an α fraction of the corresponding secret keys; this can be achieved by generating an α fraction of the public keys using the key generation algorithm and the remaining $1-\alpha$ fraction obliviously. Similarly, the sender generates a collection of ciphertexts in such a way that it only knows an α fraction of the corresponding plaintexts. Previous constructions all work with the natural choice of $\alpha = 1/2$ so that the simulator generates a collection of ciphertexts

half of which are encryptions of 0 and the other half are encryptions of 1. As noted in [Katz and Ostrovsky, 2004], this is sufficient for obtaining non-committing PKEs wherein at most one party is corrupted. Roughly speaking, the difficulty in handling simultaneous corruptions of both the sender and the receiver with $\alpha = 1/2$ is that in the simulation, the sender's choice of the α fraction of keys completely determine the receiver's choice of the α fraction of ciphertexts whereas in an actual honest encryption, these choices are completely independent (we elaborate on this in Section 4.2). The key insight in our construction is to work with a smaller value of α (turns out $1/4$ is good enough).

A toy construction.

Consider the following encryption scheme, which is a simplification of that in [Katz and Ostrovsky, 2004; Damgård and Nielsen, 2000]. The receiver generates a pair of public keys $(\mathrm{PK}_0, \mathrm{PK}_1)$ by generating one key (selected at random) using the key-generation algorithm, and the other using the oblivious sampling algorithm. To encrypt a bit b, the sender generates a pair of ciphertexts (C_0, C_1) as follows: pick a random bit r, set C_r to be $\mathsf{Enc}_{\mathrm{PK}_r}(b)$ and choose C_{1-r} using the oblivious sampling algorithm. To decrypt, the receiver decrypts exactly one of C_0, C_1 using the secret key that it knows. This construction corresponds to $\alpha = 1/2$ where α is the fraction of public keys for which the receiver knows the secret key, and also the fraction of ciphertexts for which the sender knows the plaintext. Observe that this encryption scheme has the following properties:

- It has a constant decryption error of $1/4$ if an obliviously sampled ciphertext is equally likely to decrypt to 0 or 1. As shown in [Katz and Ostrovsky, 2004], this error can be reduced by standard repetition techniques.

- It tolerates corruption of either the sender or the receiver, but not both. Consider a simulator that generates both of $(\mathrm{PK}_0, \mathrm{PK}_1)$ (along with $\mathrm{SK}_0, \mathrm{SK}_1$) using the key-generation algorithm, and a ciphertext (C_0, C_1) as follows: pick a random bit β, and set C_0 to be $\mathsf{Enc}_{\mathrm{PK}_0}(\beta)$ and C_1 to be $\mathsf{Enc}_{\mathrm{PK}_1}(1-\beta)$. Suppose the simulator later learns that this is an encryption of 0. If only the sender is corrupted, the simulator claims $r = \beta$ and that $C_{1-\beta}$ is obliviously sampled. If only the receiver is corrupted, it claims that it knows SK_β and that $\mathrm{PK}_{1-\beta}$ is oblivious sampled.

We highlight two subtleties in the above simulation strategy. First, it achieves 0 decryption error (as opposed to $1/4$ in an honest encryption); this can be fixed with a somewhat more involved simulation strategy. This in turn becomes pretty complicated once we use standard repetition techniques to reduce the decryption error. Next, it is always the case in the simulation that either both PK_0 and C_0 are obliviously sampled, or both PK_1 and C_1 are obliviously sampled. As such, this simulation strategy fails if both the sender and the receiver are corrupted, because in an actual encryption, which of PK_0, PK_1 and which of C_0, C_1 are obliviously sampled are determined independently.

Our encryption scheme.

As noted in the introduction, the key insight in our construction is to work with a small value of α. In addition, following [Damgård and Nielsen, 2000], we use a random k-bit encoding of 0 and 1, where k is the security parameter:

- The receiver generates $4k$ public keys $\text{PK}_1, \ldots, \text{PK}_{4k}$: k of them are generated using the key-generation algorithm, and the remaining $3k$ are generated using the oblivious sampling algorithm. The receiver then sends $\text{PK}_1, \ldots, \text{PK}_{4k}$ along with two random k-bit messages M_0, M_1.

- To encrypt a bit b, the sender sends $4k$ ciphertexts (one for each of $\text{PK}_1, \ldots, \text{PK}_{4k}$), of which k are encryptions of M_b, and the remaining ones are obliviously sampled.

- To decrypt, the receiver decrypts the k ciphertexts for which it knows the corresponding secret key. If any of the k plaintexts matches M_0, it outputs 0 and otherwise, it outputs 1.

Encoding 0 and 1 randomly as M_0 and M_1 is useful for two reasons:

- That an obliviously sampled ciphertext is equally likely to decrypt to 0 or 1 is no longer needed to guarantee correctness (c.f. [Damgård and Nielsen, 2000]). Indeed, reasoning about decryptions of obliviously sampled ciphertext is non-trivial for the lattice-based simulatable PKEs in [Gentry *et al.*, 2008].

- Constructing a simulator becomes much easier as we avoid having to generate distributions over k independent biased bits conditioned on the majority of the bits being 0, say. Generating such distributions arises for instance in [Canetti *et al.*, 1996] and is related to the first subtlety associated with the naive simulation strategy. In our construction, the simulated ciphertext comprises k encryptions of M_0, k encryptions of M_1, and $2k$ obliviously generated ciphertexts. Having these extra $2k$ obliviously generated ciphertexts (which is possible because $\alpha < 1/2$) is crucial for handling simultaneous corruptions of the sender and the receiver.

Trapdoor Simulatable PKEs from factoring.

Our factoring-based trapdoor simulatable PKE construction consists of two main steps. First, we modify the Rabin trapdoor permutations based on squaring modulo Blum integer so that it remains a permutation over any arbitrary integer modulus. This relies on the following number-theoretical structural lemma implicit in [Halpern, 1995; Schnorr, 1996; Fischlin and Fischlin, 2002][3]:

> Let N be an arbitrary odd k-bit integer, and let $Q_N = \{a^{2^k} \pmod N \mid a \in Z_N^*\}$. Then, the map $\psi : x \mapsto x^2$ defines a permutation over Q_N.

We also provide an efficient algorithm for inverting ψ given the factorization of N. Note that the standard algorithm for computing square roots does not guarantee that the output lies in Q_N. Moreover, the probability that a random square root lies in Q_N may be exponential small so we cannot repeatedly computing random square roots until we find one in Q_N; it's also not clear a-priori how to test membership in Q_N even given the factorization of N.

The next step transforms the family of trapdoor permutations ψ acting on the domain Q_N into a family of "enhanced" trapdoor permutations with the same domain Q_N, using an idea from [Goldreich, 2004, Section C.1]. The latter has the property that we can obliviously sample a random element y in Q_N so that given y along with the coin tosses used to sample y, it is infeasible to compute the preimage of y under the permutation (note that the naive algorithm for sampling a

[3] It was shown in [Halpern, 1995] that ψ defines a permutation over the subgroup O_N of Z_N^* of odd order, and that O_N contains Q_N; turns out $O_N = Q_N$. While Q_N is trivially sampleable, it is not clear a-priori how to sample from O_N.

random element of Q_N gives away its preimage under ψ). We will need the oblivious sampling algorithm for a random element in Q_N in our oblivious sampling algorithm for random ciphertexts. We will also need to realize trapdoor invertibility for the latter, which requires an efficient algorithm that given the factorization of N and an element y in Q_N, outputs a random 2^k'th root of y.[4] Note that iteratively computing random square roots k times does not work: after computing the first square root, we may not end up with a 2^{k-1}'th power.

4.3 Preliminaries

If $\mathcal{A}$ is a probabilistic polynomial time (hereafter, ppt) algorithm that runs on input x, $\mathcal{A}(x)$ denotes the random variable according to the distribution of the output of $\mathcal{A}$ on input x. We denote by $\mathcal{A}(x;r)$ the output of $\mathcal{A}$ on input x and random coins r. To simplify the notation, we will often omit quantifying over the distribution for r; it will usually be clear from the context when r is not fixed, that it is drawn from the uniform distribution over strings of the appropriate length.

We assume that the reader is familiar with the standard definitions of public-key encryption schemes and semantic security (c.f. [Goldwasser and Micali, 1984; Goldreich, 2004]). We stress that we allow decryption errors that are exponentially small in k:

Definition 4.3.1 (encryption scheme) *A triple* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is an encryption scheme, if* Gen *and* Enc *are ppt algorithms and* Dec *is a deterministic polynomial-time algorithm such that for every message* $m \in \{0,1\}^*$ *of polynomial length,* $\Pr[\mathsf{Gen}(1^k) \to (\mathrm{PK}, \mathrm{SK}), \mathsf{Enc}_{\mathrm{PK}}(m) \to c; \mathsf{Dec}_{\mathrm{SK}}(c) \neq m] < 2^{-\Omega(k)}$.

4.3.0.1 Non-committing encryption.

For simplicity, we present the definition of a non-committing public-key encryption scheme for single-bit messages:

Definition 4.3.2 (non-committing encryption [Canetti *et al.*, 1996]) *A* non-committing (bit) encryption scheme *consists of a tuple* $(\mathsf{NCGen}, \mathsf{NCEnc}, \mathsf{NCDec}, \mathsf{NCSim})$ *where* $(\mathsf{NCGen}, \mathsf{NCEnc}, \mathsf{NCDec})$

[4] If we are given just N and not its factorization, this problem is at least as hard is factoring random Blum integers. This is in essence why we only obtain a factoring-based trapdoor simulatable PKE and not a simulatable PKE.

is an encryption scheme and NCSim *is the simulation algorithm that on input* 1^k*, outputs* $(e, c, \sigma_{\mathrm{G}}^0, \sigma_{\mathrm{E}}^0, \sigma_{\mathrm{G}}^1, \sigma_{\mathrm{E}}^1)$ *with the following property: for* $b = 0, 1$ *the following distributions are computationally indistinguishable:*

- *the joint view of an honest sender and an honest receiver in a normal encryption of* b*:*

$$\{(e, c, \sigma_{\mathrm{G}}, \sigma_{\mathrm{E}}) \mid (e, d) = \mathsf{NCGen}(1^k; \sigma_{\mathrm{G}}), c = \mathsf{NCEnc}_e(b; \sigma_{\mathrm{E}})\}$$

- *simulated view of an encryption of* b*:*

$$\{(e, c, \sigma_{\mathrm{G}}^b, \sigma_{\mathrm{E}}^b) \mid \mathsf{NCSim}(1^k) \rightarrow (e, c, \sigma_{\mathrm{G}}^0, \sigma_{\mathrm{E}}^0, \sigma_{\mathrm{G}}^1, \sigma_{\mathrm{E}}^1)\}$$

It follows from the definition that a non-committing encryption scheme is also semantically secure.

Encrypting longer messages. Starting with a non-committing bit encryption scheme (NCGen, NCEnc, NCDec, NCSim), we may encrypt a longer message of length n by generating n independent public keys using NCGen, encrypting each bit of the message using a different key and then concatenating the n ciphertexts. Note that this is different from the case of semantically secure encryption, where we may encrypt each bit using the same public key.

4.4 Trapdoor Simulatable Public Key Encryption

A ℓ*-bit trapdoor simulatable encryption scheme* consists of an encryption scheme (Gen, Enc, Dec) augmented with (oGen, oRndEnc, rGen, rRndEnc). Here, oGen and oRndEnc are the oblivious sampling algorithms for public keys and ciphertexts, and rGen and rRndEnc are the respective inverting algorithms[5]. We require that, for all messages $m \in \{0, 1\}^\ell$, the following distributions are

[5] Existence of such inverting algorithms is called *trapdoor invertibility*. Compared to the simulatable cryptosystem (without trapdoor) defined in [Damgård and Nielsen, 2000], rGen (resp. rRndEnc) takes r_{G} (resp. $(r_{\mathrm{G}}, r_{\mathrm{E}}, m)$) as the additional trapdoor information.

computationally indistinguishable:

$$\{\mathsf{rGen}(r_{\mathrm{G}}), \mathsf{rRndEnc}(r_{\mathrm{G}}, r_{\mathrm{E}}, m), \mathrm{PK}, c \mid (\mathrm{PK}, \mathrm{SK}) = \mathsf{Gen}(1^k; r_{\mathrm{G}}), c = \mathsf{Enc}_{\mathrm{PK}}(m; r_{\mathrm{E}})\}$$
$$\text{and } \{\hat{r}_{\mathrm{G}}, \hat{r}_{\mathrm{E}}, \hat{\mathrm{PK}}, \hat{c} \mid (\hat{\mathrm{PK}}, \perp) = \mathsf{oGen}(1^k; \hat{r}_{\mathrm{G}}), \hat{c} = \mathsf{oRndEnc}_{\hat{\mathrm{PK}}}(1^k; \hat{r}_{\mathrm{E}})\}$$

It follows from the definition that a trapdoor simulatable encryption scheme is also semantically secure.

Encrypting longer messages. We note that if we started only with a trapdoor simulatable PKE for single bits, we may encrypt a longer message of length n by generating a single public key PK using Gen, and concatenating each of the message encrypted under PK.

4.5 Non-Committing Encryption from Weaker Assumptions

Theorem 4.5.1 *Suppose there exists a trapdoor simulatable encryption scheme. Then, there exists a non-committing encryption scheme as well as a universally composable oblivious transfer protocol secure against semi-honest, adaptive adversaries.*

We show how to construct a non-committing bit encryption scheme (NCGen, NCEnc, NCDec, NCSim) from a k-bit trapdoor simulatable PKE (Gen, Enc, Dec) (augmented with (oGen, oRndEnc, rGen, rRndEnc)). This is sufficient to establish the theorem by the connection between encrypting single bits and multiple bits as discussed in Sections 4.3 and 4.4. Our construction is presented in Figures 4.2 and 4.3.

4.5.1 Correctness.

We begin by establishing correctness.

- Assume that the input $[c_1, \ldots, c_{4k}]$ to the decryption algorithm is a random encryption of 0. Recall that $J = \{\mathsf{Dec}_{\mathrm{SK}_i}(c_i) \mid i \in T\}$ and we will output 0 unless $M_0 \notin J$. It is easy to see that $\Pr[M_0 \notin J] \leq \binom{3k}{k} / \binom{4k}{k} + 2^{-\Omega(k)}$ where the first summand comes from the probability

Key Generation $\mathsf{NCGen}(1^k)$:

1. Pick M_0, M_1 at random from $\{0,1\}^k$.
2. Choose a random subset $T \subseteq [4k]$ of size k.
3. For $i = 1, 2, \ldots, 4k$, generate a pair $(\mathrm{PK}_i, \mathrm{SK}_i)$ as follows:

$$(\mathrm{PK}_i, \mathrm{SK}_i) = \begin{cases} \mathsf{Gen}(1^k) & \text{if } i \in T \\ \mathsf{oGen}(1^k) & \text{otherwise} \end{cases}$$

Set $e = [M_0, M_1, \mathrm{PK}_1, \ldots, \mathrm{PK}_{4k}]$ and $d = [T, \mathrm{SK}_1, \ldots, \mathrm{SK}_{4k}]$.

Encryption $\mathsf{NCEnc}_{\mathrm{PK}}(b)$:

1. Choose a random subset $S \subseteq [4k]$ of size k.
2. For $i = 1, 2, \ldots, 4k$, generate a ciphertext c_i as follows:

$$c_i = \begin{cases} \mathsf{Enc}_{\mathrm{PK}_i}(M_b) & \text{if } i \in S \\ \mathsf{oRndEnc}_{\mathrm{PK}_i}(1^k) & \text{otherwise} \end{cases}$$

Set $c = [c_1, \ldots, c_{4k}]$.

Decryption $\mathsf{NCDec}_{\mathrm{PK}}(c)$:

1. Compute $J = \{\mathsf{Dec}_{\mathrm{PK}_i}(c_i) | i \in T\}$.
2. If $M_0 \in J$, output 0; else, output 1.

Figure 4.2: Non-Committing Encryption Scheme $(\mathsf{NCGen}, \mathsf{NCEnc}, \mathsf{NCDec})$

that $S \cap T = \emptyset$ and the second bounds the probability of a decryption error in the underlying encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$.

- Assume that the input $[c_1, \ldots, c_{4k}]$ to the decryption algorithm is a random encryption of 1. Recall that $J = \{\mathsf{Dec}_{\mathrm{SK}_i}(c_i) \mid i \in T\}$ and we will output 1 unless $M_0 \in J$. To bound $\Pr[M_0 \in J]$, observe that the distribution of J depends only on $M_1, \mathrm{PK}_1, \ldots, \mathrm{PK}_{4n}$, $T, \mathrm{SK}_1, \ldots, \mathrm{SK}_{4n}$ and the coin tosses used to generate $c_1, \ldots, c_{4k}$, and is therefore independent of the choice of a random M_0. This means that for each $i \in T$, the probability that $\mathsf{Dec}_{\mathrm{SK}_i}(c_i)$ equals M_0 is 2^{-k}. Taking a union bound, we obtain $\Pr[M_0 \in J] \leq k \cdot 2^{-k}$.

4.5.2 Security.

We need to show that for each $b = 0, 1$, a normal encryption of b and a simulated encryption of b are computationally indistinguishable. Note that the view in a normal encryption of b contains two

Simulation NCSim:

1. Pick M_0, M_1 at random from $\{0,1\}^k$.
2. Picking the sets S_0, S_1, T_0, T_1:
   - Pick two random subsets S_0, T_0 of $[4k]$ each of size k.
   - Pick two random subsets S_1, T_1 of $[4k] \setminus (S_0 \cup T_0)$ such that $|S_1 \cap T_1| = |S_0 \cap T_0|$.
3. Generating the keys: for $i = 1, 2, \ldots, 4k$, set

$$(\mathrm{PK}_i, \mathrm{SK}_i) = \begin{cases} \mathsf{Gen}(1^k; r_{\mathrm{G}}^i) & \text{if } i \in T_0 \cup S_0 \cup T_1 \cup S_1 \\ \mathsf{oGen}(1^k; \hat{r}_{\mathrm{G}}^i) & \text{otherwise} \end{cases}$$

4. Generating the ciphertext: for $i = 1, 2, \ldots, 4k$, set

$$c_i = \begin{cases} \mathsf{Enc}_{\mathrm{PK}_i}(M_0; r_{\mathrm{E}}^i) & \text{if } i \in S_0 \\ \mathsf{Enc}_{\mathrm{PK}_i}(M_1; r_{\mathrm{E}}^i) & \text{if } i \in S_1 \\ \mathsf{oRndEnc}_{\mathrm{PK}_i}(\hat{r}_{\mathrm{E}}^i) & \text{otherwise} \end{cases}$$

5. Simulating an opening to b: set $\sigma_{\mathrm{G}}^b = \{T_b, u_{\mathrm{G}}^{b,1}, \ldots, u_{\mathrm{G}}^{b,4k}\}$ and $\sigma_{\mathrm{E}}^b = \{S_b, u_{\mathrm{E}}^{b,1}, \ldots, u_{\mathrm{E}}^{b,4k}\}$, where

$$u_{\mathrm{G}}^{b,i} = \begin{cases} r_{\mathrm{G}}^i & \text{if } i \in T_b \\ \mathsf{rGen}(r_{\mathrm{G}}^i) & \text{if } i \in T_0 \cup T_1 \cup S_0 \cup S_1 \setminus T_b \\ \hat{r}_{\mathrm{G}}^i & \text{otherwise} \end{cases}$$

$$u_{\mathrm{E}}^{b,i} = \begin{cases} r_{\mathrm{E}}^i & \text{if } i \in S_b \\ \mathsf{rRndEnc}(r_{\mathrm{G}}^i, r_{\mathrm{E}}^i, M_{1-b}) & \text{if } i \in S_{1-b} \\ \hat{r}_{\mathrm{E}}^i & \text{otherwise} \end{cases}$$

Set $e = [M_0, M_1, \mathrm{PK}_1, \ldots, \mathrm{PK}_{4k}], c = [c_1, \ldots, c_{4k}]$. Additionally output $\sigma_{\mathrm{G}}^0, \sigma_{\mathrm{E}}^0, \sigma_{\mathrm{G}}^1, \sigma_{\mathrm{E}}^1$.

Figure 4.3: Non-Committing Encryption Scheme NCSim

sets T, S which we will label as T_b, S_b and we will append to the view two sets T_{1-b}, S_{1-b} that are determined as follows: pick two random subsets S_{1-b}, T_{1-b} of $[4k] \setminus (S_b \cup T_b)$ such that $|S_1 \cap T_1| = |S_0 \cap T_0|$; call this distribution H_0. We will also append to the view in a simulated encryption of b the sets T_{1-b}, S_{1-b} as determined by the experiment NCSim; call this distribution H_{4k}. We will show that the augmented distributions H_0 and H_{4k} are computationally indistinguishable in two steps:

Reasoning about the sets. First, we claim that the 4-tuple (S_0, T_0, S_1, T_1) in the augmented distribution H_0 and in H_{4k} are identically distributed. If $b = 0$, this is obvious since the distributions are defined in exactly the same way. The case for $b = 1$ follows from a symmetry argument,

namely that if we switch (S_0, T_0) with (S_1, T_1) in the experiment NCSim, we get exactly the same distribution. Henceforth, it suffices to argue that H_0 and H_{4k} are computationally indistinguishable, conditioned on some fixed (S_0, T_0, S_1, T_1) in both H_0 and H_{4k}. We may now WLOG focus on the case $b = 0$. In fact, we may as well also fix M_0, M_1 in both H_0 and H_{4k}. In addition to $S_0, T_0, S_1, T_1, M_0, M_1$, the distributions H_0, H_{4k} comprise:

- $4k$ public keys $\text{PK}_1, \ldots, \text{PK}_{4k}$ (generated using either Gen or oGen);
- $4k$ ciphertexts $c_1, \ldots, c_{4k}$ (generated using either Enc or $\mathsf{oRndEnc}$);
- $4k$ sets of coin tosses $u_{\text{G}}^1, \ldots, u_{\text{G}}^{4k}$ for generating the public/secret keys; and
- $4k$ sets of coin tosses $u_{\text{E}}^1, \ldots, u_{\text{E}}^{4k}$ for generating the ciphertexts.

That is, we have $4k$ tuples of the form $(\text{PK}_i, c_i, u_{\text{G}}^i, u_{\text{E}}^i), i = 1, \ldots, 4k$ in each view. Since S_0, T_0, S_1, T_1 are fixed, each of these $4k$ tuples are independently sampled from some distribution that only depends on the index i. Denote by $X_1, \ldots, X_{4k}$ the random variables for the $4k$ tuples in H_0, and $Y_1, \ldots, Y_{4k}$ the random variables for the $4k$ tuples in H_{4k}.

The hybrid argument. Next, we argue that X_i and Y_i are computationally indistinguishable for $i = 1, \ldots, 4k$, from which the indistinguishability of H_0 and H_{4k} follows via a hybrid argument. There are several cases we need to consider:

- $i \in T_0$ or $i \in [4k] \setminus (T_0 \cup S_0 \cup T_1 \cup S_1)$. It is easy to verify that in either of these cases, X_i and Y_i are identically distributed.
- $i \in S_1$ ("$\mathsf{oGen}, \mathsf{oRndEnc} \cong \mathsf{Gen}, \mathsf{Enc}$"). Here, X_i is the distribution

$$\{\hat{\text{PK}}, \hat{c}, \hat{r}_{\text{G}}, \hat{r}_{\text{E}} \mid (\hat{\text{PK}}, \perp) = \mathsf{oGen}(\hat{r}_{\text{G}}), \hat{c} = \mathsf{oRndEnc}_{\hat{\text{PK}}}(\hat{r}_{\text{E}})\}$$

  and Y_i is the distribution

$$\{\text{PK}, c, \mathsf{rGen}(r_{\text{G}}), \mathsf{rRndEnc}(r_{\text{G}}, r_{\text{E}}, M_1) \mid (\text{PK}, \text{SK}) = \mathsf{Gen}(r_{\text{G}}), c = \mathsf{Enc}_{\text{PK}}(M_1; r_{\text{E}})\}.$$

  Indistinguishability follows immediately from the security of the trapdoor simulatable PKE.

- $i \in S_0 \setminus T_0$ ("$\mathsf{oGen}, \mathsf{Enc} \cong \mathsf{Gen}, \mathsf{Enc}$"). Here, X_i is the distribution

$$\{\hat{\mathrm{PK}}, c, \hat{r}_{\mathrm{G}}, r_{\mathrm{E}} \mid (\hat{\mathrm{PK}}, \perp) = \mathsf{oGen}(\hat{r}_{\mathrm{G}}), c = \mathsf{Enc}_{\hat{\mathrm{PK}}}(M_0; r_{\mathrm{E}})\}$$

  and Y_i is the distribution

$$\{\mathrm{PK}, c, \mathsf{rGen}(r_{\mathrm{G}}), r_{\mathrm{E}} \mid (\mathrm{PK}, \mathrm{SK}) = \mathsf{Gen}(r_{\mathrm{G}}), c = \mathsf{Enc}_{\mathrm{PK}}(M_0; r_{\mathrm{E}})\}.$$

  Indistinguishability follows again from the security of the trapdoor simulatable PKE.

- $i \in T_1 \setminus S_1$ ("$\mathsf{oGen}, \mathsf{oRndEnc} \cong \mathsf{Gen}, \mathsf{oRndEnc}$"). Here, X_i is the distribution

$$\{\hat{\mathrm{PK}}, \hat{c}, \hat{r}_{\mathrm{G}}, \hat{r}_{\mathrm{E}} \mid (\hat{\mathrm{PK}}, \perp) = \mathsf{oGen}(\hat{r}_{\mathrm{G}}), \hat{c} = \mathsf{oRndEnc}_{\hat{\mathrm{PK}}}(\hat{r}_{\mathrm{E}})\}$$

  and Y_i is the distribution

$$\{\mathrm{PK}, \hat{c}, \mathsf{rGen}(r_{\mathrm{G}}), \hat{r}_{\mathrm{E}} \mid (\mathrm{PK}, \mathrm{SK}) = \mathsf{Gen}(r_{\mathrm{G}}), \hat{c} = \mathsf{oRndEnc}_{\mathrm{PK}}(\hat{r}_{\mathrm{E}})\}.$$

  Indistinguishability follows again from the security of the trapdoor simulatable PKE.

4.5.3 Improving the efficiency.

Instead of using sets $S, T \subset [4k]$ of size k, we choose $S, T \subset [40]$ of size 10. The previous analysis still goes through, except we now have a constant decryption error. To address this problem, we first encode the message[6] with a linear-rate error-correcting code that corrects a constant fraction of errors, and then encrypt the codeword with the encryption scheme with constant error.

4.6 Trapdoor Simulatable PKE from Hardness of Factoring

Theorem 4.6.1 *Suppose factoring Blum integers is hard on average, and that Blum integers are dense, then there exists a trapdoor simulatable PKE.*

[6] The codeword length (or, equivalently the message length) should be $\Omega(k)$. Then, by Chernoff bound, the number of decryption errors remains a constant fraction of the codeword length with overwhelming probability.

For simplicity, we only present a 1-bit trapdoor simulatable encryption scheme; we may encrypt longer messages by encrypting bit by bit.

4.6.1 A number-theoretic lemma.

Fix any k-bit integer modulus N and we will work with the group Z_N^*. We will use $\mathsf{factor}(N)$ to denote the factorization of N, and we define $Q_N = \{a^{2^k} \mid a \in Z_N^*\}$. Now, consider the map $\psi_N : Q_N \to Q_N$ given by $\psi_N(x) = x^2 \pmod N$. As shown in [Halpern, 1995, Facts 3.5-3.7], ψ_N defines a permutation on Q_N. We provide a more direct proof which also yields an efficient algorithm to invert ψ_N given $\mathsf{factor}(N)$.

Claim 4.6.2 *The map ψ_N defines a permutation on Q_N.*

Proof: Let q denote the largest odd divisor of $\phi(N)$, where $\phi(\cdot)$ is the Euler's totient function. It is easy to see that $\phi(N)$ divides $2^k q$, since $N < 2^k$. Take any $y \in Q_N$, where $y = a^{2^k}$. Then by Euler's theorem, $y^q = 1 \pmod N$ and thus $\psi_N(y^{(q+1)/2}) = y \pmod N$. Clearly, $y^{(q+1)/2} \in Q_N$, so the map ψ_N is surjective. Moreover, the range and domain of ψ_N have equal sizes, so ψ_N must define a bijection. ■

4.6.2 The construction.

We sketch the construction here; the formal construction is shown in Figure 4.4.

STEP 1: First, we construct a family of "weakly one-way" enhanced trapdoor permutations. We start by modifying ψ_N to obtain a new family of permutations π_N; the modification is analogous to that in [Goldreich, 2004, Section C.1] to obtain enhanced trapdoor permutations from Rabin's trapdoor permutations. The permutations $\pi_N : Q_N \to Q_N$ are indexed by a k-bit integer N and is given by:

$$\pi_N(x) \stackrel{\text{def}}{=} \psi_N^{k+1}(x) = x^{2^{k+1}} \pmod N$$

and the trapdoor is $\mathsf{factor}(N)$. We may sample from this family by running Bach's algorithm [Bach, 1988; Kalai, 2002] to pick a random k-bit integer along with its factorization.

It is easy to verify π_N is a family of trapdoor permutations. Clearly, π_N is a permutation because it is the $(k+1)$-fold iterate of a permutation ψ_N. Given the index N, π_N is efficiently computable by repeated squaring. Given the trapdoor $\mathsf{factor}(N)$, π_N^{-1} is efficiently computable given $\mathsf{factor}(N)$, by simply mapping y to $y^{((q+1)/2)^{k+1}}$, i.e., raising y to the $(q+1)/2$'th power $k+1$ times. Here, q denotes the largest odd divisor of $\phi(N)$, which is easy to compute with the trapdoor. Moreover, we can show that if N is a Blum integer (which occurs with probability $\Omega(1/k^2)$ [Granville and Martin, 2004; Rubinstein and Sarnak, 1994]), then inverting π_N given N is at least as hard as factoring N. This implies that π_N is one-way with probability $\Omega(1/k^2)$ over the choice of N.

STEP 2: Construct a "weak" encryption scheme using the standard construction of PKE from trapdoor permutations via the Goldreich-Levin hard-core predicate. The public key is N, the secret key is $\mathsf{factor}(N)$, and to encrypt a bit b, we pick a random $x \in Q_N, r \in \{0,1\}^k$ and output $(\pi_N(x), r, (x \cdot r) \oplus b)$, where $x \cdot r$ is the standard dot-product of k-bit strings. Again, this scheme will be semantically secure with probability $\Omega(1/k^2)$ over the choice of N.

STEP 3: To boost the security of the "weak" encryption scheme, we define a new scheme where the public key is k^3 random k-bit strings $N_1, \ldots, N_{k^3}$ (with overwhelming probability, one of these is a Blum integer), and to encrypt a bit b, we pick random $b_1, \ldots, b_{k^3}$ such that $b = b_1 \oplus \cdots \cdot b_{k^3}$ and concatenate the encryptions of $b_1, \ldots, b_{k^3}$ under the respective public keys $N_1, \ldots, N_{k^3}$. By a standard argument (c.f. [Yao, 1982; De Santis and Persiano, 1992]), this encryption scheme is semantically secure in the standard sense.

4.6.3 Analysis.

Indeed, we claim something stronger – that the encryption scheme derived in Step 3 is a trapdoor simulatable PKE.

- (Oblivious sampling & trapdoor invertibility for key generation) This is trivial, since a random public key corresponds to a string in $\{0,1\}^{4k}$. We can clearly sample such a public key without learning the secret key.

- (Oblivious sampling & trapdoor invertibility for random ciphertext) For simplicity, we present the algorithms for sampling random ciphertext for the scheme obtained in Step 2. Here, sampling is easy: on input the public key N, pick $\gamma \in Z_N^*, s \in \{0,1\}^k, \beta \in \{0,1\}$) and output (γ^{2^k}, s, β). To implement reverse sampling, we need an efficient algorithm that given $\mathsf{factor}(N)$ and $x \in Q_N$, output a random element of the set $\{\gamma \in Z_N^* \mid \gamma^{2^k} = \pi_N(x) = x^{2^{k+1}}\}$. This can be accomplished as follows: pick a random $\eta \in Z_N^*$ and output $x^2 \cdot \eta/(\eta^{2^k})^{((q+1)/2)^k}$, where q is as before the largest odd divisor of $\phi(N)$. This works because $\eta/(\eta^{2^k})^{((q+1)/2)^k}$ will be a random 2^k'th root of $1 \pmod N$.

For the actual proof of security, we will need to show that if N is a random Blum integer, then the following distributions are computationally indistinguishable for every b:

$$\{(N, \gamma, \pi_N(x), r, (x \cdot r) \oplus b)\} \text{ and } \{(N, \gamma, \gamma^{2^k}, r, \beta)\}$$

The first distribution corresponds to an encryption of b using modulus N and randomness (x, r) along with γ the output of $\mathsf{rRndEnc}$ (a random solution to the equation $\gamma^{2^k} = \pi_N(x)$). The second corresponds to an obliviously generated ciphertext along with the randomness. If there exists an efficient distinguisher, then there exists an efficient procedure $\mathcal{A}$ that on input N, γ, outputs $\pi_N^{-1}(\gamma^{2^k})$ with noticeable probability. Since squaring is a bijection on quadratic residues modulo Blum integers, the output of $\mathcal{A}$ is also the 4th root of γ^2. We may then use a reduction in [Goldreich, 2004, Section C.1] to derive from $\mathcal{A}$ an algorithm for factoring N with noticeable probability.

4.7 Oblivious Transfer and MPC

We describe the construction underlying Theorem 4.1.3, which proceeds in two steps:

STEP 1: We begin with the [Canetti *et al.*, 2002] construction of a semi-honest OT protocol as applied to our non-committing encryption scheme, and observe that the protocol is secure against malicious senders. For that, we just need to show how to extract the sender's input when the receiver is honest. In this case, the simulator will generate the public keys sent by the receiver in the first message along with the secret keys, so that it can then extract the malicious sender's input by decrypting.

STEP 2: Next, we apply the compiler presented in Chapter 5 to "boost" the security guarantee from tolerating semi-honest receivers to tolerating malicious receivers. (Note that we will not need to apply OT reversal as in Chapter 5.)

Key generation $\mathsf{Gen}(1^k)$:

1. Run Bach's algorithm using the randomness r_{G} to sample random $N_1, \ldots, N_{k^3} \in \{0,1\}^k$ along with their factorization $\mathsf{factor}(N_1), \ldots, \mathsf{factor}(N_{k^3})$.
2. Set $\textsc{pk} = [N_1, \ldots, N_{k^3}]$ and $\textsc{sk} = [\mathsf{factor}(N_1), \ldots, \mathsf{factor}(N_{k^3})]$.

Encryption $\mathsf{Enc}(b)$:

1. Parse the randomness r_{E} as $(a_1, \ldots, a_{k^3}) \in Z^*_{N_1} \times \cdots \times Z^*_{N_{k^3}}$, $r_1, \ldots, r_{k^3} \in \{0,1\}^k$ and $b_1, \ldots, b_{k^3-1} \in \{0,1\}$.
2. Compute $b_{k^3} = b \oplus b_1 \oplus \cdots \oplus b_{k^3-1}$.
3. Compute $x_i = a_i^{2^k} \in Q_{N_i}, i = 1, \ldots, k^3$.
4. Output $[\pi_{N_i}(x_i), r_i, (x_i \cdot r_i) \oplus b_i, i = 1, \ldots, k^3]$.

Decryption $\mathsf{Dec}(c)$:

1. Parse c as $[y_i, r_i, \beta_i, i = 1, \ldots, k^3]$.
2. Compute $b_i = (\pi^{-1}_{N_i}(y_i) \cdot r_i) \oplus \beta_i, i = 1, \ldots, k^3$.
3. Output $b_1 \oplus \cdots \oplus b_{k^3}$.

Oblivious key generation $\mathsf{oGen}(1^k)$:

1. Parse the randomness $\hat{r}_{\mathrm{G}} \in \{0,1\}^{k^4}$ as $N_1, \ldots, N_{k^3} \in \{0,1\}^k$.
2. Output $(N_1, \ldots, N_{k^3})$.

Trapdoor invertibility key generation $\mathsf{rGen}(r_{\mathrm{G}})$:

1. Run $\mathsf{Gen}(r_{\mathrm{G}})$ to obtain $\hat{r}_{\mathrm{G}} = (N_1, \ldots, N_{k^3})$.
2. Output $\hat{r}_{\mathrm{G}}$.

Oblivious sampling of ciphertexts $\mathsf{oRndEnc}(1^k)$:

1. Parse the randomness $\hat{r}_{\mathrm{E}}$ as $(\gamma_1, \ldots, \gamma_{k^3}) \in Z^*_{N_1} \times \cdots \times Z^*_{N_{k^3}}$, $s_1, \ldots, s_{k^3} \in \{0,1\}^k$ and $\beta_1, \ldots, \beta_{k^3} \in \{0,1\}$.
2. Compute $y_i = \gamma_i^{2^k} \in Q_{N_i}, i = 1, \ldots, k^3$.
3. Output $[y_i, s_i, \beta_i, i = 1, \ldots, k^3]$.

Trapdoor invertibility for ciphertexts $\mathsf{rRndEnc}(r_{\mathrm{G}}, r_{\mathrm{E}}, b)$:

1. Use r_{G} to compute $\mathsf{factor}(N_1), \ldots, \mathsf{factor}(N_k)$. and parse r_{E} as in Enc.
2. Set $s_i = r_i$ and $\beta_i = (x_i \cdot r_i) \oplus b_i$, $i = 1, \ldots, k^3$.
3. Pick a random γ_i uniformly from the set $\{\gamma_i \in Z^*_{N_i} \mid \gamma_i^{2^k} = \pi_{N_i}(x_i)\}$.
4. Output $\hat{r}_{\mathrm{E}} = (\gamma_1, \ldots, \gamma_{k^3}, s_1, \ldots, s_{k^3}, \beta_1, \ldots, \beta_{k^3})$.

Figure 4.4: Trapdoor Simulatable PKE from hardness of factoring Blum integers

Chapter 5

Simple, Black-Box Constructions of Adaptively Secure Protocols

5.1 Introduction

Secure multi-party computation (MPC) allows several mutually distrustful parties to perform a joint computation without compromising, to the greatest extent possible, the privacy of their inputs or the correctness of the outputs. An important criterion in evaluating the security guarantee is *how many* parties an adversary is allowed to corrupt and *when* the adversary determines which parties to corrupt. In this work, we focus on MPC protocols secure against an adversary that corrupts an arbitrary number of parties, and in addition, *adaptively* determines who and when to corrupt during the course of the computation. Even though the latter is a very natural and realistic assumption about the adversary, most of the MPC literature only addresses security against a static adversary, namely one that chooses (and fixes) which parties to corrupt before the protocol starts executing.

In the absence of an honest majority, secure MPC protocols can only be realized under computational assumptions. From both a theoretical and practical stand-point, it is desirable for these protocols to be based on general hardness assumptions, and in addition, to require only black-box access to the primitive guaranteed by the assumption (that is, the protocol refers only to the input/output behavior of the primitive). Indeed, the first MPC protocols achieving security without an honest majority [Goldreich *et al.*, 1988] require non-black-box access to the underlying

cryptographic primitives: the first step in the construction is to obtain protocols that are secure against semi-honest adversaries, and the second handles malicious behavior by having the parties prove in zero knowledge that they are adhering to the protocol constructions. It is the second step that requires the code of the underlying primitive with the use of general NP reductions to prove statements in zero knowledge. This aversely affects both computational complexity and communication complexity of the resulting protocol as well as the complexity of implementing the protocol.

In their seminal work, Ishai et al. [Ishai *et al.*, 2006] exhibited MPC protocols that are secure against a static adversary corrupting any number of parties and that rely only on black-box access to a low-level primitive, such as (enhanced) trapdoor permutations and homomorphic encryption schemes. This, along with the follow-up work of Haitner [Haitner, 2008], resolves the theoretical question of the minimal assumptions under which we may obtain black-box constructions of secure MPC protocols against a static adversary[1]. The main technical contribution in both works is to construct a secure protocol for a specific two-party functionality, that of oblivious transfer (OT). The general result then follows from a classic result of Kilian's [Kilian, 1988] showing that any multi-party functionality can be securely computed using black-box access to a secure OT protocol. However, none of these works addresses security against an adaptive adversary, which begs the following question:

> Is it possible to construct MPC protocols secure against a malicious, adaptive adversary that may corrupt any number of parties, given only black-box access to a low-level primitive?

Towards resolving this question, Ishai, Prabhakaran and Sahai [Ishai *et al.*, 2008] established an analogue of Kilian's result for an adaptive adversary. While there has been fairly extensive work on secure OT protocols against a static malicious adversary (e.g. [Naor and Pinkas, 2001; Kalai, 2005; Peikert *et al.*, 2008]), very few - namely [Beaver, 1998; Canetti *et al.*, 2002; Katz and Ostrovsky, 2004] - provide security against an adaptive adversary; moreover, all of those which do follow the [Goldreich *et al.*, 1988] paradigm and exploit non-black-box access to the underlying primitive.

[1] We note that subsequently, the works of [Ishai *et al.*, 2006] and [Haitner, 2008] were combined in a journal version [Haitner *et al.*, 2011], which we refer to henceforth.

5.1.1 Our results

Our main technical contribution is the construction of efficient OT protocols that achieve security against an adaptive adversary, while relying only upon black-box access to some low-level primitive. Specifically, we provide a compiler that transforms an OT protocol secure against a semi-honest, adaptive adversary into one that is secure against a malicious, adaptive adversary, given only black-box access to the underlying OT protocol and an "ideal" commitment scheme. In addition, we achieve security in the universal composability (UC) model, where a protocol may be executed concurrently with an unknown number of other protocols [Canetti, 2001]. This is a notable improvement over afore-mentioned works of Ishai et al. [Haitner *et al.*, 2011] which provide a compiler for semi-honest OT to malicious OT, but only for static adversaries in the stand-alone model.[2]

Theorem 5.1.1 *There exists a black-box construction of a protocol that UC realizes OT against a malicious, adaptive adversary in the $\mathcal{F}_{\text{COM}}$-hybrid model, starting from any protocol that UC realizes OT against a semi-honest, adaptive adversary.*[3] *Moreover, the construction achieves a constant multiplicative blow-up in the number of rounds.*

We note that there are two models for adaptive corruptions: with and without secure erasures, and our result applies to both models.

Our construction also improves upon the earlier work of Canetti et. al [Canetti *et al.*, 2002] achieving the same guarantee; their construction is non-black-box and incurs a blow-up in round complexity proportional to the depth of the circuit computing the semi-honest OT protocol. Combined with the 2-round semi-honest OT protocol in [Canetti *et al.*, 2002; Choi *et al.*, 2009a], we obtain the first constant-round protocol for OT in the $\mathcal{F}_{\text{COM}}$-hybrid model secure against a malicious, adaptive adversary.[4] Moreover, the protocol uses black-box access to a

[2]We note that our construction does not improve on the computational complexity of the previous compiler, as measured by the number of invocations of the underlying semi-honest OT protocol. However, we believe our construction may be combined with the OT extension protocol in [Ishai *et al.*, 2008, Section 5.3] to achieve better efficiency.

[3]In both the semi-honest and the malicious OT protocols, we allow the adaptive adversary to corrupt both the sender and the receiver.

[4]In an independent work [Garay *et al.*, 2009], Garay, Wichs and Zhou also constructed a constant-round protocol for OT in the common reference string model, secure against a malicious, adaptive adversary. Their underlying assumptions are comparatively more restrictive.

Functionality $\mathcal{F}_{\text{COM}}$

1. Upon receiving input $(\mathsf{commit}, sid, P_j, x)$ from P_i where $x \in \{0,1\}^m$, internally record the tuple (P_i, P_j, x) and send the message (sid, P_i, P_j) to the adversary; When receiving $(\mathtt{ok})$ from the adversary, output $(\mathsf{receipt}, sid, P_i)$ to P_j. Ignore all subsequent $(\mathsf{commit}, ...)$ inputs.
2. Upon receiving a value (reveal, sid) from P_i, where a tuple (P_i, P_j, x) is recorded, send (x) to the adversary; When receiving $(\mathtt{ok})$ from the adversary, output $(\mathsf{reveal}, sid, x)$ to P_j.

Figure 5.1: String Commitment Functionality

Functionality $\mathcal{F}_{\text{OT}}$

1. Upon receiving input $(\mathsf{receiver}, sid, r)$ from R where $sid = (\mathsf{S}, \mathsf{R}, sid')$ and $r \in \{0,1\}$, record the tuple (sid, r) and send $(\mathsf{recevier}, sid)$ to S and the adversary.
2. Upon receiving input $(\mathsf{sender}, sid, s_0, s_1)$ from S where $sid = (\mathsf{S}, \mathsf{R}, sid')$ and $s_0, s_1 \in \{0,1\}^\ell$, send $(\mathsf{output}, sid, s_r)$ to R and (output, sid) to the adversary and halt. If no $(\mathsf{receiver}, sid, \ldots)$ message was previously sent, ignore the message.

Figure 5.2: Oblivious Transfer Functionality

low-level primitive, that of trapdoor simulatable cryptosystems[5], which may in turn be based on the RSA, DDH, worst-case lattice assumptions or hardness of factoring.

The key conceptual insight underlying the construction is to view the [Haitner *et al.*, 2011] compiler as an instantiation of the [Goldreich *et al.*, 1988] paradigm in the $\mathcal{F}_{\text{COM}}$-hybrid model, except enforcing consistency via cut-and-choose techniques instead of using zero-knowledge proofs. This perspective leads naturally to a simpler, more modular, and more direct analysis of the previous compiler for static adversaries. In addition, we immediately obtain a string OT protocol, which is important for obtaining round-efficient MPC protocols [Lindell and Pinkas, 2007; Ishai *et al.*, 2008]. Showing that the modified compiler achieves UC security against an adaptive adversary requires new insight in constructing a simulator and in the analysis. We defer a more detailed discussion of the construction to Section 5.2, and instead focus here on the applications to secure MPC derived by combining our OT protocol with various MPC protocols in the $\mathcal{F}_{\text{OT}}$-hybrid model in [Ishai *et al.*, 2008].

[5]Trapdoor simulatable cryptosystems are introduced in [Choi *et al.*, 2009a], as a relaxation of simulatable cryptosystems [Damgård and Nielsen, 2000]. These are semantically secure encryption schemes with special algorithms for "obliviously" sampling public keys and ciphertexts without learning the respective secret keys and plaintexts. In addition, both of these oblivious sampling algorithms are efficiently invertible given the corresponding secret key.

MPC in the $\mathcal{F}_{\text{COM}}$-hybrid model.

Combining our OT protocol with [Ishai *et al.*, 2008, Theorem 2], we obtain UC-secure MPC protocols in the $\mathcal{F}_{\text{COM}}$-hybrid model against a malicious, adaptive adversary, which improves upon [Canetti *et al.*, 2002] in that we only require black-box access to the underlying primitive:

Theorem 5.1.2 *There exists a protocol in the $\mathcal{F}_{\text{COM}}$-hybrid model that uses black-box access to a (trapdoor) simulatable cryptosystem and UC realizes any well-formed multi-party functionality against a malicious, adaptive adversary that may corrupt any number of parties.*

The round complexity of the protocol is proportional to the depth of the circuit computing the functionality. By combining our OT protocol with [Ishai *et al.*, 2008, Theorem 3], we obtain a constant-round MPC protocol in the $\mathcal{F}_{\text{COM}}$ with the same guarantees, except that the adversary is limited to corrupting up to $m - 1$ parties for a m-party functionality.

The advantage of constructing UC-secure MPC protocols in the $\mathcal{F}_{\text{COM}}$-hybrid model is that they may be combined with many of the existing UC feasibility results under various set-up or modeling assumptions e.g. [Canetti *et al.*, 2002; Barak *et al.*, 2004; Canetti *et al.*, 2007; Katz, 2007], almost all of which start by showing how to UC realize $\mathcal{F}_{\text{COM}}$ in some new security model[6]. Moreover, if the protocol realizing $\mathcal{F}_{\text{COM}}$ uses black-box access to a low-level primitive, so will the combined protocol.

With a slight modification to our compiler $\mathsf{Comp}(\Pi)$, we obtain an analogous result for the case where we start with Π that is secure against semi-honest, static adversaries and end with $\mathsf{Comp}(\Pi)$ secure against malicious, static adversaries. Combining our OT protocol with [Ishai *et al.*, 2008, Theorem 2], as before, we obtain UC-secure MPC protocols in the $\mathcal{F}_{\text{COM}}$-hybrid model against a malicious, static adversary:

Theorem 5.1.3 *There exists a protocol in the $\mathcal{F}_{\text{COM}}$-hybrid model that uses black-box access to an (enhanced) trapdoor permutation and UC realizes any well-formed multi-party functionality against a malicious, static adversary that may corrupt any number of parties.*

[6]This is because it is impossible to UC realize any non-trivial 2-party functionality in the plain model (even against static adversaries) [Canetti *et al.*, 2003; Canetti, 2001].

MPC in the standalone model.

Next, we present our results for the stand-alone model with adaptive post-execution corruptions [Canetti, 2000], which is a weaker notion of security than UC security with adaptive corruptions (and in particular, our protocols in the $\mathcal{F}_{\text{COM}}$-hybrid model achieve this notion of security). Here, there is a two-party protocol that uses black-box access to a one-way function and securely realizes $\mathcal{F}_{\text{COM}}$ in the plain model without any set-up assumptions [Pass and Wee, 2009]. This yields the following corollaries (via the composition theorem in [Canetti, 2000]):

Corollary 5.1.4 *There exists a constant-round string OT protocol that uses black-box access to a (trapdoor) simulatable cryptosystem and is secure in the stand-alone model against a malicious, adaptive adversary.*

Corollary 5.1.5 *There exists a protocol that uses black-box access to a (trapdoor) simulatable cryptosystem and securely computes any well-formed multi-party functionality in the stand-alone model against a malicious, adaptive adversary that may corrupt any number of parties.*

Both of these results hold in the setting without erasures and improve on the work of Beaver [Beaver, 1998] which achieve similar security guarantees but with non-black-box access to the underlying primitive. Furthermore, if we assume secure erasures, then we may relax the assumption in both results to any (constant-round) semi-honest OT protocol against static adversaries. Here, we also rely on a result of Lindell's that transforms a semi-honest OT protocol against static adversaries into one that is secure against adaptive adversaries with secure erasures [Lindell, 2009].

5.1.2 Follow-up work

A variant of our compiler and simulation strategy was used in the follow-up works of Wee [Wee, 2010] and Goyal [Goyal, 2011] to obtain black-box, constant-round protocols for secure multi-party computation in the stand-alone model.

5.2 Construction

5.2.1 High-Level Description

We provide an overview of the [Haitner *et al.*, 2011] compiler (which is in turn reminiscent of the constructions in [Crépeau *et al.*, 1995]). Our presentation is slightly different from, and simpler than, that in the original works, and is closer in spirit to the [Goldreich *et al.*, 1988] compiler. Our presentation is easier to adapt to the UC setting and the adaptive setting (and also OT with strings instead of bits) since we do not need to rely on the intermediate notion and construction of a defensible OT protocol.[7] We focus on the main transformation Comp (shown in Fig 5.3), which "boosts" the security guarantee of an OT protocol Π from security against semi-honest receivers to security against malicious receivers while preserving the security guarantee for corrupted senders.

Phase I: Random tape generation. The sender and the receiver engage in a coin-tossing (in the well) protocol to determine a collection of $2n$ random strings for the receiver.

Phase II: Basic execution. The sender and the receiver engage in $2n$ parallel executions of Π with random inputs: the sender will choose its own inputs randomly and independently for each of the $2n$ executions, whereas the inputs and randomness for the receiver are determined by the preceding coin-tossing protocol (one random string for each execution of Π).

Phase III: Cut-and-choose. The sender and the receiver engage in a coin-tossing protocol to pick a random subset Q of n executions, and the receiver proves that it acted accordingly to Π for the n executions in Q by revealing the inputs and randomness used for those executions. The sender verifies that the inputs and randomness are indeed consistent with both the n executions of Π and the coin-tossing protocol, and if so, we are guaranteed that the receiver must have behaved honestly in at least one of the n executions of Π not in Q (except with negligible probability). Otherwise, the sender detects the malicious behavior of the receiver and aborts the protocol. Note that in this case the sender's security is preserved since, although

[7] Specifically, the previous compiler proceeds in two phases. The first [Haitner, 2008] transforms any semi-honest OT protocol into defensible OT protocols. A defensible OT protocol provides an intermediate level of security interpolating semi-honest and malicious OT. The second [Ishai *et al.*, 2006] transforms any defensible OT protocol into a malicious one.

the receiver may have learned all of the sender's random inputs in Phase II by behaving maliciously in all $2n$ OT executions, this information reveals nothing about the sender's true inputs.

Phase IV: Combiner. We may then apply a combiner that (essentially) yields a single secure OT protocol, starting a collection of n OT protocols all of which guarantee security against a malicious sender, but only one of which guarantee security against a malicious receiver.

To obtain a full-fledged string-OT protocol secure against both a malicious sender and a malicious receiver starting from a semi-honest bit-OT protocol, we proceed as in [Ishai *et al.*, 2006], with the addition of Step 3 to directly obtain a string-OT protocol and with references to semi-honest instead of defensible adversaries:

1. Use Comp to obtain a bit-OT protocol secure against a semi-honest sender and a malicious receiver.
2. Use OT reversal [Wolf and Wullschleger, 2006] (shown in Fig 5.4) to obtain a bit-OT protocol secure against a malicious sender and a semi-honest receiver.
3. Repeat in parallel to obtain a string-OT protocol secure against a malicious sender and a semi-honest receiver.
4. Use Comp again to obtain a string-OT protocol secure against malicious sender and receiver.

In this work, we will view the above construction in the $\mathcal{F}_{\text{COM}}$-hybrid model, where the $\mathcal{F}_{\text{COM}}$ functionality is used to implement the coin-tossing protocol in Phases I and III [Blum, 1981; Canetti and Rabin, 2003].

5.2.2 Achieving Security against Adaptive Adversaries

The main technical challenge in coping with adaptive adversaries in $\mathsf{Comp}(\Pi)$ lies in simulating the sender messages in Phase II of $\mathsf{Comp}(\Pi)$ for the case of a malicious receiver and honest sender. One possible approach is to simply run the honest sender's code in Phase II with random inputs. However, this approach does not allow us to handle the case that the adversary chooses to corrupt

INITIALIZATION.

Sender input: $(\mathsf{sender}, \mathit{sid}, s_0, s_1)$ where $s_0, s_1 \in \{0,1\}^\ell$.

Receiver input: $(\mathsf{receiver}, \mathit{sid}, r)$ where $r \in \{0,1\}$.

PHASE I: RANDOM TAPE GENERATION.

1. R chooses $2n$ random strings $(r_1^{\mathsf{R}}, \tau_1^{\mathsf{R}}), \ldots, (r_{2n}^{\mathsf{R}}, \tau_{2n}^{\mathsf{R}})$ and sends $(\mathsf{commit}, \mathit{sid}_i, r_i^{\mathsf{R}}, \tau_i^{\mathsf{R}})$ to $\mathcal{F}_{\text{COM}}$, for $i = 1, 2, \ldots, 2n$.
2. Upon receiving $(\mathsf{receipt}, \mathit{sid}_1), \ldots, (\mathsf{receipt}, \mathit{sid}_{2n})$ from $\mathcal{F}_{\text{COM}}$, S sends $2n$ random strings $(r_1^{\mathsf{S}}, \tau_1^{\mathsf{S}}), \ldots, (r_{2n}^{\mathsf{S}}, \tau_{2n}^{\mathsf{S}})$.
3. R sets $r_i = r_i^{\mathsf{R}} \oplus r_i^{\mathsf{S}}$ and $\tau_i = \tau_i^{\mathsf{R}} \oplus \tau_i^{\mathsf{S}}$, for $i = 1, 2, \ldots, 2n$.

PHASE II: BASIC EXECUTION.

1. S chooses $2n$ pairs of random inputs $(s_1^0, s_1^1), \ldots, (s_{2n}^0, s_{2n}^1)$.
2. S and R engages in $2n$ parallel executions of the protocol Π. In the ith execution, S inputs (s_i^0, s_i^1) and R inputs r_i with randomness τ_i and obtains output $s_i^{r_i}$.

PHASE III: CUT-AND-CHOOSE.

1. S chooses a random $q = (q_1, \ldots, q_n) \in \{0,1\}^n$. The string q is used to define a set of indices $Q \subset \{1, 2, \ldots, 2n\}$ of size n in the following way: $Q = \{2i - q_i\}_{i=1}^n$.
2. For every $i \in Q$, R sends $(\mathsf{reveal}, \mathit{sid}_i)$ to $\mathcal{F}_{\text{COM}}$ and upon receiving $(\mathsf{reveal}, \mathit{sid}_i, r_i^{\mathsf{R}}, \tau_i^{\mathsf{R}})$ from $\mathcal{F}_{\text{COM}}$, S computes (r_i, τ_i).
3. S checks that for all $i \in Q$, (r_i, τ_i) is consistent with R's messages in the i'th execution of Π. If not, S aborts and halts.

PHASE IV: COMBINER.

1. For every $j \notin Q$, R computes $\alpha_j = r \oplus r_j$ and sends $\{\alpha_j\}_{j \notin Q}$ to S.
2. S computes $\sigma_0 = s_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and $\sigma_1 = s_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$ and sends (σ_0, σ_1).
3. R computes and outputs $s_r = \sigma_r \oplus (\bigoplus_{j \notin Q} s_j^{r_j})$.

Figure 5.3: THE ADAPTIVE COMPILER $\mathsf{Comp}(\Pi)$

the sender at the end of the protocol: Once the sender's inputs are revealed, we cannot go back and change the random inputs used in Phase II to be consistent with these inputs. Another approach is to use the simulator for Π to simulate sender messages in Phase II for the case of honest sender and corrupt receiver. The problem with this approach is that the receiver may behave maliciously in some of the executions in Phase II and Π is only guaranteed to be simulatable when the receiver behaves semi-honestly. What we actually do is combine the above two approaches and simulate honest sender messages in Phase II against a malicious receiver as follows:

1. For each i, extract the receiver's input and randomness for the i'th execution of Π from the commitment in Phase I.

2. Upon receiving a message from the receiver in the i'th execution of Π, check if all of the receiver's messages so far are consistent with its input and randomness. If so, generate the sender's response by using the simulator for Π. Otherwise, corrupt the sender in the i'th execution of Π to obtain its input and random tape and complete the simulation of the sender's messages using the honest sender strategy.

We note that, unlike the compiler of [Haitner *et al.*, 2011], our compiler does not require a coin-tossing protocol to determine the set opened in the cut-and-choose in Phase III. Instead, this set is chosen by the sender alone.

5.2.3 Achieving Security against Static Adversaries

Our analysis also extends to the UC static case. More specifically, we prove the following analogue of Theorem 5.1.1:

Theorem 5.2.1 *There exists a black-box construction of a protocol that UC realizes string-OT against a static, malicious adversary in the $\mathcal{F}_{\text{COM}}$-hybrid, starting from any bit-OT protocol secure against a static, semi-honest adversary in the stand-alone model. Moreover, the construction achieves a constant multiplicative blow-up in the number of rounds.*

We note that, unlike in the adaptive case, our compiler for static adversaries requires the parties to run a coin-tossing protocol in Phase III. The reason that coin-tossing is required in the static case but

not the adaptive case seems closely related to the selective decommitment problem which occurs in the static case but not the adaptive case (see, [Dwork *et al.*, 2003]). Nevertheless, our result and analysis for static adversaries offers several improvements over that in [Haitner *et al.*, 2011]:

- The simulator in [Ishai *et al.*, 2006] uses rewinding and runs in expected polynomial time, even in the $\mathcal{F}_{\text{COM}}$-hybrid model.
- Our result immediately yields string-OT protocols and in a constant number of rounds.
- We eliminate several complications of the analysis in both [Ishai *et al.*, 2006] and [Haitner, 2008], most notably verifying that the OT reversal protocol in [Wolf and Wullschleger, 2006] works for defensible adversaries [Ishai *et al.*, 2006, Claim 5.2]. The overall analysis is simpler, more modular, and more intuitive.

Organization.

We present our analysis of Comp and OT reversal for adaptive adversaries in the UC model in Sections 5.3 and 5.4 respectively. We deal with the case of static adversaries in Section 5.5.

5.3 Achieving security against a malicious receiver

In this section, we show that Comp boosts the security guarantee from security against semi-honest receivers to security against malicious receivers.

Proposition 5.3.1 *Suppose* Π *is a protocol that UC realizes* $\mathcal{F}_{\text{OT}}$ *against a semi-honest, adaptive adversary, and let* $\mathsf{Comp}(\Pi)$ *be the protocol obtained by applying the compiler in Fig 5.3 to* Π*. Then,* $\mathsf{Comp}(\Pi)$ *UC realizes* $\mathcal{F}_{\text{OT}}$ *against an adaptive adversary with a semi-honest sender and a malicious receiver. Moreover, if* Π *is in addition secure against a malicious, adaptive sender, then* $\mathsf{Comp}(\Pi)$ *UC realizes* $\mathcal{F}_{\text{OT}}$ *against an adaptive adversary with malicious sender and receiver.*

A hybrid execution.

To facilitate the analysis, we introduce an intermediate setting (inspired by [Ishai *et al.*, 2007]) in which the protocol $\mathsf{Comp}(\Pi)$ is executed, where there is again a sender S and a receiver R

and in addition $2n$ pairs of "virtual" parties $(\mathsf{S}_1, \mathsf{R}_1), \ldots, (\mathsf{S}_{2n}, \mathsf{R}_{2n})$. The i'th execution of Π in $\mathsf{Comp}(\Pi)$ will be executed by S_i and R_i with inputs from S and R respectively. We will require that $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}$ are always semi-honest; i.e. they use a truly random tape for Π. Moreover, the environment is not aware of the "virtual parties".

PHASE I/II: BASIC EXECUTION.[8] S chooses $2n$ pairs of random inputs $(s_1^0, s_1^1), \ldots, (s_{2n}^0, s_{2n}^1)$ and R chooses $2n$ random inputs $r_1, \ldots, r_{2n}$. For each $i = 1, \ldots, 2n$, S activates S_i with $(\mathsf{sender}, \mathit{sid}_i, s_i^0, s_i^1)$ and R activates R_i with $(\mathsf{receiver}, \mathit{sid}_i, r_i)$. In $\text{HYBRID}_{\Pi,\mathcal{A},\mathcal{Z}}$, the parties S_i and R_i execute Π in parallel. In $\text{HYBRID}_{\mathcal{F}_{\text{OT}},\mathcal{A},\mathcal{Z}}$, the parties S_i and R_i call the ideal functionality $\mathcal{F}_{\text{OT}}$.

PHASE III: CUT-AND-CHOOSE. S chooses a random $q \in \{0,1\}^n$ which identifies $Q \subset \{1, 2, \ldots, 2n\}$ as in $\mathsf{Comp}(\Pi)$ and sends q to R. S checks that for all $i \in Q$, S_i is not corrupted. Otherwise, abort.

PHASE IV: COMBINER. Proceed as in Phase IV of $\mathsf{Comp}(\Pi)$.

We say that an adversary $\mathcal{A}$ in the hybrid execution is well-formed if it satisfies the following properties:

- When $\mathcal{A}$ corrupts S, it also corrupts each of $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$. Moreover, if S is semi-honest, then $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ are semi-honest.
- When $\mathcal{A}$ corrupts R, it also corrupts each of $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}$. Moreover, $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}$ are always semi-honest, even if R is malicious.
- If R is corrupted, then $\mathcal{A}$ may corrupt any of $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ with semi-honest behavior, without corrupting S.
- Upon receiving the set Q in Phase III from S, $\mathcal{A}$ may corrupt all of $\mathsf{R}_j, j \in Q$ with semi-honest behavior, even if neither R nor S is corrupted. However, if R is not corrupted, then $\mathsf{R}_j, j \notin Q$ are also not corrupted.

[8] The choice of notation is so that Phase III always corresponds to cut-and-choose and Phase IV corresponds to combiner in both $\mathsf{Comp}(\Pi)$ and in the hybrid executions.

We will also stipulate that the communication channels between S and each of $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ are private and authenticated. The same holds for the communication channels between R and each of $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}$. In addition, S learns whether each of $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ is corrupted.

Lemma 5.3.2 *For every adversary $\mathcal{A}$ that interacts with $\mathsf{Comp}(\Pi)$ in the $\mathcal{F}_{\text{COM}}$-hybrid model, there exists a well-formed adversary $\mathcal{A}'$ that interacts in the hybrid execution running Π, such that for every environment $\mathcal{Z}$,*

$$\text{EXEC}^{\mathcal{F}_{\text{COM}}}_{\mathsf{Comp}(\Pi),\mathcal{A},\mathcal{Z}} \equiv \text{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}}$$

The statement of the lemma holds for protocols Π that are secure against semi-honest, adaptive adversaries without erasures and well-formed adaptive adversaries $\mathcal{A}'$ and $\mathcal{A}''$. In the first step, we show how to enforce semi-honest behavior of $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}$ in $\text{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}}$. The high-level strategy is as follows: if a corrupted receiver in $\mathsf{Comp}(\Pi)$ deviates from semi-honest behavior in the i'th execution of Π in Phase II, we corrupt S_i in $\text{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}}$ to obtain its input and randomness, and continue the simulation by running the honest sender strategy. **Proof:** As usual, $\mathcal{A}'$ works by invoking a copy of $\mathcal{A}$ and running a simulated interaction of $\mathcal{A}$ with $\mathcal{Z}$ and the parties S and R. We will refer to the communication of $\mathcal{A}'$ with $\mathcal{Z}$ and $\mathsf{Comp}(\Pi)$ as external communication, and that with the simulated $\mathcal{A}$ as internal communication. More precisely, $\mathcal{A}'$ works as follows:

Simulating the communication with $\mathcal{Z}$: Every input value that $\mathcal{A}'$ receives from $\mathcal{Z}$ externally is written into the adversary $\mathcal{A}$'s input tape (as if coming from $\mathcal{A}$'s environment). Every output value written by $\mathcal{A}$ on its output tape is copied to $\mathcal{A}'$'s own output tape (to be read by the external $\mathcal{Z}$).

Simulating the case when neither party is corrupted:

PHASE I. $\mathcal{A}'$ internally passes $\mathcal{A}$ the message $(\mathsf{receipt}, \mathit{sid}_1), (\mathsf{receipt}, \mathit{sid}_2), \ldots, (\mathsf{receipt}, \mathit{sid}_{2n})$ as if sent from $\mathcal{F}_{\text{COM}}$ to S. Then, $\mathcal{A}'$ chooses $2n$ random strings $(r_1^{\mathsf{S}}, \tau_1^{\mathsf{S}}), \ldots, (r_{2n}^{\mathsf{S}}, \tau_{2n}^{\mathsf{S}})$, and simulates S sending R those $2n$ strings.

PHASE II. For each round in the protocol Π, if it is the receiver's turn, then for each $i = 1, \ldots, 2n$, $\mathcal{A}'$ obtains β_i from R_i for the corresponding round. Next, $\mathcal{A}'$ internally passes

$\mathcal{A}$ the message $(\beta_1, \ldots, \beta_{2n})$, as if sent from R to S. The sender's turn is handled analogously.

PHASE III. When S externally sends q which determines Q, then for each $i \in Q$: corrupt R_i to obtain (r_i, τ_i) and compute $r_i^{\mathrm{R}} = r_i \oplus r_i^{\mathrm{S}}$ and $\tau_i^{\mathrm{R}} = \tau_i \oplus \tau_i^{\mathrm{S}}$. Send $(\mathsf{reveal}, \mathit{sid}_i, r_i^{\mathrm{R}}, \tau_i^{\mathrm{R}})$ to $\mathcal{A}$ as if coming from $\mathcal{F}_{\mathrm{COM}}$.

PHASE IV. Just forward all the messages between S and R.

Simulating the case when only the sender is corrupted: This is essentially the same as when neither party is corrupted, except the values $(r_1^{\mathrm{S}}, \tau_1^{\mathrm{S}}), \ldots, (r_{2n}^{\mathrm{S}}, \tau_{2n}^{\mathrm{S}})$ in Phase I, sender messages in Phase II, and the value q in Phase III are chosen by $\mathcal{A}$.

Simulating the case when only the receiver is corrupted:

PHASE I. $\mathcal{A}'$ externally corrupts $(\mathsf{R}_1, \ldots, \mathsf{R}_{2n})$ and chooses $(r_1, \ldots, r_{2n})$ and $(\tau_1, \ldots, \tau_{2n})$ at random. Next, $\mathcal{A}'$ obtains from $\mathcal{A}$ the messages $(\mathsf{commit}, \mathit{sid}_i, r_i^{\mathrm{R}}, \tau_i^{\mathrm{R}})$ as sent by R to $\mathcal{F}_{\mathrm{COM}}$. Then, $\mathcal{A}'$ sets $r_i^{\mathrm{S}} = r_i \oplus r_i^{\mathrm{R}}$ and $\tau_i^{\mathrm{S}} = \tau_i \oplus \tau_i^{\mathrm{R}}$ for $i = 1, 2, \ldots, 2n$ and internally passes $(r_1^{\mathrm{S}}, \tau_1^{\mathrm{S}}), \ldots, (r_{2n}^{\mathrm{S}}, \tau_{2n}^{\mathrm{S}})$ to $\mathcal{A}$ as if sent by S to R.

PHASE II. We need to simulate the external messages sent by S in $\mathsf{Comp}(\Pi)$ (with the "help" of $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$). If R behaves consistently in the ith execution of Π, we will just obtain the corresponding message from S_i; otherwise, we will corrupt S_i so that we may compute those messages.

First, we handle receiver messages in $\mathsf{Comp}(\Pi)$. Whenever $\mathcal{A}$ sends a message $(\beta_1, \ldots, \beta_{2n})$ from R where β_i is the message in the ith parallel execution of Π, do the following for each $i = 1, \ldots, 2n$:

- If R_i has not aborted and β_i is consistent with (r_i, τ_i), deliver the corresponding message from R_i to S_i.
- If R_i has not aborted and β_i is not consistent with (r_i, τ_i), $\mathcal{A}'$ tells R_i to abort. In addition, $\mathcal{A}'$ corrupts S_i to obtain its input (s_i^0, s_i^1) and its randomness.
- If R_i has aborted, then record β_i and do nothing.

Next, we handle sender messages in $\mathsf{Comp}(\Pi)$. Whenever $\mathcal{A}$ expects a message $(\gamma_1, \ldots, \gamma_{2n})$ from S, where γ_i is the message in the ith parallel execution of Π, do the following for each $i = 1, \ldots, 2n$:

- If S_i is corrupted, then $\mathcal{A}'$ computes γ_i according to S_i's input and randomness and the previous messages from R_i.
- If S_i is not corrupted, then set γ_i to be the corresponding message sent from S_i to R_i.

$\mathcal{A}'$ then sends $(\gamma_1, \ldots, \gamma_{2n})$ to $\mathcal{A}$ as if sent by S to R.

PHASE III. Deliver q sent externally by S to R. Check that for all $i \in Q$, S_i is not corrupted. Otherwise, abort.

PHASE IV. Just forward all the messages between S and R.

Dealing with corruption of parties: When the simulated $\mathcal{A}$ internally corrupts R, $\mathcal{A}'$ externally corrupts R and thus $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}$, and learns the values $r_1, \ldots, r_{2n}$ and $\tau_1, \ldots, \tau_{2n}$ (in addition to the input r). $\mathcal{A}'$ then sets $r_i^{\mathrm{R}} = r_i \oplus r_i^{\mathrm{S}}$ and $\tau_i^{\mathrm{R}} = \tau_i \oplus \tau_i^{\mathrm{S}}$ for $i = 1, 2, \ldots, 2n$ and internally passes $(r_1^{\mathrm{R}}, \tau_1^{\mathrm{R}}), \ldots, (r_{2n}^{\mathrm{R}}, \tau_{2n}^{\mathrm{R}})$ to $\mathcal{A}$ as the randomness for R in $\mathsf{Comp}(\Pi)$. Similarly, when the simulated $\mathcal{A}$ internally corrupts S, $\mathcal{A}'$ externally corrupts S and thus $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ and learns the values $(s_1^0, s_1^1), \ldots, (s_{2n}^0, s_{2n}^1)$ along with the randomness used by $\mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ in the $2n$ executions of Π. $\mathcal{A}'$ then internally passes all of these values to $\mathcal{A}$ as the randomness for S in $\mathsf{Comp}(\Pi)$. In addition, for all $i \in Q$, $\mathcal{A}'$ passes the value $(r_i^{\mathrm{R}}, \tau_i^{\mathrm{R}})$ to $\mathcal{A}$ as the value sent from $\mathcal{F}_{\mathrm{COM}}$ to S in Phase III.

It is straight-forward to verify that in Phase III, checking S_i is not corrupted in $\mathrm{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}}$ is identical to R behaving consistently in the ith execution of Π in $\mathsf{Comp}(\Pi)$. Thus, the abort condition at the end of Phase III are identical. We may therefore conclude that the ensembles EXEC and HYBRID are identical. ■

Lemma 5.3.3 *For every well-formed adversary $\mathcal{A}'$ that interacts in the hybrid execution running Π, there exists a well-formed adversary $\mathcal{A}''$ that interacts in the hybrid execution running $\mathcal{F}_{\mathrm{OT}}$, such*

that for every environment $\mathcal{Z}$,

$$\text{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}} \stackrel{c}{\approx} \text{HYBRID}_{\mathcal{F}_{\text{OT}},\mathcal{A}'',\mathcal{Z}}$$

The statement of the lemma is intended for protocols Π that are secure against semi-honest, adaptive adversaries without erasures and well-formed adaptive adversaries $\mathcal{A}'$ and $\mathcal{A}''$. However, we note that the lemma also holds for protocols Π that are secure against semi-honest, adaptive adversaries *with* erasures and well-formed adaptive adversaries $\mathcal{A}'$ and $\mathcal{A}''$ *with* erasures as well as protocols Π that are secure against semi-honest, static adversaries and well-formed static adversaries $\mathcal{A}'$ and $\mathcal{A}''$.

Proof: The idea is that we may interpret $\text{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}}$ as an execution involving $4n + 2$ parties $\mathsf{S}, \mathsf{R}, \mathsf{S}_1, \ldots, \mathsf{S}_{2n}, \mathsf{R}_1, \ldots, \mathsf{R}_{2n}$ jointly running some protocol that uses Π as a sub-routine, and $\text{HYBRID}_{\mathcal{F}_{\text{OT}},\mathcal{A}'',\mathcal{Z}}$ as an execution involving the same $4n+2$ parties running the same protocol except with an ideal $\mathcal{F}_{\text{OT}}$ functionality instead of Π. The claim then follows from the UC composition [Canetti, 2001]. ■

Lemma 5.3.4 *For every well-formed adversary $\mathcal{A}''$ that interacts in the hybrid execution running $\mathcal{F}_{\text{OT}}$, there exists an ideal-process adversary $\mathcal{S}$, such that for every environment $\mathcal{Z}$,*

$$\text{HYBRID}_{\mathcal{F}_{\text{OT}},\mathcal{A}'',\mathcal{Z}} \approx_s \text{IDEAL}_{\mathcal{F}_{\text{OT}},\mathcal{S},\mathcal{Z}}$$

The statement of the lemma is intended for adaptive adversaries $\mathcal{A}''$ and $\mathcal{S}$. However, we note that the lemma also holds for adaptive adversaries $\mathcal{A}''$ and $\mathcal{S}$ *with* erasures as well as static adversaries $\mathcal{A}''$ and $\mathcal{S}$.

Proof: Again, we first specify $\mathcal{S}$ depending on the corruption pattern:

Simulating the communication with $\mathcal{Z}$: Every input value that $\mathcal{S}$ receives from $\mathcal{Z}$ externally is written into the adversary $\mathcal{A}''$'s input tape (as if coming from $\mathcal{A}''$'s environment). Every output value written by $\mathcal{A}''$ on its output tape is copied to $\mathcal{S}$'s own output tape (to be read by the external $\mathcal{Z}$).

Simulating the case when neither party is corrupted:

PHASE I/II. Internally pass $(\mathsf{receiver}, \mathit{sid}_1)$, ..., $(\mathsf{receiver}, \mathit{sid}_{2n})$ and $(\mathsf{output}, \mathit{sid}_1)$, ..., $(\mathsf{output}, \mathit{sid}_{2n})$ to $\mathcal{A}''$ as if $\mathcal{F}_{\text{OT}}$ did.

PHASE III. Internally pass a random $q \in \{0,1\}^n$ to $\mathcal{A}''$ as if S sent it to R. For each $i \in Q$, when $\mathcal{A}''$ corrupts R_i, pick a random $r_i \in \{0,1\}$ and a random $s_i^{r_i} \in \{0,1\}^\ell$.

PHASE IV. Wait until $(\mathsf{receiver}, \mathit{sid})$ comes externally from $\mathcal{F}_{\text{OT}}$. Send random $\{\alpha_j\}_{j \notin Q}$ as if sent from R. Wait until $(\mathsf{output}, \mathit{sid})$ comes externally from $\mathcal{F}_{\text{OT}}$. Internally pass random (σ_0, σ_1) to $\mathcal{A}''$ as if S sent the pair to R.

Simulating the case when only the sender is corrupted:

PHASE I/II. Internally pass $(\mathsf{receiver}, \mathit{sid}_1)$, ..., $(\mathsf{receiver}, \mathit{sid}_{2n})$ to $\mathcal{A}''$ as if $\mathcal{F}_{\text{OT}}$ did. When $\mathcal{A}''$ as S_i sends $(\mathsf{sender}, \mathit{sid}_i, s_i^0, s_i^1)$ to $\mathcal{F}_{\text{OT}}$, $\mathcal{S}$ records (s_i^0, s_i^1) and internally passes $(\mathsf{output}, \mathit{sid}_1)$, ..., $(\mathsf{output}, \mathit{sid}_{2n})$ to $\mathcal{A}''$ as if $\mathcal{F}_{\text{OT}}$ did.

PHASE III. Proceed as in the case neither party is corrupted, except q is chosen by $\mathcal{A}''$.

PHASE IV. Wait until $(\mathsf{receiver}, \mathit{sid})$ comes externally from $\mathcal{F}_{\text{OT}}$. Internally pass random $\{\alpha_j\}_{j \notin Q}$ to $\mathcal{A}''$ as if R did. When $\mathcal{A}''$ as S sends (σ_0, σ_1), compute $s_0 = \sigma_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and $s_1 = \sigma_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$. Externally send $(\mathsf{sender}, s_0, s_1)$ to $\mathcal{F}_{\text{OT}}$.

Simulating the case when only the receiver is corrupted:

PHASE I/II. $\mathcal{S}$ picks $2n$ pairs of random inputs $(s_1^0, s_1^1), \ldots, (s_{2n}^0, s_{2n}^1)$. If $\mathcal{A}''$ as R_i sends $(\mathsf{receiver}, \mathit{sid}_i, r_i)$ to $\mathcal{F}_{\text{OT}}$, $\mathcal{S}$ records r_i and internally passes $(\mathsf{receiver}, \mathit{sid}_i)$ to $\mathcal{A}''$ as if $\mathcal{F}_{\text{OT}}$ did. Then, $\mathcal{S}$ internally passes $(\mathsf{output}, \mathit{sid}_i, s_i^{r_i})$ to $\mathcal{A}''$ as if $\mathcal{F}_{\text{OT}}$ sent the message to R_i. If $\mathcal{A}''$ corrupts S_i, then $\mathcal{S}$ presents (s_i^0, s_i^1) as S_i's input to $\mathcal{A}''$.

PHASE III. Pick a random $q \in \{0,1\}^n$ and internally pass q to $\mathcal{A}''$ as if S did. Compute $Q \subset \{1, 2, \ldots, 2n\}$ as in $\mathsf{Comp}(\Pi)$. Check that for all $i \in Q$, S_i is not corrupted. Otherwise, $\mathcal{S}$ simulates an abort from S.

PHASE IV. Compute $j^* \notin Q$ where S_{j^*} is not corrupted; output $\mathsf{failure}$ if such a j^* does not exist. When $\mathcal{A}''$ sends $\{\alpha_j\}_{j \notin Q}$ as R, compute $r = \alpha_{j^*} \oplus r_{j^*}$ and externally send

$(\mathsf{receiver}, \mathit{sid}, r)$ to $\mathcal{F}_{\text{OT}}$. Upon receiving $(\mathsf{output}, \mathit{sid}, s_r)$ externally from $\mathcal{F}_{\text{OT}}$, compute (σ_0, σ_1) so that σ_r is consistent with s_r as follows:

- If $r = 0$, then $\sigma_0 = s_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and σ_1 is a random string in $\{0,1\}^\ell$.
- If $r = 1$, then σ_0 is a random string in $\{0,1\}^\ell$ and $\sigma_1 = s_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$.

$\mathcal{S}$ then sends (σ_0, σ_1) to $\mathcal{A}''$ as if sent by S to R.

Dealing with corruptions: Corruptions of $\mathsf{R}_1, \ldots, \mathsf{R}_{2n}, \mathsf{S}_1, \ldots, \mathsf{S}_{2n}$ may be handled as above. For corruptions of R and S, we will consider two cases depending on the corruption schedule. In the first case, at least one of the parties is corrupted before the message (σ_0, σ_1) is sent.

- Once S is corrupted, $\mathcal{S}$ learns the actual input (s_0, s_1). If S is corrupted before the messages (σ_0, σ_1) are computed, then $\mathcal{S}$ may simply present $(s_1^0, s_1^1), \ldots, (s_{2n}^0, s_{2n}^1)$ (as chosen in Phase I) as the randomness of S. Otherwise, $\mathcal{S}$ modifies $s_{j^*}^{1-r_{j^*}}$ (if necessary) so that both relations $\sigma_0 = s_0 \oplus (\bigoplus_{j \notin Q} s_j^{\alpha_j})$ and $\sigma_1 = s_1 \oplus (\bigoplus_{j \notin Q} s_j^{1-\alpha_j})$ are satisfied.
- Once R is corrupted, $\mathcal{S}$ learns the actual input r. If R is corrupted before the messages $\{\alpha_j\}_{j \notin Q}$ are computed, then $\mathcal{S}$ may simply present $(r_1, \ldots, r_{2n})$ (as chosen in Phase I) as the randomness of R. Otherwise, $\mathcal{S}$ modifies $\{r_j\}_{j \notin Q}$ so that $r_j = r \oplus \alpha_j$. In addition, $\mathcal{S}$ presents $s_i^{r_i}$ as the output of $\mathsf{R}_i, i = 1, 2, \ldots, 2n$.

In the other case, neither party is corrupted when the message (σ_0, σ_1) is sent.

- Once S is corrupted, we will modify both $s_{j^*}^0$ and $s_{j^*}^1$ so that (σ_0, σ_1) is consistent with (s_0, s_1).
- Once R is corrupted, we will first modify $\{r_j\}_{j \notin Q}$ as in the previous case and then modify $s_{j^*}^{r_{j^*}}$ so that σ_r is consistent with s_r.

We claim that if $\mathcal{S}$ does not output failure, then the ensembles $\text{HYBRID}_{\mathcal{F}_{\text{OT}}, \mathcal{A}'', \mathcal{Z}}$ and $\text{IDEAL}_{\mathcal{F}_{\text{OT}}, \mathcal{S}, \mathcal{Z}}$ are identical. This is clear up to the end of Phase III. For Phase IV, observe that if S and S_{j^*} are not corrupted, then from the view of $\mathcal{A}''$ and $\mathcal{Z}$ in $\text{HYBRID}_{\mathcal{F}_{\text{OT}}, \mathcal{A}'', \mathcal{Z}}$, the string $s_{j^*}^{1-r_{j^*}}$ is truly random. As such, σ_{1-r} is also truly random. Similarly, if R is not corrupted, then from the view of $\mathcal{A}''$ and $\mathcal{Z}$, the n values $\{r_j\}_{j \notin Q}$ are truly random and thus $\{\alpha_j\}_{j \notin Q}$ are also truly random. Furthermore, if

neither S nor R is corrupted just before the message (σ_0, σ_1) is sent, then from the view of $\mathcal{A}''$ and $\mathcal{Z}$, both $s^0_{j^*}$ and $s^1_{j^*}$ are truly random, and thus both σ_0 and σ_1 are truly random.

It remains to show that $\mathcal{S}$ outputs $\mathsf{failure}$ with negligible probability. Observe that $\mathcal{S}$ only outputs $\mathsf{failure}$ if at the start of Phase IV, all of the following conditions hold:

- Neither party has aborted. In addition, the sender is honest at the start of Phase IV, so the challenge q is chosen at random.
- Amongst the n pairs of parties $(\mathsf{S}_1, \mathsf{S}_2), \ldots, (\mathsf{S}_{2n-1}, \mathsf{S}_{2n})$, exactly one party in each pair is corrupted. Otherwise, if there is a pair where both parties are corrupted, then S will abort at the end of Phase III; and if there is a pair where neither party is corrupted, then there is an uncorrupted S_{j^*}.
- The set Q corresponding to the challenge q is exactly the set of n uncorrupted parties (one in each pair).

Clearly, the last condition only holds with probability 2^{-n} over a random choice of q. ■

5.4 Malicious sender and semi-honest receiver

In this section, we reverse the OT protocol from the previous section to obtain one that is secure for a malicious sender and a semi-honest receiver. The construction (shown in Fig 5.4) is the same as that in [Wolf and Wullschleger, 2006], the novelty lies in the analysis which establishes security against an adaptive adversary. We note that the analysis though tedious, is fairly straight-forward.

Proposition 5.4.1 *For every adaptive adversary $\mathcal{A}$ that interacts with the protocol ψ in the $\mathcal{F}_{\text{OT}}$-hybrid model, there exists an adaptive adversary $\mathcal{S}$ that interacts with $\mathcal{F}_{\text{OT}}$, such that for every environment $\mathcal{Z}$,*

$$\text{EXEC}^{\mathcal{F}_{\text{OT}}}_{\psi,\mathcal{A},\mathcal{Z}} \equiv \text{IDEAL}_{\mathcal{F}_{\text{OT}},\mathcal{S},\mathcal{Z}}.$$

Moreover, the corruption pattern in $\mathcal{S}$ is the reverse of that in $\mathcal{A}$.

INITIALIZATION.

Sender input: $(\mathsf{sender}, sid, s_0, s_1)$ where $s_0, s_1 \in \{0,1\}$ and $sid = (\mathsf{S}, \mathsf{R}, sid')$.

Receiver input: $(\mathsf{receiver}, sid, r)$ where $r \in \{0,1\}$.

PHASE I: CALL $\mathcal{F}_{\text{OT}}$.

1. S sends $(\mathsf{receiver}, sid_1, s_0 \oplus s_1)$ to
2. Upon receiving $(\mathsf{receiver}, sid_1)$ from $\mathcal{F}_{\text{OT}}$, R chooses a bit $\rho \in_R \{0,1\}$ and sends $(\mathsf{sender}, sid_1, \rho, \rho \oplus r)$ to $\mathcal{F}_{\text{OT}}$.

PHASE II: REVERSE.

1. Upon receiving $(\mathsf{output}, sid_1, a)$ from $\mathcal{F}_{\text{OT}}$, S computes $\alpha = s_0 \oplus a$ and sends α to R.
2. Upon receiving α, R computes and outputs $\rho \oplus \alpha$.

Figure 5.4: THE OT-REVERSAL PROTOCOL ψ

The statement of the lemma is intended for protocols ψ that are secure against semi-honest, adaptive adversaries without erasures and adaptive adversaries $\mathcal{A}$ and $\mathcal{S}$. However, we note that the lemma also holds for protocols ψ that are secure against semi-honest, adaptive adversaries *with* erasures and semi-honest, adaptive adversaries $\mathcal{A}$ and $\mathcal{S}$ *with* erasures as well as protocols ψ that are secure against semi-honest, static adversaries and static adversaries $\mathcal{A}$ and $\mathcal{S}$.

Proof: As usual, $\mathcal{S}$ works by invoking a copy of $\mathcal{A}$ and running a simulated interaction of $\mathcal{A}$ with $\mathcal{Z}$ and the parties S and R in the $\mathcal{F}_{\text{OT}}$-hybrid model. We will refer to the communication of $\mathcal{S}$ with $\mathcal{Z}$ and ψ as external communication, and that with the simulated $\mathcal{A}$ as internal communication. In addition, we will refer to the $\mathcal{F}_{\text{OT}}$ functionality in the real execution as the internal $\mathcal{F}_{\text{OT}}$, and that in the ideal execution as the external $\mathcal{F}_{\text{OT}}$. $\mathcal{S}$ works as follows:

Simulating the communication with $\mathcal{Z}$: Every input value that $\mathcal{S}$ externally receives from $\mathcal{Z}$ is written into the adversary $\mathcal{A}$'s input tape (as if coming from $\mathcal{A}$'s environment). Every output value written by $\mathcal{A}$ on its output tape is copied to $\mathcal{S}$'s own output tape (to be read by the external $\mathcal{Z}$).

Simulating the case when neither party is corrupted: Internally pass $(\mathsf{receiver}, sid_1)$ to $\mathcal{A}$ as if $\mathcal{F}_{\text{OT}}$ did. Wait until $(\mathsf{receiver}, sid)$ comes externally from $\mathcal{F}_{\text{OT}}$, and then internally pass

$(\mathsf{output}, \mathit{sid}_1)$ to $\mathcal{A}$ as if $\mathcal{F}_{\text{OT}}$ did. Wait until $(\mathsf{output}, \mathit{sid})$ comes externally from $\mathcal{F}_{\text{OT}}$, and then choose $\alpha \in \{0,1\}$ at random and internally pass α to $\mathcal{A}$ as if S sent it to R.

Simulating the case when only the sender is corrupted: When $\mathcal{A}$ as S sends $(\mathsf{receiver}, \mathit{sid}_1, d)$ to $\mathcal{F}_{\text{OT}}$, $\mathcal{S}$ first waits until $(\mathsf{receiver}, \mathit{sid})$ comes externally from $\mathcal{F}_{\text{OT}}$, and then $\mathcal{S}$ chooses $a \in \{0,1\}$ at random and internally passes $(\mathsf{output}, \mathit{sid}_1, a)$ to $\mathcal{A}$ as if $\mathcal{F}_{\text{OT}}$ did.

When $\mathcal{A}$ as S sends α, $\mathcal{S}$ externally sends $(\mathsf{sender}, \mathit{sid}, a \oplus \alpha, a \oplus \alpha \oplus d)$ to $\mathcal{F}_{\text{OT}}$.

Simulating the case when only the receiver is corrupted: Internally pass $(\mathsf{receiver}, \mathit{sid}_1)$ to $\mathcal{A}$ as if $\mathcal{F}_{\text{OT}}$ did. When $\mathcal{A}$ as R sends $(\mathsf{sender}, \mathit{sid}_1, a_0, a_1)$ to $\mathcal{F}_{\text{OT}}$, $\mathcal{S}$ sets $\rho = a_0$, $r = a_0 \oplus a_1$ and externally sends $(\mathsf{receiver}, \mathit{sid}, r)$ to $\mathcal{F}_{\text{OT}}$. $\mathcal{S}$ also internally passes $(\mathsf{output}, \mathit{sid}_1)$ to $\mathcal{A}$ as if $\mathcal{F}_{\text{OT}}$ did.

Upon receiving $(\mathsf{output}, \mathit{sid}, s_r)$ externally from $\mathcal{F}_{\text{OT}}$, $\mathcal{S}$ internally passes $\alpha = s_r \oplus \rho$ to $\mathcal{A}$ as if S sent it.

Dealing with corruptions: When R is corrupted, $\mathcal{S}$ needs to present $\mathcal{A}$ with a consistent random tape comprising of a single bit ρ. When S is corrupted, $\mathcal{S}$ needs to present $\mathcal{A}$ with the output bit a which S receives from the internal $\mathcal{F}_{\text{OT}}$. We consider four cases depending on the corruption schedule:

- Case 1: R is corrupted before it sends its input to the internal $\mathcal{F}_{\text{OT}}$. In this case, S proceeds as in the case when only the receiver is corrupted to compute ρ and r. If and when S is corrupted, $\mathcal{S}$ computes $a = \rho \oplus rd$ where d is S's input to the internal $\mathcal{F}_{\text{OT}}$ (set to $s_0 \oplus s_1$ if S is honest when it submits its input to the internal $\mathcal{F}_{\text{OT}}$).
- Case 2: Neither party is corrupted when α is sent. In this case, $\mathcal{S}$ picks a random $\alpha \in \{0,1\}$. Then, when R is corrupted, $\mathcal{S}$ learns both its input r and its output s_r, and computes $\rho = \alpha \oplus s_r$. When S is corrupted, $\mathcal{S}$ learns its input s_0, s_1 and computes $a = \alpha \oplus s_0$.

If neither Case 1 nor Case 2 holds, then the adversary $\mathcal{A}$ corrupts either R or S (or both) and learns at least one of ρ and a before seeing α.

- Case 3: $\mathcal{A}$ learns a first. This means $\mathcal{A}$ corrupts S first and corrupts R (if at all) after S receives a from the internal $\mathcal{F}_{\text{OT}}$. Then, $\mathcal{S}$ proceeds as in the case where only the sender is corrupted and picks a random $a \in \{0,1\}$. When R is corrupted, $\mathcal{S}$ learns r and computes $\rho = a \oplus rd$ (where d is again S's input to the internal $\mathcal{F}_{\text{OT}}$).
- Case 4: $\mathcal{A}$ learns ρ first. This means either $\mathcal{A}$ corrupts R first, or $\mathcal{A}$ corrupts R before S receives a from the internal $\mathcal{F}_{\text{OT}}$.[9] In this case, $\mathcal{S}$ picks $\rho \in \{0,1\}$ at random when R is corrupted, and subsequently (if and when $\mathcal{A}$ corrupts S) computes $a = \rho \oplus rd$.

Finally, we need to check that $\text{EXEC}^{\mathcal{F}_{\text{OT}}}_{\psi,\mathcal{A},\mathcal{Z}} \equiv \text{IDEAL}_{\mathcal{F}_{\text{OT}},\mathcal{S},\mathcal{Z}}$, which is similar to that in [Wolf and Wullschleger, 2006] which addresses static corruptions. ■

5.5 Security against Static Adversaries

In this section we prove Theorem 5.2.1 by modifying our analysis for the case where we start with a protocol Π secure against semi-honest, static adversaries and end with a protocol $\mathsf{Comp}(\Pi)$ secure against malicious, static adversaries. In order for the analysis to go through, we must ensure that when dealing with a static adversary, the corruption pattern in the underlying components of $\mathcal{F}_{\text{COM}}$ and Π is always static. We note that it is, in fact, the case that the corruption pattern for $\mathcal{F}_{\text{COM}}$ and the real adversaries is always static. However, it turns out that even when dealing with a static adversary, the corruption pattern for "virtual" parties executing Π may be adaptive. Thus, the analysis for the static case does not immediately follow from the adaptive case. Our key insight for proving the static case, inspired by [Ishai *et al.*, 2008], is that in order to simulate the adaptive corruption of "virtual parties," it suffices for Π to be secure against adaptive adversaries *with* erasures. More specifically, we prove the following theorem, for the case of adaptive adversaries with erasures:

Theorem 5.5.1 *There exists a black-box construction of a protocol that UC realizes OT against a malicious, adaptive adversary* with *erasures in the $\mathcal{F}_{\text{COM}}$-hybrid model, starting from any protocol that UC realizes OT against a semi-honest, adaptive adversary* with *erasures. Moreover, the construction achieves a constant multiplicative blow-up in the number of rounds.*

[9] In particular, it could be that $\mathcal{A}$ corrupts S at the start of the protocol (learning nothing at this point), and then corrupts R immediately after it sends its input to the internal $\mathcal{F}_{\text{OT}}$.

PHASE III: CUT-AND-CHOOSE.

1. R chooses a random $q_{\mathsf{R}} \in \{0,1\}^n$ and sends $(\mathsf{commit}, sid_{2n+1}, q_{\mathsf{R}})$ to $\mathcal{F}_{\mathrm{COM}}$.
2. Upon receiving $(\mathsf{receipt}, sid_{2n+1})$ from $\mathcal{F}_{\mathrm{COM}}$, S sends a random $q_{\mathsf{S}} \in \{0,1\}^n$.
3. R sets $q = q_{\mathsf{R}} \oplus q_{\mathsf{S}}$. The string $q = (q_1, \ldots, q_n) \in \{0,1\}^n$ is used to define a set of indices $Q \subset \{1, 2, \ldots, 2n\}$ of size n in the following way: $Q = \{2i - q_i\}_{i=1}^n$. R sends $(\mathsf{reveal}, sid_{2n+1})$ to $\mathcal{F}_{\mathrm{COM}}$ so that S may also compute Q.
4. For every $i \in Q$, R sends (reveal, sid_i) to $\mathcal{F}_{\mathrm{COM}}$ and upon receiving $(\mathsf{reveal}, sid_i, r_i^{\mathsf{R}}, \tau_i^{\mathsf{R}})$ from $\mathcal{F}_{\mathrm{COM}}$, S computes (r_i, τ_i).
5. S checks that for all $i \in Q$, (r_i, τ_i) is consistent with R's messages in the i'th execution of Π. If not, S aborts and halts.

Figure 5.5: PHASE III OF THE ADAPTIVE WITH ERASURES COMPILER $\mathsf{Comp}(\Pi)$

Next, recall the following theorem from [Lindell, 2009]:

Lemma 5.5.2 ([Lindell, 2009]) *There exists a black-box construction of a $t + O(1)$-round protocol that UC realizes OT against a semi-honest, adaptive adversary with erasures, starting from any t-round protocol that UC realizes OT against a semi-honest, static adversary.*

By combining the compiler of [Lindell, 2009] with our compiler for adaptively secure protocols with erasures in Theorem 5.5.1, we obtain a compiler for statically secure protocols which concludes the proof of Theorem 5.2.1.

Towards proving Theorem 5.5.1, we modify our original compiler $\mathsf{Comp}(\Pi)$ for the setting of adaptive corruptions with erasures. The modified compiler differs from the adaptive compiler only in Phase III, where the string q (which determines the set Q) is chosen via a coin-tossing protocol in the static compiler and is chosen by S alone in the adaptive compiler (see Figure 5.5). As we discuss in more detail below, this modification is necessary in order to simulate the case of a corrupt sender and honest receiver without adaptively corrupting the virtual parties R_i in Phase III.

To prove Theorem 5.5.1 it is sufficient to reprove the analogues of Lemmas 5.3.2, 5.3.3, 5.3.4 and Proposition 5.4.1 for the case where we begin and end with protocols secure against adaptive adversaries with erasures. Now, note that the proofs of Lemmas 5.3.3, 5.3.4 and Proposition 5.4.1 proceed identically when beginning with an oblivious transfer protocol Π secure against semi-honest, adaptive adversaries with erasures and concluding with a protocol $\mathsf{Comp}(\Pi)$ secure under against malcious, adaptive adversaries with erasures.

In contrast, in the proof of Lemma 5.3.2, we do rely on the fact that Π is secure even under adaptive corruptions without erasures. Thus, we reprove Lemma 5.3.2 below for the case of adaptive corruptions *with* erasures.

Lemma 5.5.3 *Let Π be an OT-protocol secure against semi-honest adversaries and adaptive corruptions with erasures. For every adversary $\mathcal{A}$ that interacts with $\mathsf{Comp}(\Pi)$ in the $\mathcal{F}_{\text{COM}}$-hybrid model, there exists a well-formed adversary $\mathcal{A}'$ that interacts in the hybrid execution running Π, such that for every environment $\mathcal{Z}$,*

$$\text{EXEC}^{\mathcal{F}_{\text{COM}}}_{\mathsf{Comp}(\Pi),\mathcal{A},\mathcal{Z}} \equiv \text{HYBRID}_{\Pi,\mathcal{A}',\mathcal{Z}}$$

Proof: As in the proof of Lemma 5.3.2, $\mathcal{A}'$ works by invoking a copy of $\mathcal{A}$ and running a simulated interaction of $\mathcal{A}$ with $\mathcal{Z}$ and the parties S and R. We will refer to the communication of $\mathcal{A}'$ with $\mathcal{Z}$ and $\mathsf{Comp}(\Pi)$ as external communication, and that with the simulated $\mathcal{A}$ as internal communication. More precisely, $\mathcal{A}'$ works as follows:

Simulating the communication with $\mathcal{Z}$: This is handled as in the proof of Lemma 5.3.2.

Simulating the case when neither party is corrupted: This is handled as in the proof of Lemma 5.3.2 except $\mathcal{A}'$ chooses the set Q and corrupts all parties R_i for $i \in Q$ *before* Phase II begins. During Phase II, $\mathcal{A}'$ computes the next messages on behalf of R_i for $i \in Q$ according to the code of the honest receiver and forwards to $\mathcal{A}$.

Simulating the case when only the sender is corrupted: Unlike in the proof of Lemma 5.3.2, $\mathcal{A}'$ will pick q in advance (before Phase II begins) and will later fix the outcome of the coin-toss in Phase III to q. In Phase II, for sessions i in Q, $\mathcal{A}'$ computes the next messages according to the code of the honest receiver and forwards to $\mathcal{A}$. For sessions i not in Q, $\mathcal{A}'$ forwards the messages of the virtual party R_i to $\mathcal{A}$.

PHASE I. This is handled as in the proof of Lemma 5.3.2.

PHASE II. $\mathcal{A}'$ chooses $q \in \{0,1\}^n$ (and the corresponding set Q) uniformly at random. For each $i \in Q$, $\mathcal{A}'$ corrupts R_i before the execution of Π commences. Next, for each round

in the protocol Π, whenever $\mathcal{A}$ expects a message $(\beta_1, \ldots, \beta_{2n})$ from R, then for each $i = 1, \ldots, 2n$, if $i \notin Q$, $\mathcal{A}'$ obtains β_i from R_i for the corresponding round. Otherwise, $\mathcal{A}'$ computes β_i according to R_i's input and randomness and the previous messages from S_i. Finally, $\mathcal{A}'$ internally passes $\mathcal{A}$ the message $(\beta_1, \ldots, \beta_{2n})$, as if sent from R to S. The sender's turn is handled as in the proof of Lemma 5.3.2.

PHASE III. $\mathcal{A}'$ internally passes $\mathcal{A}$ the message $(\mathsf{receipt}, sid_{2n+1})$ as if sent from F_{COM} to S. Then, $\mathcal{A}$ chooses $q_{\mathsf{S}} \in \{0,1\}^n$ and $\mathcal{A}'$ simulates S sending R this string. Next, $\mathcal{A}'$ internally passes $\mathcal{A}$ the message $(\mathsf{reveal}, sid_{2n+1}, q_{\mathsf{R}})$ where q_{R} is chosen such that $q_{\mathsf{R}} = q \oplus q_{\mathsf{S}}$. For each $i \in Q$, $\mathcal{A}'$ computes $r_i^{\mathsf{R}} = r_i \oplus r_i^{\mathsf{S}}$ and $\tau_i^{\mathsf{R}} = \tau_i \oplus \tau_i^{\mathsf{S}}$. $\mathcal{A}'$ sends $(\mathsf{reveal}, sid_i, r_i^{\mathsf{R}}, \tau_i^{\mathsf{R}})$ to $\mathcal{A}$ as if coming from $\mathcal{F}_{\text{COM}}$.

PHASE IV. This is handled as in the proof of Lemma 5.3.2.

Simulating the case when only the receiver is corrupted: We note that it is in simulating this case where adaptive corruptions of virtual parties become necessary. We will need to adaptively corrupt a virtual party S_i during Phase II in the case that virtual party R_i begins to deviate from semi-honest behavior. In the proof of Lemma 5.3.2, $\mathcal{A}'$ corrupts S_i, learns its input and randomness, and continues playing the role of sender honestly with party R_i using the revealed input and randomness. When dealing with adaptive corruptions *with* erasures, $\mathcal{A}'$ may not learn S_i's entire random tape upon corruption. However, we note that in this case adaptive corruptions *with* erasures are sufficient: In the adaptive corruptions with erasures model, upon corruption of S_i, the adversary learns the internal state, which includes the sender's input, and by correctness, sufficient information to complete the execution of the protocol.

PHASE I. This is handled as in the proof of Lemma 5.3.2.

PHASE II. This simulation proceeds identically to the proof of Lemma 5.3.2 except for the following case: Whenever $\mathcal{A}$ sends a message $(\beta_1, \ldots, \beta_{2n})$ from R where β_i is the message in the ith parallel execution of Π, if R_i has not aborted and β_i is not consistent with (r_i, τ_i), $\mathcal{A}'$ tells R_i to abort. In addition, $\mathcal{A}'$ corrupts S_i. Since Π is only adaptively secure *with* erasures, we do not obtain its input (s_i^0, s_i^1) and its entire randomness upon

corruption, but only its input (s_i^0, s_i^1) and a partial view. However, we note that this partial view is sufficient for the simulation to proceed as in the proof of Lemma 5.3.2.

PHASE III. This is handled as in the proof of Lemma 5.3.2.

PHASE IV. This is handled as in the proof of Lemma 5.3.2.

Dealing with corruption of parties: This is handled as in the proof of Lemma 5.3.2 except we assume honest parties may erase data. Thus, upon corruption of parties the adversary learns the internal state, which includes the party's input, and by correctness, sufficient information to continue the execution of the protocol.

■

Part III

Conclusions

Chapter 6

Conclusions

In this thesis we resolve several questions on black-box complexity and adaptive, universal composibility of cryptogarphic tasks. Below we describe future research directions and open problems relating to our work.

Black-Box Complexity: In Chapter 2 we show a black-box construction of a non-malleable encryption scheme from any semantically-secure encryption scheme. A remaining open question is to determine the black-box complexity of encryption schemes that guarantee a stronger form of security, security under *adaptive chosen-ciphertext (CCA2)* attacks. For instance, the relationship between CCA2-secure encryption and semantically-secure encryption is still unresolved. [Gertner *et al.*, 2007] consider a restricted type of black-box reduction, called a "shielding reduction" and prove that no shielding reduction from CCA2-secure encryption to semantically-secure encryption exists. However, the question of whether an unrestricted black-box (or even non-black-box) reduction from CCA2-secure encryption to semantically-secure encryption exists remains open. Another related question is to determine the relationship between CCA2-secure encryption and enhanced trapdoor permutations (eTDP). In their seminal work, [Dolev *et al.*, 2000] presented a non-black-box construction of CCA2-secure encryption from eTDP, but over a decade later no black-box construction is known. A major open problem is to either show a black-box reduction of CCA2-secure encryption to eTDP or prove that such a reduction does not exist.

In Chapter 3 we prove that there is no black-box reduction from optimally-fair coin-tossing

with "few" rounds to one-way function (OWF). Our work leaves open the question of whether there exists a black-box reduction of optimally-fair coin-tossing with polynomial number of rounds to one-way function (OWF). Resolving this question either positively or negatively is a compelling open problem.

A high-level open problem relating to black-box complexity is determining the relative strength of fully black-box reductions (that access both the underlying primitive and the adversary only in an input-output manner) versus reductions that access the adversary in a black-box manner, but access the underlying primitive in a non-black-box manner. [Brakerski *et al.*, 2011] and [Pass *et al.*, 2011] both raised similar questions in their recent works and give partial answers by showing that in some limited cases, access to the code of the underlying primitive does not increase the power of the reduction.

Universal Composability and Adaptive Security: Although our constructions of UC and adaptively-secure MPC in Chapters 4 and 5 are relatively efficient, they are still not efficient enough to be used in practice. Constructing highly efficient protocols for UC oblivious transfer (OT) secure against malicious, adaptive adversaries, based on either general assumptions or on specific hardness assumptions, is an important open problem.

The protocol presented in Chapter 5 relies on black-box access to an Ideal Commitment functionality $\mathcal{F}_{\text{COM}}$. Unlike in the standard model where commitments are relatively simple constructs, the adaptive UC Ideal Commitment functionality is a "complete" functionality (in the sense of [Maji *et al.*, 2010]). Thus, in order to realize the adaptive UC Ideal Commitment functionality, we require set up assumptions such as a common reference string (CRS), or a public-key infrastruction (PKI). An interesting parallel line of research is to determine the minimal set-up assumptions necessary for realizing the Ideal Commitment functionality (or other complete functionalities) in the adaptive, UC model.

Part IV

Appendices

Appendix A

Appendix for Coin Tossing

A.1 The Attack of [Cleve and Impagliazzo, 1993]

Cleve and Impagliazzo [Cleve and Impagliazzo, 1993] showed that any coin-tossing protocol in the fail-stop model with respect to computationally unbounded adversaries, has bias at least $\Omega(1/\sqrt{r})$. Since the details will be necessary for our analysis, we describe the [Cleve and Impagliazzo, 1993] result in greater detail in the following. Let $\Pi = (\mathsf{A}, \mathsf{B})$ be a coin-tossing protocol and let R_{A} and R_{B} denote the random tapes used by A and B. Let $W = [w_1, w_2, \ldots, w_{2r}]$ denote the messages that are sent when (A, B) is executed (at round i, A sends w_{2i-1} to B and B sends w_{2i} to A). Let $\Pi = \Pi(R_{\mathsf{A}}, R_{\mathsf{B}})$ be the output of (A, B) when run to completion. Define $\Pi_1, \Pi_2, \ldots, \Pi_{2r}$ as follows. For $i \in \{1, 2, \ldots, r\}$, Π_{2i-1} is the output of B when A quits right before sending its message in round i, and Π_{2i} is the output of A when B quits right before sending its message in round i. Note that all these quantities are functions of $(R_{\mathsf{A}}, R_{\mathsf{B}})$.

Define the random variables $P_0, P_1, \ldots, P_{2r}$ as follows: For $j \in \{0, 1, \ldots, 2r\}$,

$$P_j(R_{\mathsf{A}}, R_{\mathsf{B}}) = E_{(R'_{\mathsf{A}}, R'_{\mathsf{B}})}[\Pi(R'_{\mathsf{A}}, R'_{\mathsf{B}}) \mid W_j].$$

where $W_j = [w_1, \ldots, w_j]$ are the first j messages. Intuitively, P_j is the expected final value of the protocol Π conditioned on the values of the first j messages exchanged by the parties. We stress that $P_j(R_{\mathsf{A}}, R_{\mathsf{B}}) = P_j(R'_{\mathsf{A}}, R'_{\mathsf{B}})$ for all pairs $(R_{\mathsf{A}}, R_{\mathsf{B}}), (R'_{\mathsf{A}}, R'_{\mathsf{B}})$ that define the same first j messages.

In [Cleve and Impagliazzo, 1993] it is proved that with probability greater than $1/5$, there exists a $j \in \{1, \ldots, 2r\}$, such that $|P_j - P_{j-1}| > \frac{1}{32\sqrt{2r}}$. Define the random variables $Q_1, \ldots, Q_{2r}$ as follows:

$$Q_j(R_{\mathsf{A}}, R_{\mathsf{B}}) = E_{R'_{\mathsf{A}}, R'_{\mathsf{B}}}[\Pi_j(R'_{\mathsf{A}}, R'_{\mathsf{B}}) \mid W_{j-1}].$$

where $W_{j-1} = [w_1, \ldots, w_{j-1}]$ are the first $j-1$ messages. Intuitively, Q_j is the expected value output by the other party if one party quits right after w_{j-1} was sent, conditioned on the values of the first $j-1$ messages exchanged by the parties. As with P_j, we have that $Q_j(R_{\mathsf{A}}, R_{\mathsf{B}}) = Q_j(R'_{\mathsf{A}}, R'_{\mathsf{B}})$ for all pairs $(R_{\mathsf{A}}, R_{\mathsf{B}}), (R'_{\mathsf{A}}, R'_{\mathsf{B}})$ that define the same first $j-1$ messages. We alternatively define Q_j in the following way, which we will use in our analysis:

$$Q_j(R_{\mathsf{A}}, R_{\mathsf{B}}) = E_{R'_{\mathsf{A}}, R'_{\mathsf{B}}}[\Pi_j(R'_{\mathsf{A}}, R'_{\mathsf{B}}) \mid W_j].$$

where $W_j = [w_1, \ldots, w_j]$ are the first j messages. Note that the difference in our alternate definition is that we condition over the j-th message w_j as well as the first $j-1$ messages. Since $\Pi_j(R_{\mathsf{A}}, R_{\mathsf{B}})$ is independent of the j-th message, we have that for all $(R_{\mathsf{A}}, R_{\mathsf{B}})$, $E_{R'_{\mathsf{A}}, R'_{\mathsf{B}}}[\Pi_j(R'_{\mathsf{A}}, R'_{\mathsf{B}}) \mid W_j] = E_{R'_{\mathsf{A}}, R'_{\mathsf{B}}}[\Pi_j(R'_{\mathsf{A}}, R'_{\mathsf{B}}) \mid W_{j-1}]$. Thus, the two definitions are equivalent.

Since, for all $j \in \{1, \ldots, 2r\}, |P_j - Q_j| + |Q_j - P_{j-1}| \geq |P_j - P_{j-1}|$, we may express the event that there exists a $j \in \{1, \ldots, 2r\}$ such that $|P_j - P_{j-1}| > \frac{1}{32\sqrt{2r}}$ as the union of these eight events:

$$G_{\mathsf{A},0} : \exists i \in [r] \text{ s.t. } P_{2i-1} - Q_{2i-1} > \tfrac{1}{64\sqrt{2r}} \qquad H_{\mathsf{A},0} : \exists i \in [r] \text{ s.t. } P_{2i-2} - Q_{2i-1} > \tfrac{1}{64\sqrt{2r}}$$

$$G_{\mathsf{A},1} : \exists i \in [r] \text{ s.t. } P_{2i-1} - Q_{2i-1} < -\tfrac{1}{64\sqrt{2r}} \qquad H_{\mathsf{A},1} : \exists i \in [r] \text{ s.t. } P_{2i-2} - Q_{2i-1} < -\tfrac{1}{64\sqrt{2r}}$$

$$G_{\mathsf{B},0} : \exists i \in [r] \text{ s.t. } P_{2i} - Q_{2i} > \tfrac{1}{64\sqrt{2r}} \qquad H_{\mathsf{B},0} : \exists i \in [r] \text{ s.t. } P_{2i-1} - Q_{2i} > \tfrac{1}{64\sqrt{2r}}$$

$$G_{\mathsf{B},1} : \exists i \in [r] \text{ s.t. } P_{2i} - Q_{2i} < -\tfrac{1}{64\sqrt{2r}} \qquad H_{\mathsf{B},1} : \exists i \in [r] \text{ s.t. } P_{2i-1} - Q_{2i} < -\tfrac{1}{64\sqrt{2r}}$$

Since $\Pr[G_{\mathsf{A},0} \vee G_{\mathsf{A},1} \vee G_{\mathsf{B},0} \vee G_{\mathsf{B},1} \vee H_{\mathsf{A},0} \vee H_{\mathsf{A},1} \vee H_{\mathsf{B},0} \vee H_{\mathsf{B},1}] > \frac{1}{5}$, at least one of these eight events must occur with probability greater than $\frac{1}{40}$. Suppose $\Pr[G_{\mathsf{A},1}] > \frac{1}{40}$. Then consider the following adversarial $\widehat{\mathsf{A}}$: simulate A until a round i occurs where $P_{2i-1} - Q_{2i-1} < -\frac{1}{64\sqrt{2r}}$ in

which case quit.

Now, when $(\widehat{\mathsf{A}}, \mathsf{B})$ is executed, it follows that:

$$\Pr_{(R_{\mathsf{A}},R_{\mathsf{B}})}[b = 1 \mid (a|b) \leftarrow \langle \widehat{\mathsf{A}}, \mathsf{B} \rangle] > \frac{1}{2} + \frac{1}{40} \cdot \frac{1}{64\sqrt{2r}} \tag{A.1.1}$$

Intuitively, this holds because with probability $1/40$ one of the necessary events must hold, and in such a case the adversary biases the outcome by $1/64\sqrt{2r}$.

Observe that in order to carry out this "attack", the adversary $\widehat{\mathsf{A}}$ must be able to compute all of the P_i and Q_i probabilities (in order to find a round in which the difference occurs). However, it appears that this computation cannot be carried out efficiently, implying that a ppt $\widehat{\mathsf{A}}$ cannot carry out the attack. (If $\widehat{\mathsf{A}}$ could be implemented efficiently then this would contradict the existence of oblivious transfer because by [Moran *et al.*, 2009] it is possible to limit the bias to $O(1/r)$ under this assumption.)

A.2 The Independence Learner of [Barak and Mahmoody, 2009]

In this section we show how to prove Lemma 3.3.3 which is implicit in [Barak and Mahmoody, 2009] by deducing it from the results explicitly proved there.

We will use the results of Section 4 of [Barak and Mahmoody, 2009]. We first need to clarify the renaming of the parameters. Here we use m to denote the number of oracle queries by Alice and Bob whereas it is denoted by n there (we saved n for the security parameter of the one-way function). What we call as ϵ in this paper is equal to ϵ/n in [Barak and Mahmoody, 2009] which would in fact be equal to ϵ/m in our terms. Namely, ϵ is the input parameter in [Barak and Mahmoody, 2009], but we prefer to take $\epsilon/m = \epsilon$ as the input parameter.

We need to define some events and distributions used in [Barak and Mahmoody, 2009] which matter to us. At any point during the protocol where Eve is done with her learning algorithm for that round, let W be the messages sent so far, and let I be Eve's knowledge about the random oracle. The notation $\mathcal{EXEC}(M, I)$ is used in [Barak and Mahmoody, 2009] to denote what we called here $\mathbf{AB}$ which is the joint distribution over Alice and Bob's views so far conditioned on (M, I). The event $\mathsf{Good}(M, I)$ is defined over $\mathcal{EXEC}(M, I)$ as follows. Let (A, B) be an instance of the space

$\mathcal{EXEC}(M, I)$. Then we say that $\mathsf{Good}(M, I)$ holds over (A, B) iff $Q(\mathsf{A}) \cap Q(\mathsf{B}) \subset Q(I)$; namely Alice and Bob do not have any "private" query in common out of $Q(I)$. The event Fail defined over a complete execution of the system is equal to the event that $Q(\mathsf{A}) \cap Q(\mathsf{B}) \subset Q(I)$ is violated at *any* point during the execution. Note that the event $\neg\mathsf{Good}$ implies that Fail has already happened.

The independence learner given the parameter ϵ can be described in two lines as follows (see Section 3 of [Barak and Mahmoody, 2009]):

As long as there exists a query $q \notin Q(I)$ such that $\Pr_{(\mathsf{A},\mathsf{B})\leftarrow\mathcal{EXEC}(M,I)}[q \in Q(\mathsf{A}) \cup Q(\mathsf{B})] \geq \epsilon$ then ask q from the oracle and add q paired with its answer to I.

Security of the Learner. The distributions $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{B}}$ are the same as the product distributions defined in Lemma 7 in [Barak and Mahmoody, 2009]. It is proved in Lemma 8 of [Barak and Mahmoody, 2009] that the statistical distance between $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ and $\mathcal{GEXEC}(M, I)$ (which is the same as $\mathbf{AB} \mid \mathsf{Good}(M, I)$) is at most $O(\epsilon)$ which is equal to $O(m\epsilon)$ in our terms. But we are not done yet about proving the security property of the learner, because the event $\neg\mathsf{Good}(M, I)$ can still increase the statistical distance between $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ and $\mathbf{AB}$. We will show that with probability at least $1 - O(\sqrt{\epsilon}) = 1 - O(\sqrt{m\epsilon})$ it holds that $\Pr[\mathsf{Good}] \geq 1 - O(\sqrt{m\epsilon})$. But before showing the latter claim, note that the statistical distance between $\mathcal{GEXEC}(M, I)$ and $\widehat{\mathbf{A}} \times \widehat{\mathbf{B}}$ is at most $O(\epsilon) = O(m\epsilon) < 1/2$ (by Lemma 8 of [Barak and Mahmoody, 2009]) and therefore if any query $q \notin Q(I)$ has chance at most ϵ of being in $Q(\mathsf{A}) \cup Q(\mathsf{B})$ for $(\mathsf{A}, \mathsf{B}) \leftarrow \mathcal{GEXEC}(M, I)$, it also has chance at most 2ϵ to be in $Q(\widehat{\mathbf{A}}) \cup Q(\widehat{\mathbf{B}})$ (which can be made ϵ by rescaling ϵ by a factor of 2).

Now we use Lemma 4 of [Barak and Mahmoody, 2009] which proves that $\Pr[\mathsf{Fail}] \leq O(\epsilon) = O(m\epsilon)$. Lemma 9 of [Barak and Mahmoody, 2009] which was originally proved in [Impagliazzo and Rudich, 1989] (as Lemma 6.4) informally speaking proves that elementary average arguments generalize to an online setting as well. Namely, $\Pr[\mathsf{Fail}] \leq O(m\epsilon)$ implies that with probability $1-O(\sqrt{m\epsilon})$ over the execution of the system, it holds at *all* time during the execution of the system that the probability of Fail happening at some point before conditioned on the current (M, I) is at most $O(\sqrt{m\epsilon})$. Note that $\Pr_{\mathcal{EXEC}(M,I)}[\mathsf{Fail}] \geq \Pr_{\mathcal{EXEC}(M,I)}[\neg\mathsf{Good}]$, which finishes the proof of the security property. It is also easy to see that by making the value of ϵ smaller, it only can decrease the probability of Fail and thus improve the security property.

Efficiency of the Learner. In Lemma 11 of [Barak and Mahmoody, 2009] it is proved that up to an event Bad of probability at most $O(\epsilon) = O(m\epsilon)$, the expected number of queries asked by the learning algorithm is at most $O(m^2/\epsilon) = O(m/\epsilon)$. So if we stop the learning after the Bad event happens we might loose the security of the learner by at most $O(m\epsilon) \leq O(\sqrt{m\epsilon})$ which is still fine and yet we preserve the required efficiency. All that is needed in the proof of the efficiency is that whenever Eve asks a query while the event Bad has not happened this query is using the threshold parameter *at most* ϵ. So if we use larger values for ϵ we still preserve the efficiency with respect to parameter ϵ.

Robustness of the Learner. The proof of Robustness was discussed above along proving the Security and the Efficiency properties.

Part V

Bibliography

Bibliography

[Bach, 1988] Eric Bach. How to generate factored random numbers. *SIAM Journal on Computing*, 17(2):179–193, 1988.

[Barak and Mahmoody-Ghidary, 2007] Boaz Barak and Mohammad Mahmoody-Ghidary. Lower bounds on signatures from symmetric primitives. In *Proc. 48th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 680–688, 2007.

[Barak and Mahmoody, 2009] Boaz Barak and Mohammad Mahmoody. Merkle puzzles are optimal - an O(n^2)-query attack on any key exchange from a random oracle. In *Advances in Cryptology – CRYPTO 2009*, pages 374–390, 2009.

[Barak *et al.*, 2004] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *The 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 186–195, 2004.

[Barak, 2001] Boaz Barak. How to go beyond the black-box simulation barrier. In *The 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 106–115, 2001.

[Barak, 2002] Boaz Barak. Constant-round coin-tossing with a man in the middle or realizing the shared random string model. In *The 43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 345–355, 2002.

[Beaver and Haber, 1992] Donald Beaver and Stuart Haber. Cryptographic protocols provably secure against dynamic adversaries. In *Advances in Cryptology – EUROCRYPT 1992*, pages 307–323, 1992.

[Beaver, 1997] Donald Beaver. Plug and play encryption. In *Advances in Cryptology – CRYPTO 1997*, pages 75–89, 1997.

[Beaver, 1998] Donald Beaver. Adaptively secure oblivious transfer. In *Advances in Cryptology – ASIACRYPT 1998*, pages 300–314, 1998.

[Ben-Or *et al.*, 1988] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness Theorems for Noncryptographic Fault-Tolerant Distributed Computations. In *Proc. 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1–10, 1988.

[Blum *et al.*, 1988] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *Proc. 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 103–112, 1988.

[Blum, 1981] Manuel Blum. Coin flipping by telephone. In *Advances in Cryptology – CRYPTO 1981*, pages 11–15, 1981.

[Brakerski *et al.*, 2011] Zvika Brakerski, Jonathan Katz, Gil Segev, and Arkady Yerukhimovich. Limits on the power of zero-knowledge proofs in cryptographic constructions. In *Theory of Cryptography – TCC 2011*, pages 559–578, 2011.

[Canetti and Rabin, 2003] Ran Canetti and Tal Rabin. Universal composition with joint state. In *Advances in Cryptology – CRYPTO 2003*, pages 265–281, 2003.

[Canetti *et al.*, 1996] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *Proc. 28th Annual ACM Symposium on Theory of Computing (STOC)*, pages 639–648, 1996.

[Canetti *et al.*, 2002] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 494–503, 2002.

[Canetti *et al.*, 2003] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *Advances in Cryptology – EUROCRYPT 2003*, pages 68–86, 2003.

[Canetti *et al.*, 2004] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In *Advances in Cryptology – EUROCRYPT 2004*, pages 207–222, 2004.

[Canetti *et al.*, 2005] Ran Canetti, Shai Halevi, and Jonathan Katz. Adaptively-secure, non-interactive public-key encryption. In *Theory of Cryptography – TCC 2005*, pages 150–168, 2005.

[Canetti *et al.*, 2007] Ran Canetti, Yevgeniy Dodis, Rafael Pass, and Shabsi Walfish. Universally composable security with global setup. In *Theory of Cryptography – TCC 2007*, pages 61–85, 2007.

[Canetti, 2000] Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[Canetti, 2001] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *The 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 136–145, 2001.

[Choi *et al.*, 2008] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *Theory of Cryptography – TCC 2008*, pages 427–444, 2008.

[Choi *et al.*, 2009a] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Improved non-committing encryption with applications to adaptively secure protocols. In *Advances in Cryptology – ASIACRYPT 2009*, pages 287–302, 2009.

[Choi *et al.*, 2009b] Seung Geol Choi, Dana Dachman-Soled, Tal Malkin, and Hoeteck Wee. Simple, black-box constructions of adaptively secure protocols. In *Theory of Cryptography – TCC 2009*, pages 387–402, 2009.

[Cleve and Impagliazzo, 1993] Richard Cleve and Russell Impagliazzo. Martingales, collective coin flipping and discrete control processes. Unpublished, 1993.

[Cleve, 1986] Richard Cleve. Limits on the security of coin flips when half the processors are faulty (extended abstract). In *Proc. 18th Annual ACM Symposium on Theory of Computing (STOC)*, pages 364–369, 1986.

[Cramer and Shoup, 1998] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *Advances in Cryptology – CRYPTO 1998*, pages 13–25, 1998.

[Cramer and Shoup, 2004] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Advances in Cryptology – EUROCRYPT 2004*, pages 45–64, 2004.

[Cramer *et al.*, 2007] Ronald Cramer, Goichiro Hanaoka, Dennis Hofheinz, Hideki Imai, Eike Kiltz, Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Bounded CCA2-secure encryption. In *Advances in Cryptology – ASIACRYPT 2007*, pages 502–518, 2007.

[Crépeau *et al.*, 1995] Claude Crépeau, Jeroen van de Graaf, and Alain Tapp. Committed oblivious transfer and private multi-party computation. In *Advances in Cryptology – CRYPTO 1995*, pages 110–123, 1995.

[Dachman-Soled *et al.*, 2009] Dana Dachman-Soled, Tal Malkin, Mariana Raykova, and Moti Yung. Efficient robust private set intersection. In *Applied Cryptography and Network Security, 7th International Conference, ACNS 2009*, pages 125–142, 2009.

[Damgård and Ishai, 2005] Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In *Advances in Cryptology – CRYPTO 2005*, pages 378–394, 2005.

[Damgård and Nielsen, 2000] Ivan Damgård and Jesper Buus Nielsen. Improved non-committing encryption schemes based on a general complexity assumption. In *Advances in Cryptology – CRYPTO 2000*, pages 432–450, 2000.

[De Santis and Persiano, 1992] Alfredo De Santis and Giuseppe Persiano. Zero-knowledge proofs of knowledge without interaction. In *Proc. 33rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 427–436, 1992.

[Dolev *et al.*, 2000] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.

[Dwork *et al.*, 2003] Cynthia Dwork, Moni Naor, Omer Reingold, and Larry J. Stockmeyer. Magic functions. *Journal of the ACM*, 50(6):852–921, 2003.

[Elkind and Sahai, 2002] Edith Elkind and Amit Sahai. A unified methodology for constructing public-key encryption schemes secure against adaptive chosen-ciphertext attack. Cryptology ePrint Archive, Report 2002/024, 2002. `http://eprint.iacr.org/`.

[Feige *et al.*, 1999] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM J. on Comput.*, 29(1):1–28, 1999.

[Fischlin and Fischlin, 2002] Marc Fischlin and Roger Fischlin. The representation problem based on factoring. In *Topics in Cryptology - CT-RSA 2002, The Cryptographer's Track at the RSA Conference*, pages 96–113, 2002.

[Garay *et al.*, 2009] Juan A. Garay, Daniel Wichs, and Hong-Sheng Zhou. Somewhat non-committing encryption and efficient adaptively secure oblivious transfer. In *Advances in Cryptology – CRYPTO 2009*, pages 505–523, 2009.

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

[Gentry *et al.*, 2008] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 197–206, 2008.

[Gertner *et al.*, 2000] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *Proc. 41st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 325–335, 2000.

[Gertner *et al.*, 2001] Yael Gertner, Tal Malkin, and Omer Reingold. On the impossibility of basing trapdoor functions on trapdoor predicates. In *Proc. 42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 126–135, 2001.

[Gertner *et al.*, 2007] Yael Gertner, Tal Malkin, and Steven Myers. Towards a separation of semantic and cca security for public key encryption. In *Theory of Cryptography – TCC 2007*, pages 434–455, 2007.

[Goldreich *et al.*, 1986] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.

[Goldreich *et al.*, 1988] O. Goldreich, S. Micali, and A. Wigderson. How to Play Any Mental Game. In *Proc. 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 218–229, 1988.

[Goldreich, 2004] Oded Goldreich. *Foundations of Cryptography: Volume II, Basic Applications*. Cambridge University Press, 2004.

[Goldwasser and Micali, 1984] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comp. Sys. Sci.*, 28(21):270–299, 1984.

[Gordon and Katz, 2009] S. Dov Gordon and Jonathan Katz. Complete fairness in multi-party computation without an honest majority. In *Theory of Cryptography – TCC 2009*, pages 19–35, 2009.

[Gordon and Katz, 2010] S. Dov Gordon and Jonathan Katz. Partial fairness in secure two-party computation. In *Advances in Cryptology – EUROCRYPT 2010*, pages 157–176, 2010.

[Gordon *et al.*, 2008] S. Dov Gordon, Carmit Hazay, Jonathan Katz, and Yehuda Lindell. Complete fairness in secure two-party computation. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 413–422, 2008.

[Goyal *et al.*, 2010] Vipul Goyal, Yuval Ishai, Mohammad Mahmoody, and Amit Sahai. Interactive locking, zero-knowledge PCPs, and unconditional cryptography. In *Advances in Cryptology – CRYPTO 2010*, pages 173–190, 2010.

[Goyal, 2011] Vipul Goyal. Constant round non-malleable protocols using one way functions. In *Proc. 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 695–704, 2011.

[Granville and Martin, 2004] Andrew Granville and Greg Martin. Prime number races, 2004. http://arxiv.org/abs/math/0408319.

[Haitner and Reingold, 2007] Iftach Haitner and Omer Reingold. A new interactive hashing theorem. In *IEEE Conference on Computational Complexity (CCC)*, 2007.

[Haitner *et al.*, 2007] Iftach Haitner, Jonathan J. Hoch, Omer Reingold, and Gil Segev. Finding collisions in interactive protocols - a tight lower bound on the round complexity of statistically-hiding commitments. In *Proc. 48th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 669–679, 2007.

[Haitner *et al.*, 2008] Iftach Haitner, Jonathan J. Hoch, and Gil Segev. A linear lower bound on the communication complexity of single-server private information retrieval. In *Theory of Cryptography – TCC 2008*, pages 445–464, 2008.

[Haitner *et al.*, 2009] Iftach Haitner, Minh-Huyen Nguyen, Shien Jin Ong, Omer Reingold, and Salil Vadhan. Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. *SIAM Journal on Computing*, 39(3):1153–1218, 2009.

[Haitner *et al.*, 2011] Iftach Haitner, Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions of protocols for secure computation. *SIAM Journal on Computing*, 40(2):225–266, 2011.

[Haitner, 2008] Iftach Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In *Theory of Cryptography – TCC 2008*, pages 412–426, 2008.

[Halpern, 1995] Joseph Y. Halpern. Reasoning about knowledge: A survey. Manuscript, 1995.

[Håstad *et al.*, 1999] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[Impagliazzo and Luby, 1989] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *Proc. 30th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 230–235, 1989.

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

[Ishai *et al.*, 2006] Yuval Ishai, Eyal Kushilevitz, Yehuda Lindell, and Erez Petrank. Black-box constructions for secure computation. In *Proc. 38th Annual ACM Symposium on Theory of Computing (STOC)*, pages 99–108, 2006.

[Ishai *et al.*, 2007] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–30, 2007.

[Ishai *et al.*, 2008] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In *Advances in Cryptology – CRYPTO 2008*, pages 572–591, 2008.

[Jarecki and Lysyanskaya, 2000] Stanislaw Jarecki and Anna Lysyanskaya. Adaptively secure threshold cryptography: Introducing concurrency, removing erasures. In *Advances in Cryptology – EUROCRYPT 2000*, pages 221–242, 2000.

[Kalai, 2002] Adam Kalai. Generating random factored numbers, easily. In *Proc. of the Thirteenth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms — (SODA 2002)*, pages 412–412, 2002.

[Kalai, 2005] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In *Advances in Cryptology – EUROCRYPT 2005*, pages 78–95, 2005.

[Katz and Ostrovsky, 2004] Jonathan Katz and Rafail Ostrovsky. Round-optimal secure two-party computation. In *Advances in Cryptology – CRYPTO 2004*, pages 335–354, 2004.

[Katz, 2007] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In *Advances in Cryptology – EUROCRYPT 2007*, pages 115–128, 2007.

[Kilian, 1988] Joe Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 20–31, 1988.

[König *et al.*, 1999] H. König, C. Schütt, and N. Tomczak-Jaegermann. Projection constants of symmetric spaces and variants of khintchine's inequality. *J. Reine Agnew. Math.*, 511:1–42, 1999.

[Kushilevitz, 1992] Eyal Kushilevitz. Privacy and communication complexity. *SIAM Journal on Discrete Mathematics*, 5(2):273–284, 1992.

[Lamport, 1979] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, 1979.

[Lindell and Pinkas, 2007] Yehuda Lindell and Benny Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *Advances in Cryptology – EUROCRYPT 2007*, pages 52–78, 2007.

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

[Lindell, 2009] Andrew Y. Lindell. Adaptively secure two-party computation with erasures. In *Topics in Cryptology - CT-RSA 2009, The Cryptographer's Track at the RSA Conference*, pages 117–132, 2009.

[Luby and Rackoff, 1988] M. Luby and C. Rackoff. How to construct pseudorandom permutations and pseudorandom functions. *SIAM Journal on Computing*, 17(2), 1988.

[Maji and Prabhakaran, 2010] Hemanta Maji and Manoj Prabhakaran. Personal communication. 2010.

[Maji *et al.*, 2009] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Complexity of multi-party computation problems: The case of 2-party symmetric secure function evaluation. In *Theory of Cryptography – TCC 2009*, pages 256–273, 2009.

[Maji *et al.*, 2010] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A zero-one law for cryptographic complexity with respect to computational uc security. In *Advances in Cryptology – CRYPTO 2010*, pages 595–612, 2010.

[Moran *et al.*, 2009] Tal Moran, Moni Naor, and Gil Segev. An optimally fair coin toss. In *Theory of Cryptography – TCC 2009*, pages 1–18, 2009.

[Myers and Shelat, 2009] Steven Myers and Abhi Shelat. Bit encryption is complete. In *Proc. 50th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 607–616, 2009.

[Naor and Pinkas, 2001] Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *Proceedings of the of the Twelfth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms — (SODA 2001)*, pages 448–457, 2001.

[Naor and Yung, 1989] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 33–43, 1989.

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

[Naor *et al.*, 1998] Naor, Ostrovsky, Venkatesan, and Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *joc*, 11(2):87–108, 1998.

[Naor, 1991] Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[O'Neill *et al.*, 2011] Adam O'Neill, Chris Peikert, and Brent Waters. Bi-deniable public-key encryption. In *To appear in Advances in Cryptology – CRYPTO 2011*, 2011.

[Pass and Wee, 2009] Rafael Pass and Hoeteck Wee. Black-box constructions of two-party protocols from one-way functions. In *Theory of Cryptography – TCC 2009*, pages 403–418, 2009.

[Pass *et al.*, 2006] Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *Advances in Cryptology – CRYPTO 2006*, pages 271–289, 2006.

[Pass *et al.*, 2011] Rafael Pass, Wei-Lung Dustin Tseng, and Muthuramakrishnan Venkitasubramaniam. Towards non-black-box lower bounds in cryptography. In *Theory of Cryptography – TCC 2011*, pages 579–596, 2011.

[Peikert and Waters, 2008] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 187–196, 2008.

[Peikert *et al.*, 2008] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *Advances in Cryptology – CRYPTO 2008*, pages 554–571, 2008.

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

[Reingold *et al.*, 2004] Omer Reingold, Luca Trevisan, and Salil Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of Cryptography – TCC 2004*, pages 1–20, 2004.

[Rompel, 1990] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. 22nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 387–394, 1990.

[Rubinstein and Sarnak, 1994] Michael Rubinstein and Peter Sarnak. Chebyshevs bias. *Experiment. Math*, 3(3):173–197, 1994.

[Sahai, 1999] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *Proc. 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 543–553, 1999.

[Schnorr, 1996] Claus-Peter Schnorr. Security of 2^t-root identification and signatures. In *Advances in Cryptology – CRYPTO 1996*, pages 143–156, 1996.

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

[Wee, 2010] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *Proc. 51st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 531–540, 2010.

[Wolf and Wullschleger, 2006] Stefan Wolf and Jürg Wullschleger. Oblivious transfer is symmetric. In *Advances in Cryptology – EUROCRYPT 2006*, pages 222–232, 2006.

[Yao, 1982] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions. In *Proc. 23rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.