KTH Numerical Analysis
and Computer Science

Alternative Variants of Zero-Knowledge Proofs

RAFAEL PASS

Licentiate Thesis
Stockholm, Sweden 2004

TRITA-NA-0440
ISSN 0348-2952
ISRN KTH/NA/R--04/40--SE
ISBN 91-7283-933-3

KTH Numerisk analys och datalogi
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie licentiatexamen måndagen den 17 jan 2005 i Kollegiesalen, Administrationsbyggnaden, Kungl Tekniska högskolan, Valhallavägen 79, Stockholm.

Tryck: Universitetsservice US AB

Abstract

Zero-knowledge proofs are one of the most important cryptographic notions. Since their introduction in the early 80's by Goldwasser, Micali and Rackoff, they have proven very useful in the design of cryptographic protocols. Nevertheless, many limitations (in terms of efficiency and robustness under concurrent executability of protocols) have also been noticed. In order to overcome these limitations two lines of research have been investigated in the literature:

1. Models with some "limited" intervention of a trusted party (for example during a set-up phase).

2. Weakenings of the notion of zero-knowledge.

In this thesis we attempt to further the understanding of the notion of zero-knowledge proofs by addressing both the above lines of research. More precisely,

1. Concerning the first line of research, we show that the definition of zero-knowledge in certain popular models (namely the Common Reference String and Random Oracle Models) captures a somewhat different semantics than the standard zero-knowledge definition. In particular, we show that there exists a certain *natural* security property that is captured by the standard definition, but not by these new definitions. This is the property of *deniability*, which intuitively means that an interaction between two parties does not leave any "trace". We then focus on investigating the possibility of obtaining *deniable zero-knowledge* protocols in these models.

2. Concerning the second line of research, we propose a new and different weakening of the notion of zero-knowledge proofs. This weakening allows us to a) overcome known impossibility results concerning the notion of zero-knowledge, while b) providing an "almost" as strong security guarantee.

Acknowledgments

I would like to express my deepest gratitude to Johan Håstad. Johan's clear view on all aspects of theoretical computer science, ranging from philosophical considerations to technical issues, have had a great impact on my research and development. Furthermore, Johan has an amazing way of understanding "raw" ideas that I barely can formulate and extracting the essence out of them. Perhaps most importantly, Johan's exquisite intuition for what is "interesting" in research has allowed me to gain a greater understanding of my area of research. It has been a privilege and a pleasure to interact with him!

Other people that have directly influenced the research in this thesis include Boaz Barak, Ran Canetti, Shafi Goldwasser, Gustav Hast, Silvio Micali, Tal Rabin, Alon Rosen, Yael Tauman-Kalai, and Douglas Wikström. Alon Rosen deserves special attention; my close collaborations with him has taught me a lot about research in general, and Cryptography in particular. The general presentation of many of the results in this thesis owes a lot to him.

I would also like to thank the Theory group at KTH, and in particular Lars Engebretsen, Isaac Elias, Mikael Goldmann, Jonas Holmerin, Mårten Trolin, and my room mates Gustav Hast, Anna Redz and Magnus Rosell. Furthermore, the results in the thesis have also benefited from "non-technical" conversations with friends. Thanks to Marcus Better, Crispin Dickson, Ulrik Dahlerus, Daniel Lovas, and Andreas Röberg, for many insightful conversations about Cryptography, and life in general.

Finally, thanks to my parents Julia and Natan, my sisters Esther Lou and Ariella, and my beautiful Sandra, for supporting me and surviving my life in Zero-knowledge.

Contents

Chapter 1

Introduction

Cryptography, Greek for "hidden writing", is the science of how parties can communicate without trusting each other and/or the environment they are communicating in. Traditionally the science of cryptography was primarily used by the military and the secret services in order to achieve secure message transmission, but with the recent explosion of electronic communication, cryptography is today the concern of almost anyone.

The past three decades have witnessed an unprecedented progress in the field of cryptography. In these years basic cryptographic notions such as encryption[1] and digital signatures[2] have been put under rigorous treatment and numerous solutions realizing these task have been proposed. Furthermore, during these years several new and exciting cryptographic notions/applications have emerged.

1.1 Zero-Knowledge Proofs

One of the most basic and important examples of cryptographic notions is the one of *zero-knowledge interactive proof systems* [38]. Loosely speaking, zero-knowledge proofs are interactive protocols[3] that enable one entity (called the *prover*) to convince another entity (called the *verifier*) of the validity of a mathematical statement, without revealing *anything* beyond the assertion of the statement. For concreteness, assume that a prover wants to convince a verifier that a certain equation has a solution. A naive way of doing this would be by simply giving the solution to the verifier. This approach, however, reveals information to the verifier (namely the solution to the equation). A zero-knowledge proof, on the other hand, would convince the verifier without revealing anything else.

This notion is formalized by requiring the existence of a simulator that can efficiently generate a transcript which is "indistinguishable" from the view of the

[1]Encryption allows users to conceal the content of messages from unauthorized users

[2]A digital signature takes the place of a handwritten signature in the electronic world.

[3]In interactive protocols the participating entities take turns in exchanging messages.

verifier in a real execution with the prover. This, in particular, implies that anything a verifier can learn after having communicated with a prover, he could have efficiently generated by himself without the help of the prover.

The notion of zero-knowledge proofs has had an tremendous impact on the field of cryptography, both directly and indirectly. We mention a few examples.

Identification protocols One of the most commonly used cryptographic tools is that of an identification protocol.[4] Identification protocols allow users to "prove" their identity to an authority. This is normally done by letting each user posses a "secret" that is associated with its identity. In the identification process the user is asked to show that it indeed possess the secret using a protocol. It is here important that the protocol does not reveal the actual secret. (If the protocol would reveal the secret then an adversary, that listens in on the conversation, could later falsely identify himself as the user). Zero-knowledge proofs provide a natural solution to this problem.

Security definitions of other notions The notion of zero-knowledge can be used in other contexts than that of interactive proofs. In fact, it formalizes in a natural way a situation when a participant P in a cryptographic protocol is not able to extract any useful information from the execution of the protocol. Practically all security definitions of more complicated cryptographic notions (such as coin-tossing over the phone, voting, electronic auctions, mental poker, etc.) rely on the concept of zero-knowledge.

Constructions of secure protocol A very useful paradigm for constructing cryptographic protocols is to first construct a protocol that is secure against entities that follow the protocol, and thereafter "compiling" this protocol into a new protocol that is secure also against cheating entities that deviate from the protocol in order to extract information [36]. Arguably, the most important application of zero-knowledge proof is in providing such a compilation.

1.2 Concurrent Composition of Cryptographic Protocols

The original setting in which cryptographic protocols were investigated consisted of a single execution of the protocol at a time. That is, two parties execute a single protocol once, without being able to communicate with the outside world. A more realistic setting, especially in the time of the Internet, is one that allows the *concurrent* execution of protocols [27, 25]. In the concurrent setting many protocols are executed at the same time, involving multiple parties that may be talking with the same (or many) other parties simultaneously.

The concurrent setting presents the new risk of a coordinated attack in which an adversary controls many parties, interleaving the executions of the protocols while trying to extract knowledge based on the existence of multiple concurrent

[4] Such protocols are for example used in log-in procedures by Internet banks.

executions. In particular, the adversary may use messages received in one of the executions in order to cheat in a different execution.

It would be most desirable to have cryptographic protocols that retain their security properties even when executed concurrently. Unfortunately, it has been observed that the construction of such protocol seems more problematic than in the stand-alone setting. We start by outlining different types of concurrent composition, and thereafter return to discuss these limitation in Section 1.3.

Self-composition The simplest form of concurrent composition is that of concurrent *self-composition*. Roughly speaking, we say that a protocol is secure under concurrent self-composition (or just concurrent composition) if the protocol remains secure even if many simultaneous executions of that protocol are taking place. Zero-knowledge proofs that have this property are called *concurrent zero-knowledge proofs*.

General Composition A more general form of concurrent composition is that of concurrent *general composition*. Whereas the notion of concurrent self-composition only considers the security of a protocol when concurrently executed with many instances of the *same* protocol, the notion of general composition considers the security of a protocol that is concurrently executed with many other, possibly different, protocols. One important framework for addressing these security demands is the framework for Universal Composability (UC for short) [12]. As the name indicates the goal of the framework is the possibility to devise cryptographic "subroutines" that can be composed in arbitrary ways possibly under concurrent executions. It was recently shown that the notions of Universal Composability and general composability are "essentially" equivalent [48].

1.3 Limitations of zero-knowledge proofs

Although the notion of zero-knowledge proofs has proved very useful in the design of cryptographic protocols, many limitations are also known, both in the stand-alone setting and in the concurrent setting.

The stand-alone setting It was early shown that two communication rounds (i.e., one message sent from the verifier to the prover, followed by one message from the prover to the verifier) are not sufficient in order to implement zero-knowledge proofs [37]. Furthermore it was shown that a certain type of zero-knowledge proofs called *black-box* zero-knowledge proofs[5] necessite four communication rounds. Even though there exist non black-box zero-knowledge protocols (which all use more than

[5]As will be explained in more detail later, the notion of black-box zero-knowledge requires the existence of a simulator which only uses the verifier V as a black-box in order to perform the simulation.

four communication rounds) [1] all currently known *practical* protocols are black-box zero-knowledge.

The concurrent setting In all currently known concurrent zero-knowledge protocols a heavy price has to be paid in terms of the number of communications rounds. Whereas the number of communication rounds in stand-alone secure zero-knowledge protocols is *constant* (and, in particular, independent of the required level of security), the number of messages in the known concurrent zero-knowledge protocols is required to grow *as a function of the level of security* [59, 47, 57]. Furthermore, severe lower bounds on the number of communications rounds have been shown for black-box zero-knowledge proofs [16]. We nevertheless mention that the question of whether there exists a (non black-box) concurrent zero-knowledge protocol, that only uses a constant number of communication rounds, is still open.

Finally, concerning universal composable protocols, it is known that universally composable zero-knowledge proofs can not be obtained in the standard model [12].

1.4 Ways to Overcome the Limitations

In order to overcome the above-mentioned limitations/problems two lines of research have been investigated in literature.

- **Resorting to models with "stronger" set-up assumptions**. The most popular strengthening of the plain model is the assumption of a *shared object*. Other strengthenings include physical assumptions such as timing (c.f. [25]).
- **Weakening the notion of zero-knowledge**.

Shared Object Models One important vein of research attempts to overcome limitations of zero-knowledge proofs by assuming that the protocol participants have access to a shared object with certified properties. Such an object can be viewed as a very limited intervention by a trusted third party.

It has been shown that efficient, or powerful, protocols can be constructed in such settings. Two of the most popular of these models are the *Common Reference String Model* [9], and the *Random Oracle Model* [6]. The Common Reference String Model relates to a setting where all parties have access to string that has been *ideally* chosen from the uniform distribution, while the Random Oracle Model relates to a setting where all parties have access to an *ideal* random function.

In both the above models it can be shown that *non-interactive* (i.e., a single message is sent by the prover to the verifier) zero-knowledge proofs can be obtained. Whereas the former model has mostly been used in order to obtain efficient (or practical) protocols, the latter has primarily been used to show strong results in the framework for universal composability. In particular, the existence of UC zero-knowledge has been shown in the CRS model [12].

Weakenings of zero-knowledge Another vein of research attempts to provide weakenings of the notion of zero-knowledge proofs. Although this investigation started in the late 80's, surprisingly only very few meaningful weakenings of the notion of zero-knowledge have appeared in the literature. An important notion surging from this research is the notion of *witness indistinguishable proofs* [31]. Intuitively an interactive proof is witness indistinguishable if the proof does not leak any information about what "witness" the prover uses in the proof. Recall the example of a prover that wants to convince a verifier that a specific equation has a solution. A witness indistinguishable proof does not reveal what solution the prover has found to the equation. (Note, however, that if there only exists one solution to the equation, this solution can be entirely revealed). An important aspect of witness indistinguishable proofs is that the witness indistinguishability property holds also when multiple witness indistinguishable proofs are concurrently executed. In particular, this allows for the construction of efficient "concurrent" witness indistinguishable proofs using only a constant number of communication rounds.

1.5 Brief Summary of Our results

In this thesis we address both the above mentioned research directions.

Shared Object Models We take a critical look at the definition of zero-knowledge in the Common Reference String (CRS) and the Random Oracle (RO) Model. Our results show that there exists a specific natural security property that is not captured by these definition, although it is captured by the standard zero-knowledge definition. This is the property of *deniability*. Deniability means that the transcript of the interaction should not leave any traces of the fact that the interaction took place. Note that deniability is not merely an academic concern, but rather reflects an important issue of integrity for many real-life applications. For example, it might not be desirable that a shop owner obtains a written evidence of the fact that a certain customer bought something in his shop.

The issue of deniability also addresses some fundamental definitional issues in the framework for Universal Composability. In particular, it shows that the notion of UC zero-knowledge does not capture the concern for deniability.

We next investigate the possibility of obtaining zero-knowledge proofs in these models, that do satisfy deniability. Our results are different for the CRS and the RO models.

Concerning the CRS model, our results are negative in nature. In fact, we show that known impossibility for zero-knowledge in the standard model (i.e., without shared objects) also hold for the CRS model with respect to deniable zero-knowledge proofs.

Concerning the RO model, on the other hand, we obtain both negative and positive results. In fact, we are able to determine the exact number of communication rounds needed in order to implement deniable zero-knowledge proofs in RO model. More specifically, we show that two-rounds are both necessary and sufficient to implement deniable zero-knowledge protocols in the RO model. We furthermore show that two-rounds are sufficient in order to implement a protocol that also is secure under concurrent composition.

Relaxing zero-knowledge We propose a generalization of zero-knowledge proofs called simulatable proofs. This notion encompasses in a natural way both the notion of zero-knowledge and the notion of witness indistinguishability. Whereas zero-knowledge proofs require the existence of a *polynomial-time* simulator, we say that an interactive proof is $T(\cdot)$-simulatable if there exist a simulator, for the protocol, with running time $T(\cdot)$. It can be seen that the notion of witness indistinguishability, in fact, is equivalent to simulation by an unbounded machine.

We then propose to investigate the notion of quasi-polynomial time simulatable proofs (this is trivially a weakening of zero-knowledge and a strengthening of witness indistinguishability). We show that such proof systems can replace the use of zero-knowledge proofs in many applications. Nevertheless this relaxation allows us to overcome most of the impossibility results known for zero-knowledge proofs. In particular, it provides a straight-forward way of dealing with issues of concurrency. As an example, we mention that we are able to construct a two-round quasi-polynomial time simulatable protocol that is secure under concurrent executions (note that two-rounds are not sufficient to implement even *stand-alone* secure zero-knowledge protocols).

Our presentation also contains various different protocols solving different problems related to the notion of quasi-polynomial time simulations, and their application to protocol composition. We finish our investigation by providing a complete classification of the number of communication rounds needed in order to implement quasi-polynomial time simulatable protocols in two different setting.[6]

1.6 How to Read This Thesis

Chapter 2 contains general definitions and preliminaries. Most of these definitions are quite standard, but in some cases we provide generalizations of the standard definitions. This chapter also contains two new lemmas concerning zero-knowledge protocols in models with shared objects. Chapter 3 contains our results on relaxations of the notion of zero-knowledge. In Chapter 4 we investigate the notion of zero-knowledge in models with shared objects.

[6]The two settings referred to here are interactive proof with computational soundness (a.k.a arguments) and interactive proofs with unconditional soundness.

1.7 Contributions

This work in this thesis is based on the following two papers:

- R. Pass, *On Deniability in the Common Reference String and Random Oracle Models*, In Advances in Cryptology - CRYPTO 2003, Lecture Notes in Computer Science 2729, pages 195–210, 2003
- R. Pass, *Simulation in Quasi-Polynomial Time and Its Application to Protocol Composition*, In Advances in Cryptology - EUROCRYPT 2003, Lecture Notes in Computer Science 2656, pages 160–176, 2003

Chapter 2

Preliminaries

2.1 General

2.1.1 Basic notation

We let N denote the set of all integers. For any integer $m \in N$, denote by $[m]$ the set $\{1, 2, \ldots, m\}$. For any $x \in \{0,1\}^*$, we let $|x|$ denote the size of x (i.e., the number of bits used in order to write it). For two machines M, A, we let $M^A(x)$ denote the output of machine M on input x and given oracle access to A. The term negligible is used for denoting functions that are (asymptotically) smaller than one over any polynomial. More precisely, a function $\nu(\cdot)$ from non-negative integers to reals is called negligible if for every constant $c > 0$ and all sufficiently large n, it holds that $\nu(n) < n^{-c}$.

2.1.2 Probabilistic notation

Denote by $x \overset{\text{R}}{\leftarrow} X$ the process of uniformly choosing an element x in a set X. If $B(\cdot)$ is an event depending on the choice of $x \overset{\text{R}}{\leftarrow} X$, then $\Pr_{x \leftarrow X}[B(x)]$ denotes the probability that $B(x)$ holds when x is chosen with probability $1/|X|$. Namely,

$$\Pr_{x \leftarrow X}[B(x)] = \sum_x \frac{1}{|X|} \cdot \chi(B(x))$$

where χ is an indicator function so that $\chi(B) = 1$ if event B holds, and equals zero otherwise. We denote by U_n the uniform distribution over the set $\{0,1\}^n$.

2.1.3 Witness Relations

We recall the definition of a witness relation for an $\mathcal{NP}$ language [32].

Definition 1 (Witness relation) *A witness relation for a language $L \in \mathcal{NP}$ is a binary relation R_L that is polynomially bounded, polynomial time recognizable and* *characterizes L by*

$$L = \{x : \exists y \, s.t. \, (x, y) \in R_L\}$$

We say that y is a witness for the membership $x \in L$ if $(x, y) \in R_L$. We will also let $R_L(x)$ denote the set of witnesses for the membership $x \in L$, i.e.,

$$R_L(x) = \{y : (x, y) \in L\}$$

In the following, we assume a fixed witness relation R_L for each language $L \in \mathcal{NP}$.

2.1.4 Indistinguishability

We define indistinguishability for *non-uniform* machines. This notion of indistinguishability is sometimes called *indistinguishability by circuits.*

Let $S \subseteq \{0,1\}^*$ be a set of strings. A probability ensemble indexed by S is a sequence of random variables indexed by S. Namely, any $X = \{X_w\}_{w \in S}$ is a random variable indexed by S.

Definition 2 ($T(\cdot)$-indistinguishability) *Two ensembles $X = \{X_w\}_{w \in S}$ and $Y = \{Y_w\}_{w \in S}$ are said to be indistinguishable in time $T(\cdot)$ if for every probabilistic algorithm D with running time $T(\cdot)$ in the length of its second input, there exists a negligible function $\nu(\cdot)$ so that for every $w \in S$, and every auxiliary input $z \in \{0,1\}^*$:*

$$|Pr[D(X_w, w, z) = 1] - Pr[D(Y_w, w, z) = 1]| < \nu(|w|)$$

The algorithm D is referred to as the distinguisher.

In general we will most often be interested in ensembles that are indistinguishable for polynomial time. We call such ensembles computationally indistinguishable, or simple indistinguishable.

Definition 3 (Computational indistinguishability) *Two ensembles $X = \{X_w\}_{w \in S}$ and $Y = \{Y_w\}_{w \in S}$ are said to be computationally indistinguishable if X, Y are $p(\cdot)$-indistinguishable for every polynomial $p(\cdot)$.*

We next define statistical closeness, by removing the bound on the running time of the distinguisher.

Definition 4 (Statistical Closeness) *Two ensembles $X = \{X_w\}_{w \in S}$ and $Y = \{Y_w\}_{w \in S}$ are said to be statistically close if X, Y are $f(\cdot)$-indistinguishable for every function $f(\cdot)$.*

Strong indistinguishability We also define a stronger version of super-polynomial time indistinguishability. Whereas the definition of $T(\cdot)$-indistinguishability only requires that the distinguishing gap is negligible, the stronger definition requires the gap to be at most $1/T(\cdot)$.

Definition 5 (Strong $T(\cdot)$-indistinguishability) *Two ensembles $X = \{X_w\}_{w\in S}$ and $Y = \{Y_w\}_{w\in S}$ are said to be strongly indistinguishable in time $T(\cdot)$ if for every probabilistic algorithm D with running time $T(\cdot)$ in the length of its second input, every $w \in S$, and every auxiliary input $z \in \{0,1\}^*$:*

$$|Pr[D(X_w, w, z) = 1] - Pr[D(Y_w, w, z) = 1]| < 1/T(|w|)$$

Note that if two ensembles X, Y are strongly $f(n)$-indistinguishable for every function $f(n)$, then X and Y are identical.

2.2 Cryptographic Assumptions

2.2.1 One-way Functions

Intuitively one-way functions are functions that are easy to compute, but hard to invert. Here "easy" means, achievable in polynomial time, and "hard" normally means not achievable in polynomial time. In the following we will sometimes rely on slightly stronger assumptions than the most commonly used. Namely we assume the existence of one-way functions where inverting the function is hard for subexponential time.[1]

Definition 6 (One-wayness for time $T(\cdot)$) *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called one-way for time $T(\cdot)$ if the following two conditions hold:*

- *Easy to compute: There exist a (deterministic) polynomial-time algorithm A such that on input x, A outputs $f(x)$.*
- *Hard to invert: For every probabilistic algorithm A' with running time bounded by $T(n)$, all sufficiently large n's, every auxiliary input $z \in \{0,1\}^*$,*

$$Pr[A'(f(U_n), z) \in f^{-1}(f(U_n))] < \frac{1}{poly(T(n))}$$

where U_n is a random variable uniformly distributed in $\{0,1\}^n$.

Definition 7 (One-wayness) *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called one-way if there f is one-way for time $p(n)$ for every polynomial $p(n)$.*

Definition 8 (One-wayness for subexponential circuits) *A function $f : \{0,1\}^* \to \{0,1\}^*$ is called one-way for subexponential circuits if there exist a κ such that f is one-way for time 2^{n^κ}.*

[1] We note that this assumptions is, nevertheless, very plausible and has for example been used to construct resettable zero-knowledge in [14].

2.2.2 Hard-core Predicates

A predicate b is a called a hard-core of a function f if it is infeasible to predict $b(x)$, given only $f(x)$, significantly better than using a random guess.

Definition 9 (Hard-core for time $T(\cdot)$) *A polynomial-time computable predicate $b : \{0,1\}^* \to \{0,1\}$ is called a hard-core for time $T(\cdot)$ of a function f if for every probabilistic algorithm A' with running time bounded by $T(n)$, all sufficiently large n's, and every auxiliary input $z \in \{0,1\}^*$,*

$$Pr[A'(f(U_n), z) = B(U_n)] < \frac{1}{2} + \frac{1}{poly(T(n))}$$

where U_n is a random variable uniformly distributed in $\{0,1\}^n$.

The standard notion of hard-core predicates is defined for polynomial-time algorithms.

Definition 10 (Hard-core) *A polynomial-time-computable predicate $b : \{0,1\}^* \to \{0,1\}$ is called a hard-core of a function f if there b is a hard-core for time $p(n)$ of f, for every polynomial $p(n)$.*

In the following we will employ hard-core predicates with subexponential-time security.

Definition 11 (Hard-core for subexponential circuits) *A polynomial-time-computable predicate $b : \{0,1\}^* \to \{0,1\}$ is called a hard-core for subexponential circuits of a function f if there exist a κ such that b is a hard-core for time 2^{n^κ}, for f.*

Goldreich and Levin [68] have shown that a simple hard-core predicate can be constructed assuming the existence of one-way functions. We note that the Goldreich-Levin predicate is also a hard-core predicate for subexponential circuits of a function that is one-way for subexponential circuits.

Theorem 1 (Goldreich-Levin) *If there exist a function $f : \{0,1\}^* \to \{0,1\}^*$ that is one-way for subexponential circuits, then there exist a pair f',b', where $f' : \{0,1\}^* \to \{0,1\}^*$ is a one-way function for subexponential circuits, and $b' : \{0,1\}^* \to \{0,1\}$ is a hard-core predicate for subexponential circuits, for f'. Furthermore, if f is one-to-one, then f' is so as well.*

2.3 Cryptographic Primitives

2.3.1 Interactive Proofs

We use the standard definitions of interactive proofs (and interactive Turing machines) [38, 32] and arguments [10]. Given a pair of interactive Turing machines, P and V, we denote by $\langle P(y), V(z)\rangle(x)$ the random variable representing the (local) output of V when interacting with machine P on common input x, when the random input to each machine is uniformly and independently chosen, and P (resp. V) has auxiliary input y (resp. z).

Definition 12 (Interactive Proof System) *A pair of interactive machines (P, V) is called an* interactive proof system *for a language L if machine V is polynomial-time and the following two conditions hold with respect to some negligible function $\nu(\cdot)$:*

- Completeness: *For every $x \in L$ there exists a (witness) string y such that for every auxiliary input $z \in \{0,1\}^*$*

$$Pr[\langle P(y), V(z)\rangle(x) = 1] \geq 1 - \nu(|x|)$$

- Soundness: *For every $x \notin L$, every interactive machine B and every $y, z \in \{0,1\}^*$*

$$Pr[(\langle B(y), V(z)\rangle(x) = 1] \leq \nu(|x|)$$

In case that the soundness condition is required to hold only with respect to a computationally bounded prover, the pair (P, V) is called an interactive argument *system.*

Definition 12 can be relaxed to require only soundness error that is bounded away from $1 - \nu(|x|)$. This is so, since the soundness error can always be made negligible by sufficiently many parallel repetitions of the protocol. However, in the case of interactive arguments, we do not know whether this condition can be relaxed. In particular, in this case parallel repetitions do not necessarily reduce the soundness error (cf. [5]).

Interactive proofs with efficient provers For cryptographic application it is necessary that the prover strategy is efficient.

Definition 13 (Efficient Provers) *Let (P, V) be an interactive proof (argument) system for the language $L \in \mathcal{NP}$ with the witness relation R_L. We say that (P, V) has an* efficient prover *if P is a probabilistic polynomial-time algorithm and the completeness condition of Definition 12 holds for every $x \in L$, every $y \in R_L(x)$, and every $z \in \{0,1\}^*$.*

Arthur-Merlin Protocols In many application of interactive proofs it is desirable that the verifier only sends random messages. We call such interactive proofs/arguments public-coin or Arthur-Merlin[2]

2.3.2 Zero-knowledge

Loosely speaking, an interactive proof is said to be *zero-knowledge* ($\mathcal{ZK}$) if it yields nothing beyond the validity of the assertion being proved. This is formalized by requiring that the output of every (possibly malicious) verifier interacting with the honest prover P and be simulated by a probabilistic expected polynomial-time machine S (a.k.a. the *simulator*). The idea behind this definition is that whatever V^* might have learned from interacting with P, he could have learned by himself (by running the simulator S).

The notion of $\mathcal{ZK}$ was introduced by Goldwasser, Micali and Rackoff [38]. To make $\mathcal{ZK}$ robust in the context of (sequential) protocol composition, Goldreich and Oren [37] suggested to augment the definition so that the above requirement holds also with respect to all $z \in \{0,1\}^*$, where both V^* and S are allowed to obtain z as auxiliary input.

Definition 14 (Zero-knowledge) *Let (P, V) be an interactive proof (argument) system for the language $L \in \mathcal{NP}$ with the witness relation R_L. We say that (P, V) is zero-knowledge, if for every probabilistic polynomial-time interactive machine V^* there exists a probabilistic expected polynomial-time algorithm S (called the simulator) such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$)*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary $y \in R_L(x)$*
- $\{S(x, z)\}_{z\in\{0,1\}^*, x\in L}$

That is, for every probabilistic algorithm D running in time polynomial in the length of its first input, every polynomial p, all sufficiently long $x \in L$, all $y \in R_L(x)$ and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$|Pr[D(x, z', (\langle P(y), V^*(z)\rangle(x))) = 1] - Pr[D(x, z', S(x, z)) = 1]| < \frac{1}{p(|x|)}$$

One might also consider a weaker variant of $\mathcal{ZK}$ called honest verifier zero-knowledge proofs. As the name suggests a protocol is honest verifier zero-knowledge (HVZK) if the above condition holds for the honest verifier $V^* = V$ (and not necessarily for verifiers V^* that do not follow the protocol).

Statistical and Perfect Zero-knowledge A variant of $\mathcal{ZK}$ that we will use in this thesis is one in which the output of the simulator is statistically close, or even identically distributed to the output of the verifier in a real interaction.

Definition 15 (Statistical/Perfect zero-knowledge) *Let (P, V) be an interactive proof (argument) system for the language $L \in \mathcal{NP}$ with the witness relation R_L. We say that (P, V) is statistical zero-knowledge, if for every probabilistic polynomial-time interactive machine V^* there exists a probabilistic expected polynomial-time algorithm S such that the following two ensembles are statistically close*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary* $y \in R_L(x)$
- $\{S(x,z)\}_{z\in\{0,1\}^*, x\in L}$

We say that (P,V) *is* perfect zero-knowledge *if the above ensembles are identically distributed.*

Black-box zero-knowledge One can consider a "well-behaved" variant of $\mathcal{ZK}$ proofs called black-box zero-knowledge. Loosely speaking, an interactive proof is black-box zero-knowledge if there exists a simulator S that uses the verifier V^* as a black-box in order to perform the simulation.

Definition 16 (Black-box zero-knowledge) *Let* (P,V) *be an interactive proof (argument) system for the language* $L \in \mathcal{NP}$ *with the witness relation* R_L*. We say that* (P,V) *is* black-box zero-knowledge, *if there for every polynomial* $p(n)$ *exists a probabilistic expected polynomial time oracle machine* S *such that for every probabilistic polynomial-time interactive machine* V^* *that uses at most* $p(n)$ *random coins, the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in* $|x|$*)*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary* $y \in R_L(x)$
- $\{S^{V^*}(x,z)\}_{z\in\{0,1\}^*, x\in L}$

At first sight the definition of black-box $\mathcal{ZK}$ might seem very restrictive: the simulator is supposed to act as the prover, except that the simulator does not have the witness! Note, however that the simulator has an important advantage that the prover does not have, namely that it can *rewind* and restart the verifier.

Most known $\mathcal{ZK}$ protocols (with the exception of [1]) and all practical zero-knowledge protocols are black-box $\mathcal{ZK}$.

Sequential Composition of Zero-knowledge Proofs The standard definition of $\mathcal{ZK}$ only guarantees security in the stand-alone setting, i.e., when only considering a single execution of the protocol in isolation. Nevertheless, Goldreich and Oren [37] have shown that Definition 16 is closed under sequential composition, that is sequential repetitions of a $\mathcal{ZK}$ protocol results in a new protocol that still remains $\mathcal{ZK}$.

Concurrent Self-Composition of Zero-knowledge Proofs In asynchronous settings, such as the Internet, it has been noticed that the zero-knowledge property does not imply security under concurrent executions. Dwork, Naor and Sahai [25] introduced the concept of concurrent zero-knowledge, i.e zero knowledge protocols that retain their zero-knowledge property even when an adversary is allowed to interact in many concurrent execution of the same protocol. Since security is only guaranteed to hold when considering many concurrent execution of the *same* protocol, this kind of composition is called concurrent *self*-composition.

The definition of concurrent zero-knowledge follows the definition of $\mathcal{ZK}$ (See Definition 16), yet it requires the existence of a simulator for every "concurrent" verifier V^*. A concurrent verifier V^* is allowed to communicate with multiple provers at the same time and can schedule the messages in the different executions in an arbitrary way.

Dwork et al. proposed a protocol, based on timing assumptions, achieving concurrent $\mathcal{ZK}$. Various protocols have later been presented based on different set-up assumptions [65] [21] [14].

On the other hand, considering the standard model, without any set-up assumptions, Canetti et al [16] have recently shown that protocols for black-box concurrent $\mathcal{ZK}$ for non-trivial languages require $\Omega(\frac{\log n}{\log \log n})$ number of communication rounds (building on the works of [47] [60]). Richardson and Kilian [59] constructed the first concurrent zero-knowledge argument in the standard model. Their protocol, being black-box concurrent zero-knowledge, uses $O(n^{\epsilon})$ number of rounds. Kilian and Petrank later showed that the round complexity of that protocol could be reduced to only a poly-logarithmic number of rounds. Finally, Prabhakaran, Rosen and Sahai [57] achieved a round complexity of $\tilde{O}(\log n)$.

Recently, Barak [1] presented the first $\mathcal{ZK}$ argument having non black-box simulators under reasonable assumptions.[2] Barak's protocol preserves the zero-knowledge property under concurrent executions where the number of protocols run is bounded by an a-priori specified polynomial in the security parameter. Nevertheless, the question whether there exists a constant-round concurrent zero-knowledge argument (with remains secure under unbounded concurrent composition) still remains open.

Universally Composable Zero-knowledge Recently, an even more realistic setting was considered in the framework for Universal Composability (UC) [12]. The UC setting considers the security of a protocol when it is "plugged-in" into arbitrary other protocol executions. Loosely speaking, while the notion of concurrent zero-knowledge only guarantees the zero-knowledge property of the protocol when many executions of the *same* protocol are simultaneously executed, universally composable zero-knowledge guarantees security even when the verifier is simultaneously participating in other protocol executions. We omit a formal definition of UC $\mathcal{ZK}$ and refer the reader to [12].

Although a very powerful notion, the existence of UC $\mathcal{ZK}$ in the standard model (without set-up assumptions) has been ruled out [12]. Nevertheless, Canetti and Fischlin have shown that UC $\mathcal{ZK}$ can be constructed in the Common Reference String model [13].

[2] Actually, Hada and Tanaka [42] had previously shown the existence of a three round non black-box zero-knowledge protocol under very strong assumptions (that the authors themselves qualified as unreasonable).

2.3.3 Commitment Schemes

Commitment schemes are used to enable a party, known as the *sender*, to commit itself to a value while keeping it secret from the *receiver* (this property is called hiding). Furthermore, the commitment is binding, and thus in a later stage when the commitment is opened, it is guaranteed that the "opening" can yield only a single value determined in the committing phase. Commitment schemes come in two different flavors, perfectly-binding and perfectly-hiding. We sketch the properties of each one of these flavors. Full definitions can be found in [32].

Perfectly-binding: In perfectly binding commitments, the binding property holds against unbounded adversaries, while the hiding property only holds against computationally bounded (non-uniform) adversaries. Loosely speaking, the perfectly-binding property asserts that the transcript of the interaction fully determines the value committed to by the sender. The computational-hiding property guarantees that the commitments to any two different values are computationally indistinguishable.

Perfectly-hiding: In perfectly-hiding commitments, the hiding property holds against unbounded adversaries, while the binding property only holds against computationally bounded (non-uniform) adversaries. Loosely speaking, the perfectly-hiding property asserts that commitments to any two different values are identically distributed. The computational-binding property guarantees that no (non-uniform) polynomial time machine is able to open a given commitment in two different ways.

We mention that in some applications it is often convenient to relax the perfectly-binding or the perfectly-hiding properties to only statistical binding or hiding. Loosely speaking, the statistical binding property asserts that with overwhelming probability (instead of probability 1) over the over the coin-tosses of the receiver, the transcript of the interaction fully determines the committed value. The statistical hiding property asserts that that commitments to any two different values are statistically close (i.e. have negligible statistical difference instead of being identically distributed).

Non-interactive perfectly-binding commitment schemes can be constructed using any 1–1 one-way function (see Section 4.4.1 of [32]). Allowing some minimal interaction (in which the receiver first sends a single message), statistically-binding commitment schemes can be obtained from any one-way function [50, 43]. Perfectly-hiding commitment schemes can be constructed from any one-way permutation [52]. However, *constant-round* schemes are only known to exist under stronger assumptions; specifically, assuming the existence of a collection of certified clawfree functions [34] (see also [32], Section 4.8.2.3). Constant-round statistically-hiding commitments can be constructed under the potentially weaker assumption of of collision-resistant hash functions [53, 22].

2.3.4 Proofs of Knowledge

Informally an interactive proof is a proof of knowledge if the prover convinces the verifier not only of the validity of a statement, but also that it possesses a witness for the statement. This notion is formalized by the introduction of an machine E, called an extractor. As the name suggests, the extractor E is supposed to extract a witness from any malicious prover that succeeds in convincing an honest verifier. More formally,

Definition 17 (Proof of knowledge) *Let (P, V) be an interactive proof system for the language L. We say that (P, V) is a proof of knowledge for the witness relation R_L for the language L it there for every polynomial $p(n)$ exists a probabilistic expected polynomial-time machine E (called extractor) and a negligible function $\nu(n)$ such that for every probabilistic polynomial-time machine P^* which uses at most $p(n)$ random coins, every $x \in \{0,1\}^n$, every auxiliary input z to P^*,*

$$Pr[\langle P^*(z), V\rangle(x) = 1] \;<\; Pr[E^{P^*}(x, z) \in R_L(x)] + \nu(n)$$

In the following we will also be needing a restricted form of proofs of knowledge, namely special-sound proofs. Such protocols are of special interest to us since many efficient protocol (such as [41, 63]) actually are of this type.

Definition 18 (Special soundness) *Let Π is a three round interactive proof for the language $L \in \mathcal{NP}$, with witness relation R_L. We say that the protocol Π is special sound if there exist a probabilistic polynomial-time extractor machine E, such that for all $x \in L$ and all pairs of accepting transcripts for proving x, $T_1 = (a, b_1, c_1), T_2 = (a, b_2, c_2)$, where $b_1 \neq b_2$, $E(T_1, T_2) \in R_L(x)$.*

It can be seen that if an interactive proof is special sound it is also a proof of knowledge. [21]

2.3.5 Witness Indistinguishability

The notion Witness Indistinguishability ($\mathcal{WI}$) was introduced by Feige and Shamir in [29] as a weaker alternative to zero-knowledge. It has later proved to be an excellent tool to achieve zero-knowledge [28], [30], [59], [1]. Intuitively an interactive proof of an $\mathcal{NP}$ relation, in which the prover uses one of several secret witnesses is witness indistinguishable if the verifier can not tell what witness the prover has used. We further say that an interactive proof is witness independent if the verifier's view is equally distributed independently of what witness the prover has used. More formally (following [32]),

Definition 19 (Witness Indistinguishability) *Let (P, V) be an interactive proof for the language $L \in \mathcal{NP}$, and R_L be a fixed witness relation for L. We say that (P, V) is witness indistinguishable for R_L if for every probabilistic polynomial-time algorithm V^* and every two sequences $W^1 = \{w_x^1\}_{x\in L}$ and $W^2 = \{w_x^2\}_{x\in L}$, such* *that* $w_x^1, w_x^2 \in R_L(x)$*, the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in* $|x|$*):*

- $\{\langle P(w_x^1), V^*(z)\rangle(x)\}_{x\in L, z\in\{0,1\}^*}$
- $\{\langle P(w_x^2), V^*(z)\rangle(x)\}_{x\in L, z\in\{0,1\}^*}$

That is, for every probabilistic algorithm D *running in time polynomial in the length of its first input, every polynomial* p*, all sufficiently long* $x \in L$*, and all auxiliary inputs* $z, z' \in \{0,1\}^*$ *it holds that*

$$\begin{aligned} &|Pr[D(x, z', \langle P(w_x^1), V^*(z)\rangle(x))) = 1] \\ -&Pr[D(x, z', \langle P(w_x^2), V^*(z)\rangle(x))) = 1]| < \frac{1}{p(|x|)} \end{aligned}$$

We further say that (P, V) *is witness independent for* R_L *if the above ensembles are identically distributed.*

Zaps Zaps are two-round $\mathcal{WI}$ public-coin proofs where the first message can be fixed once and for all [66]. Dwork and Naor showed that zaps for languages in $\mathcal{NP}$ can be constructed based on trap-door permutations [66]. Their construction relies on the existence of non-interactive $\mathcal{ZK}$ proofs in the Common Reference String Model (See Definition 23) and can be outlined as follows:

Suppose there exists a non-interactive $\mathcal{ZK}$ proof for the language L using a CRS string of length l. Then the following protocol is a $\mathcal{WI}$ proof of $x \in L$.

A ZAP - the two-round $\mathcal{WI}$ proof of Dwork and Naor

$V \to P$: Sends a random k-bit string $\rho = b_1...b_k$ which is interpreted as $B_1...B_m$, where B_i denotes the i'th block of l consecutive bits.

$P \to V$: The prover chooses and sends a random l-bit string $C = c_1...c_l$. For $j = 1$ to m the prover also sends a non-interactive $\mathcal{ZK}$ proof that $x \in L$ using $B_j \oplus C$ as CRS string.

2.3.6 Witness Hiding

Witness Hiding ($\mathcal{WH}$) interactive proofs were introduced by Feige and Shamir in [29]. Intuitively an interactive proof for an $\mathcal{NP}$ relation is witness hiding if a verifier interacting with the prover can not compute any new witnesses that he did not know before the interaction. In order to define this notion formally we start by defining hard instance ensembles (following [32]):

Definition 20 (Hard instance ensembles) *Let $L \in \mathcal{NP}$, R_L be a witness relation for L and $X = \{X_n\}_{n\in\mathrm{N}}$ a probability ensemble such that X_n ranges over $L \cap \{0,1\}^n$. We say that X is* hard for R_L *if for every expected polynomial time probabilistic witness finding algorithm F, all $z \in \{0,1\}^{poly(n)}$, the probability*

$$Pr[F(X_n, z) \in R_L(X_n)]$$

is negligible (as a function of n).

Witness hiding can now be defined, using hard-instance subsets:

Definition 21 (Witness Hiding) *Let (P,V) be an interactive proof for a language $L \in \mathcal{NP}$, R_L a witness relation for L, and let $X = \{X_n\}_{n\in\mathrm{N}}$ be a hard-instance ensemble for R_L. We say that (P,V) is* witness hiding for the relation R_L under the instance ensemble X *if for every probabilistic polynomial time algorithm V^* and all $z \in \{0,1\}^*$ the probability:*

$$Pr[\langle P(Y_n), V^*(z)\rangle(X_n) \in R_L(X_n)]$$

is negligible (as a function of n), where Y_n is arbitrarily distributed over $R_L(X_n)$.

We simply say that (P,V) is witness hiding for R_L *if it is witness hiding under all hard-instance ensembles X for R_L.*

2.4 Models with Shared Objects

In order to overcome limitations in the standard (plain) model several models with various extra set-up assumptions have been suggested.

The most popular[3] way of circumventing impossibility results in the plain model is to extend the model by introducing a *shared object* having some *certified* properties. Two important examples of such models are the *Common Reference String* (CRS) model [9], and the *Random Oracle* (RO) model [6]. The Common Reference String Model relates to a setting where all parties have access to string that has been *ideally* chosen from the uniform distribution, while the Random Oracle Model relates to a setting where all parties have access to an *ideal* random function.

In both the above models it can be shown that *non-interactive* (i.e., a single message is sent by the prover to the verifier) zero-knowledge proofs can be obtained. Whereas the latter model has mostly been used in order to obtain efficient (or practical) protocols, the former has primarily been used to show strong results in the framework for universal composability. In particular the existence of UC $\mathcal{ZK}$ has been shown in the CRS model [12].

[3]Different set-up assumptions include timing assumptions [25, 65], and the Public-Key models [14, 21].

2.4.1 The Common Reference String Model

The Common Reference String (CRS) Model was first introduced by Blum, Feldman and Micali in order to construct non-interactive zero-knowledge proofs [9]. It has recently become a popular model, used to construct protocols preserving security also in the concurrent setting. Most notably, it has be used in order to provide strong positive results in the framework for Universal Composability (e.g. [12] [13] [17]). In the CRS model, a random string is chosen at start-up and is then made available to all parties.

Definition 22 (Interactive proof in the CRS model) *A pair of interactive machines, (P, V), is called an interactive proof system in the CRS model, for a language $L \in \mathcal{NP}$, if the machine V is polynomial-time and the following two conditions hold*

- Completeness: *For every $x \in L$ there exists a (witness) string y such that for every auxiliary input $z \in \{0,1\}^*$*

$$Pr[(\langle P(y), V(z)\rangle(x,r) = 1] \geq 1 - \nu(|x|)$$

  where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$

- Soundness: *For every $x \notin L$, every interactive machine B and every $y, z \in \{0,1\}^*$*

$$Pr[\langle B(y), V(z)\rangle(x,r) = 1] \leq \nu(|x|)$$

where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$

Interactive arguments are defined in analogy with interactive proofs, with the only difference that the soundness condition only needs to hold against polynomially bounded adversaries B.

Definition 23 (Zero-knowledge in the CRS model) *Let (P, V) be an interactive proof (argument) system in the CRS model for the language $L \in \mathcal{NP}$ with the witness relation R_L. We say that (P, V) is zero-knowledge in the CRS model if there for every probabilistic polynomial-time interactive machine V^* exists an expected polynomial-time algorithm S (called the simulator) such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$)*

- $\{(r, \langle P(y), V^*(z)\rangle(x,r))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary $y \in R_L(x)$*
- $\{S(x,z)\}_{z\in\{0,1\}^*, x\in L}$

where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$.
That is, for every probabilistic algorithm D running in time polynomial in the length

of its first input, every polynomial p, all sufficiently long $x \in L$, all $y \in R_L(x)$ and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$|Pr[D(x, z', (r, \langle P(y), V^*(z)\rangle(x, r))) = 1] - Pr[D(x, z', S(x, z)) = 1]| < \frac{1}{p(|x|)}$$

where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$.

On the Power of the Simulator in the CRS Model We note that in Definition 23 the simulator is allowed to *choose* the CRS string. In fact, this is an important part of all known simulation techniques in the CRS model. The simulator normally chooses the CRS in a particular way, building in some "trap-door" information in it. This trap-door information will thereafter allow the simulator to "emulate" the honest prover without actually possessing a witness to the statement to be proved. (Note that the real prover or verifier are clearly not allowed to choose the CRS. This would completely jeopardize the security of the protocol).

2.4.2 The Random Oracle Model

In the Random Oracle (RO) model, a random function $RO : \{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$ is selected at startup, and is thereafter made accessible to all parties through oracle calls. The model was formally introduced by Bellare and Rogaway in 1993 [6] in order to bridge the gap between provably secure and practical cryptography.[4] The RO model attempt to fill this space by providing a framework for the construction of practical, yet provably secure protocols. Since its introduction the RO model has indeed been successfully used to formally analyze the security of many practical protocols.

Note, however, that in the "real" world truly random functions unfortunately do not exist. Bellare and Rogaway therefore suggest, as a heuristic step, to implement the RO using a hashfunction. This last step has meet serious criticism (for example it has been shown that there exists encryption schemes that are provably secure in the RO model, yet are insecure for every instantiation of the RO with hashfunction [15]).

Interactive proofs in the RO model are defined in analogy with interactive proofs in the CRS model:

Definition 24 (Interactive proof in the RO model) *A pair of interactive machines, (P, V), is called an interactive proof system in the RO model, for a language $L \in \mathcal{NP}$, if the machine V is polynomial-time and the following two conditions hold*

[4] The methodology of characterizing hashfunctions as random functions had, however, already been used for a long time as a heuristic and can be traced back to the Fiat-Shamir signature scheme from 1986 [31].

- Completeness: *For every $x \in L$ there exists a (witness) string y such that for every auxiliary input $z \in \{0,1\}^*$*

$$Pr[\langle P^{RO}(y), V^{RO}(z)\rangle(\{x\}) = 1] \geq 1 - \nu(|x|)$$

where RO is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)} \rightarrow \{0,1\}^{poly(|x|)}$

- Soundness: *For every $x \notin L$, every interactive machine B and every $y, z \in \{0,1\}^*$*

$$Pr[\langle B^{RO}(y), V^{RO}(z)\rangle(\{x\}) = 1] \leq \nu(|x|)$$

where RO is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)} \rightarrow \{0,1\}^{poly(|x|)}$

Informally we say that an interactive proof (P, V) in the RO model for the language $L \in \mathcal{NP}$, with witness relation R_L, is zero-knowledge in the RO model if there for every probabilistic polynomial-time verifier V^* exists an expected polynomial time probabilistic simulator algorithm S such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$):

- $\{(RO, \langle P^{RO}(y_x), V^{*RO}(z)\rangle(x))\}_{x \in L, z \in \{0,1\}^*}$ for arbitrary $y_x \in R_L(x)$

- $\{S(x, z)\}_{x \in L, z \in \{0,1\}^*}$

where RO is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)} \rightarrow \{0,1\}^{poly(|x|)}$

In analogy with the zero-knowledge definition in the CRS model, the simulator is allowed to choose the random oracle, and has for this purpose a special fill out function. We refer the reader to [6] for a more formal definition.

The simulation of the random oracle gives the simulator two extra advantages (in comparison to the plain-model simulator):

- The simulator can see for what values parties ask the random oracle,

- The simulator chooses in what way to answer the queries to the random oracle.

2.4.3 Cryptographic Primitives with Shared Objects

We will in the following use the meta-phrase *shared object*, meaning either a CRS string or a random oracle. The shared object R is thus a random oracle of length either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$.

In this section, we extend the notions of $\mathcal{WI}$, $\mathcal{WH}$ and proofs of knowledge, to models with shared objects by providing all parties with access to the shared object.

Definition 25 (Witness Indistinguishability with Shared Objects) *Let (P, V) be an interactive proof in a model with shared objects for the language $L \in \mathcal{NP}$, and R_L be a fixed witness relation for L. We say that (P, V) is* witness indistinguishable *for R_L if for every probabilistic polynomial-time algorithm V^* and every two sequences $W^1 = \{w_x^1\}_{x \in L}$ and $W^2 = \{w_x^2\}_{x \in L}$, such that $w_x^1, w_x^2 \in R_L(x)$, the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$):*

- $\{\langle P^R(w_x^1), V^{*R}(z)\rangle(x)\}_{x \in L, z \in \{0,1\}^*}$
- $\{\langle P^R(w_x^2), V^{*R}(z)\rangle(x)\}_{x \in L, z \in \{0,1\}^*}$

where R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$. That is, for every probabilistic algorithm D running in time polynomial in the length of its first input, every polynomial p, all sufficiently long $x \in L$, and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$\begin{aligned} &|Pr[D^R(x, z', \langle P^R(w_x^1), V^{*R}(z)\rangle(x))) = 1] \\ -&Pr[D^R(x, z', \langle P^R(w_x^2), V^{*R}(z)\rangle(x))) = 1]| < \frac{1}{p(|x|)} \end{aligned}$$

where R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$

We note that in the case of *interactive* proofs in models with shared object model, the definition can in fact be slightly weakened, by not giving the distinguisher access to the shared object (See Appendix A).

Definition 26 (Witness Hiding with Shared Objects) *Let (P, V) be an interactive proof in a model with shared objects for a language $L \in \mathcal{NP}$, R_L a witness relation for L, and let $X = \{X_n\}_{n \in \mathrm{N}}$ be a hard-instance ensemble for R_L. We say that (P, V) is* witness hiding for the relation R_L under the instance ensemble X *if for every probabilistic polynomial-time algorithm V^* and all $z \in \{0,1\}^*$ the probability:*

$$Pr[\langle P^R(Y_n), V^{*R}(z)\rangle(X_n) \in R_L(X_n)]$$

is negligible (as a function of n), where Y_n is arbitrarily distributed over $R_L(X_n)$, and R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$.

Definition 27 (Proof of Knowledge with Shared Objects) *Let (P, V) be an interactive proof system in a model with shared objects for the language L. We say that (P, V) is a proof of knowledge for the witness relation R_L for the language L it there for every polynomial $p(n)$ exists a probabilistic expected polynomial-time machine E (called extractor) and a negligible function $\nu(n)$ such that for every* *probabilistic polynomial-time machine P^* which uses at most $p(n)$ random coins, such that for every P^*, every $x \in \{0,1\}^n$, every auxiliary input z to P^*,*

$$Pr[\langle P^{*R}(z), V^R\rangle(x) = 1] \ < \ Pr[E^{P^{*R}}(x,z) \in R_L(x)] + \nu(n)$$

where R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$.

Remark 1 *We note that we have used a strict definition of proofs of knowledge. In our definition the extractor is* not *allowed to change the shared object, i.e., the CRS string of the random oracle. Other, more relaxed, definitions (see for example [61]) have been used in the context of non-interactive proofs. In these definitions, as opposed to our treatment, the extractor is allowed "emulate" the shared object for the prover during the extraction process.*

2.5 ZK in the CRS/RO Model Implies $\mathcal{WH}$ and $\mathcal{WI}$

In this section, we show two lemmas concerning the $\mathcal{WH}$ and $\mathcal{WI}$ properties of zero-knowledge proofs (or arguments), in the CRS and RO models.

Lemma 1 *Suppose that (P,V) is a zero-knowledge proof (argument), in the CRS/RO model, for the language L. Then, for all witness relations R_L for L, (P,V) is witness hiding in the CRS/RO model.*

Proof Let R be the shared object in the model, i.e. a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$. Suppose, for contradiction, that there exists a witness relation R_L for L such that (P,V) is not $\mathcal{WH}$ for R_L. That is, there exists a hard instance probability ensemble $\{X_n\}_{n\in\mathrm{N}}$, for R_L, for which an adversary A can extract a witnesses, after having interacted with a prover using (P,V), with non-negligible probability. In other words, there exists a PPT machine V^* and a non-negligible function p such that for infinitely many n there exists a $z' \in \{0,1\}^{poly(n)}$ such that

$$Pr[\langle P^R(Y_n^R), V^{*R}(z')\rangle(X_n^R) \in R_L(X_n^R)] > p(n)$$

where Y_n is arbitrarily distributed over $R_L(X_n^R)$.

The zero-knowledge property of (P,V) yields the existence of a simulator S, such that for all $x \in L$, and all $z \in \{0,1\}^*$, and arbitrary $y_x \in R_L(x)$ the following distributions are indistinguishable

- $(R, \langle P^R(y_x), V^{*R}(z)\rangle(x)))$
- $S(z,x)$

This, in particular, means that the following distributions also are indistinguishable:

- $(R, \langle P^R(Y_n, V^{*R}(z')\rangle(X_n)))$

- $S(z', X_n)$

Since, by our assumptions, V^* succeeds in finding a witness for elements from the hard-instance ensemble X_n with non-negligible probability, it follows that S will able to do the same thing (without the help of the prover). This thus contradicts the hard-instance property of X_n. ■

Remark 2 *We note that a corresponding lemma was proved for the plain model (i.e., that $\mathcal{ZK}$ in the plain model, without shared objects, implies $\mathcal{WH}$) in [29].*

Lemma 2 *Let the language $L \in \mathcal{NP}$, R_L be a witness relation for L, and (P, V) be a zero-knowledge proof (argument) in the CRS/RO model for L with efficient prover for R_L. Then (P, V) is witness indistinguishable for R_L in the CRS/RO model.*

Proof Let R be the shared object in the model, i.e. a random variable uniformly distributed in either $1^{poly(n)} \to \{0,1\}$ or $\{0,1\}^{poly(n)} \to \{0,1\}^{poly(n)}$. Suppose, for contradiction, that (P, V) is not $\mathcal{WI}$ for R_L in a model with shared objects. That is there exists a PPT machine V^*, a PPT distinguisher D and a polynomial $p(n)$ such that for infinitely many n there exists an $x \in L \cap \{0,1\}^n$, two witnesses $w_1, w_2 \in R_L(x)$, and auxiliary inputs $z, z' \in \{0,1\}^*$ such that $D(z')$ distinguishes between the following distributions with distinguishing gap $p(n)$:

- $\{\langle P^R(w_1), V^{*R}(z)\rangle\}$
- $\{\langle P^R(w_2), V^{*R}(z)\rangle\}$

i.e.

$$\begin{aligned} &|Pr[D^R(x, z', \langle P^R(w_1), V^{*R}(z)\rangle(x))) = 1] \\ -&Pr[D^R(x, z', \langle P^R(w_2), V^{*R}(z)\rangle(x))) = 1]| > p(|x|) \end{aligned}$$

That is,

$$\begin{aligned} &\frac{1}{\#U_R} \sum_{\sigma \in U_R} |Pr[D^\sigma(x, z', \langle P^\sigma(w_1), V^{*\sigma}(z)\rangle(x))) = 1] \\ &-Pr[D^\sigma(x, z', \langle P^\sigma(w_2), V^{*\sigma}(z)\rangle(x))) = 1]| > p(|x|) \end{aligned}$$

where U_R is either the set of functions $1^{poly(n)} \to \{0,1\}$ or $\{0,1\}^{poly(n)} \to \{0,1\}^{poly(n)}$.

It was shown in [28] how a *bias test* can be used to show the existence of a PPT machine V^*, a PPT distinguisher $\tilde{D}$ and a polynomial p' such that for infinitely many n there exists an $x \in L \cap \{0,1\}^n$, two witnesses $w^1, w^2 \in R_L(x)$ and auxiliary inputs $\tilde{z}, \tilde{z}' \in \{0,1\}^*$ such that

$$\begin{aligned} &|\frac{1}{\#U_R} \sum_{\sigma \in U_R} (Pr[\tilde{D}^\sigma(x, \tilde{z}', \langle P^\sigma(w_1), V^{*\sigma}(\tilde{z})\rangle(x))) = 1] \\ -&Pr[\tilde{D}^\sigma(x, \tilde{z}', \langle P^\sigma(w_2), V^{*\sigma}(\tilde{z})\rangle(x))) = 1])| > p'(|x|) \end{aligned}$$

(Roughly, $\tilde{D}$ simulates the behavior of D but inverts D's output when feed a shared object σ such that $Pr[D^\sigma(x, z', \langle P^\sigma(w_1), V^{*\sigma}(z)\rangle(x))) = 1] < Pr[D^\sigma(x, z', \langle P^\sigma(w_2), V^{*\sigma}(z)\rangle(x))) = 1]$, to prevent a cancellation effect. In order for $\tilde{D}$ to know when to invert, $\tilde{D}$ starts by estimating the above probabilities by sampling. It is thus essential that the auxiliary input $\tilde{z}'$ given to D' contains the witnesses w_1, w_2, and the auxiliary inputs z, z', and that the prover strategy P is efficient.) We can now rewrite the inequality as follows,

$$\begin{aligned} &|Pr[\tilde{D}^{R_1}(x, \tilde{z}', \langle P^{R_1}(w_1), V^{*R_1}(\tilde{z})\rangle(x))) = 1] \\ -&Pr[\tilde{D}^{R_2}(x, \tilde{z}', \langle P^{R_2}(w_2), V^{*R_2}(\tilde{z})\rangle(x))) = 1]| > p'(|x|) \end{aligned}$$

where R_1, R_2 is are random variables distributed in the same way as R.

Now let $S = (S_1, S_2)$ be the zero-knowledge simulator for (P, V), such that $S_1(x, y)$ denotes the simulation of the shared object, and $S_2(x, y)$ the simulation of the output of the verifier, on common input x and auxiliary input z. Consider the following expression:

$$\begin{aligned} &|Pr[\tilde{D}^{R_1}(x, \tilde{z}', \langle P^{R_1}(w_1), V^{*R_1}(\tilde{z})\rangle(x))) = 1] - Pr[\tilde{D}^{S_1(x,\tilde{z})}(x, \tilde{z}', S_2(x, \tilde{z})]| \\ +&|Pr[\tilde{D}^{S_1(x,\tilde{z})}(x, \tilde{z}', S_2(x, \tilde{z})] - Pr[\tilde{D}^{R_2}(x, \tilde{z}', \langle P^{R_2}(w_2), V^{*R_2}(\tilde{z})\rangle(x))) = 1]| \end{aligned}$$

It follows from the fact that S is a zero-knowledge simulator that the expression is negligible (since both the first and the second terms are negligible). On the other hand, by the triangle inequality, the expression is larger or equal to

$$\begin{aligned} &|Pr[\tilde{z}^{R_1}(x, \tilde{z}', \langle P^{R_1}(w_1), V^{*R_1}(\tilde{z})\rangle(x))) = 1] \\ -&Pr[\tilde{z}^{R_2}(x, \tilde{z}', \langle P^{R_2}(w_2), V^{*R_2}(\tilde{z})\rangle(x))) = 1]| > p'(|x|) \end{aligned}$$

The lemma follows by contradiction. ■

Remark 3 *The corresponding lemma was proved for the plain model in [29], and for non-interactive proofs in the CRS model in [28].*

We note that the proof of the fact that $\mathcal{ZK}$ implies $\mathcal{WH}$ in models with shared object is a straight-forward adaptation of the proof of this fact for the plain model [29]. Interestingly, concerning $\mathcal{WI}$, on the other hand, such a simple adaptation can no longer be done. This was shown in [28] already for the case of non-interactive $\mathcal{ZK}$ in the CRS model. We mention that reason for this problem stems from the fact that the definition of $\mathcal{ZK}$ in models with shared object allows the simulator to "choose" the shared object, when performing the simulation.

More precisely, the notion of $\mathcal{WI}$ in the models with shared object considers the output of a verifier after interacting with provers that use different witnesses, but the *same* shared object. In order for the proof that $\mathcal{ZK}$ implies $\mathcal{WI}$ in the plain model to go through, we would need to require that the zero-knowledge simulator could produce a valid simulation for "almost" every randomly chosen shared object.

To sum up, although the above lemmas show positive results concerning the security properties of zero-knowledge protocols in models with shared objects, the non-triviality of the proof of Lemma 2, by itself, shows that special care has to be taken in such models. In Chapter 4 we turn to a more in-depth analysis of problems related to the above issue.

Part I

Results in the Plain Model

Chapter 3

Simulatable Proofs

Abstract

We propose the study of $T(n)$-simulatable proofs, i.e., interactive proofs that have the property that the view of a verifier in an interaction with a prover can be simulated (without the help of the prover) in time $T(n)$. We show that this notion encompassed both the notion of zero-knowledge (i.e. polynomial-time simulatability) and the notion of witness indistinguishability (which we show to be equivalent to simulatability by an unbounded machine).

We next turn to investigate the possibility of $T(n)$-simulatable proofs between the "extremes" of zero-knowledge and witness indistinguishability. In our treatment we focus on quasi-polynomial time simulatable proofs (which thus is a weakening of zero-knowledge proofs, but a strengthening of witness indistinguishable proofs). We show that protocols satisfying this notion can be constructed in settings where the standard zero-knowledge definition is too restrictive. Specifically, we provide constant-round protocols (proven secure under super-polynomial hardness assumptions) for a specific type of quasi-polynomial time simulatable arguments and show that such arguments can be used in advanced (concurrent) composition operations, without any set-up assumptions.

Finally, we provide a complete classification of the round-complexity of quasi-polynomial time simulatable proofs and arguments (interestingly such a classification is not known for zero-knowledge proofs).

3.1 Introduction

The notion of zero-knowledge defines in a very natural way the property that a participant of a protocol is not able to extract *any* information from a protocol execution. The definition of zero-knowledge is captured through the simulation paradigm. Namely, an interactive proof is said to be zero-knowledge if there exist a, so called, simulator that can simulate the behavior of every, possibly malicious, verifier, that is communicating with a prover. The idea behind the simulation paradigm is as follows: Assuming that a malicious verifier succeeds in doing something after having interacted with a prover, then by running the simulator, he could have done it himself, without any interaction with a prover.

Limitations of zero-knowledge proofs Although the notion of zero-knowledge has proven very successful in the design of increasingly complex cryptographic tasks, many severe limitations are known:

1. The impossibility of non-trivial 2-round zero-knowledge arguments. (straightforward extension, from proofs to arguments, of the result of [37])
2. The impossibility of non-trivial 3-round black-box zero-knowledge arguments. [35]
3. The impossibility of non-trivial constant-round concurrent black-box zero-knowledge arguments. [16]
4. The impossibility of non-trivial constant-round black-box zero-knowledge arguments with *strict* polynomial-time simulators. [4]

Limitations of the "simulation paradigm" Shortly after the conception of zero-knowledge proofs, the simulation paradigm and the notion of zero-knowledge was used to formalize the security requirements for general protocols [36]. More specifically, the definition of secure two-party and multi-party computation relies on the simulation paradigm and consequently inherits some of the limitations of the notion of zero-knowledge. Furthermore, recently several severe impossibility results have been showed concerning the concurrent composability of secure two-party and multi-party protocols [48, 49].

New ideas We note that although the notion of zero-knowledge in itself is very beautiful, zero-knowledge is most often used as the "means" to prove the security of protocols, rather than the end-goal.[1] It might thus be conceivable that the notion of zero-knowledge can be relaxed in such a way that it is both "good" enough for applications, yet that it allows for the construction of protocols that bypass the above-mentioned limitations of zero-knowledge (in terms of efficiency and concurrent composability). A first step in this direction was taken by Feige and Shamir, by defining the notion of witness indistinguishability. In this chapter we suggest a different relaxation of the zero-knowledge definition. Our notion is a strict strengthening of the notion of witness indistinguishability, yet it allows us to construct round-efficient protocols in the plain model without any set-up assumptions on which advanced (concurrent) composition operations can be performed. In particular, we able to construct protocols that compose concurrently with themselves and other protocols.

[1] In some rare cases this is not true, like for example Deniable authentication [50].

3.1.1 Simulatable proofs

We propose to study a generalization of zero-knowledge proof, called $T(\cdot)$-*simulatable* proofs. Whereas the standard definition of zero-knowledge requires the existence of a polynomial-time simulator, a $T(\cdot)$-simulatable proof requires the existence of a simulator with running time bounded by $T(\cdot)$. We show that the notion of $T(n)$-simulatable proofs, not only generalizes the notion of zero-knowledge proofs, but it also encompasses the notion of witness indistinguishability. In fact, an interactive proof is witness indistinguishable if and only if it is simulatable by an unbounded machine.

We note that the notion of $T(\cdot)$-simulatability is interesting also between the "extremes" (of zero-knowledge and witness indistinguishability), i.e., when $T(n)$ is a (sub-exponential) super-polynomial function. In fact, the notion of $T(\cdot)$-simulatability provides a natural way of quantifying how much "knowledge" an interactive protocol "leaks".

In this chapter we investigate the possibility of obtaining quasi-polynomial time simulatable proofs and arguments in settings where the notion of zero-knowledge is too restrictive. In order to motivate the new definition we start by mentioning some of the features of $T(n)$-simulatable proofs in general, and quasi-polynomial time proofs in particular.

A strengthening of $\mathcal{WI}$ Clearly, since quasi-polynomial time simulatable arguments are also exponential-time simulatable, quasi-polynomial time simulation is a strictly stronger requirement than $\mathcal{WI}$.

Can be used to replace $\mathcal{ZK}$ In many applications of zero-knowledge proof, these proofs systems can be replaced by $T(n)$-simulatable proofs, *provided one is willing to make quantitatively stronger hardness assumptions.* More specifically, since our notion builds on the simulation paradigm the "logic" behind zero-knowledge also holds for $T(\cdot)$-simulatable proofs: Assume that an adversary receives a proof of a statement and then attempts to break a certain primitive P (which might be related to the statement proved). If the proof that the adversary receives is zero-knowledge it is sufficient to assume that the primitive is hard to break for polynomial-time, in order to make sure that the adversary still won't be able to break it after receiving the proof. This is so since if there would exist an adversary that succeeds in breaking P after receiving the proof, then there exist a polynomial-time simulator who would also succeed (without receiving the proof), contradicting the underlying hardness assumption on the primitive P.

Now, assume that the adversary instead receives a $T(n)$-simulatable proof. Using the same argument, it is sufficient to make the quantitatively stronger assumption that the underlying primitive P is hard for time $T(n)$, in order to make sure that the adversary won't be able to break it after receiving the proof. Note that in the case of quasi-polynomial time simulatable proofs (i.e., when $T(n) = n^{poly(\log n)}$) it is sufficient to assume that the underlying primitive is hard for quasi-polynomial time. Since most natural problems that we believe are hard (on the average) for polynomial time are also believed hard for quasi-polynomial time, this assumption seems rather mild.

Definition extends to general protocol security Whereas the definition of witness indistinguishability only relates to interactive proofs (where there is a witness), the notion of $T(\cdot)$-simulatability applies also to other definitions that rely on the simulation paradigm, such as for example the definition of secure computation. Since most definitions of protocol security actually rely on the simulation paradigm, the feature that our relaxation directly extends to these definition is of great importance.

Guarantee security in the On-line/Off-line Model In many settings it seems reasonable to assume that parties are given a certain on-line time and a certain, longer, off-line time. Such an *on-line/off-line model* can be modeled by letting parties run in polynomial time while being on-line, and time $T(n)$ off-line.

A certain type of $T(n)$-simulatable protocols (which we call strongly $T(n)$-simulatable protocols) have the property of being zero-knowledge in the on-line/off-line model. Roughly speaking, we say that an interactive proof is *strongly* $T(n)$*- simulatable* if it is $T(n)$-simulatable and the output of the simulator is indistinguishable *in time* $T(n)$ from the output of the verifier in a real interaction (whereas $T(n)$-simulatability only requires that these distributions are polynomial-time indistinguishable). In other words, strong $T(n)$-simulatability means that there exist an off-line simulator (with running time $T(n)$) for every on-line adversary such that the output of of the simulator is indistinguishable in time $T(n)$ (i.e., off-line) from the output of the adversary after interaction with a real on-line prover. Strongly $T(n)$-simulatable protocols thus guarantee that anything that a verifier can compute after interaction with a prover, he could have computed by himself off-line.

Allow for Advanced Composition We show that it suffices to relax the zero-knowledge definition to quasi-polynomial time simulatability to obtain protocols on which advanced protocol composition operations can be performed, similar to those guaranteed by the notion of Universal Composability. The key to this is the possibility of constructing quasi-polynomial time simulatable protocols that are proven secure using black-box techniques, *but without the use of rewind*, i.e. straight-line simulatable protocols.

3.1.2 A Caveat

In the following we show that the relaxed notion of quasi-polynomial time simulatability is "easier" to achieve than that of zero-knowledge. More specifically, we show how to construct efficient constant-round protocols that compose concurrently, whereas it is unknown (or even impossible in certain settings) to construct such protocols achieving the standard zero-knowledge definition.

It is nevertheless worthwhile to note that the relaxed definition can not be used in all settings where the standard zero-knowledge definition is used. Recently the concern for *deniability* of zero-knowledge protocols, i.e., the fact that the transcript of a protocol execution does not yield any evidence of the interaction, was addressed in [55] (see Chapter 4). It was formally shown the zero-knowledge definitions in the Common Reference String (and Random Oracle) models do not satisfy deniability. (Furthermore it is shown that known black-box impossibility results for the plain model without set-up assumptions, e.g. [35] [16], also hold for the Common Reference String model when considering deniable protocols). Concerning quasi-polynomial time simulatable proofs, we note that since the running time of the simulator is longer than the allowed running time of the verifier, the simulator can not be run by the verifier. Quasi-polynomial time simulatable proofs therefore do not guarantee deniability.

3.1.3 Our Results

Our main contributions can be summarized as follows:

- We formally define the notion of $T(n)$-simulatable proofs and show that it encompasses the notion of $\mathcal{WI}$. Our characterization of $\mathcal{WI}$ in terms of a simulation-based definition sheds new light on the notion of $\mathcal{WI}$, and might lead to alternative constructions of $\mathcal{WI}$ proofs. Furthermore, whereas the notion of $\mathcal{WI}$ is only defined for $\mathcal{NP}$-languages, our simulation-based characterization of $\mathcal{WI}$ can also be applied to languages outside of $\mathcal{NP}$.

- We show the robustness of the notion of $T(n)$-simulatability by demonstrating a sequential composition lemma.

- We identify a certain class of $T(n)$-simulatable protocols that can be used in advanced composition operations. The power of such protocols is demonstrated in a composition theorem. More precisely, we formally show that *straight-line concurrent $T(n)$-simulatability* (i.e., concurrent $T(n)$-simulatability without the use of rewinding) is a sufficient requirement for a general type of asynchronous protocol composition.

- We construct two different protocols:

  - On the practical side, we construct an *efficient* 4-round straight-line concurrent $n^{poly(logn)}$-perfectly simulatable argument for $\mathcal{NP}$, under the assumption of one-to-one one-way functions secure against subexponential circuits, and perfectly hiding commitments. In analogy with the definition of perfect zero-knowledge, perfect simulation here means that the simulator's output has the same distribution as the verifier's output in a real interaction with a prover. Firstly, since the protocol is straight-line concurrent $n^{poly(logn)}$-simulatable it can be used with our composition theorem. Secondly, since the protocol is perfectly simulatable it is also strongly simulatable, which implies that it is concurrent zero-knowledge in the on-line/off-line model mentioned above. We also mention that the protocol can be constructed through an efficient generic transformation from so called Σ-protocols (i.e., 3-round special-sound public-coin honest-verifier perfect zero-knowledge proofs).

  – On the theoretical side, we construct a 2-round argument for $\mathcal{NP}$ that is straight-line concurrent black-box strict-$n^{poly(logn)}$-simulatable[2], under the assumption that one-to-one one-way functions secure against subexponential circuits exists, and the existence of zaps [66] . We have thus shown that all impossibility results regarding zero-knowledge, mentioned in section 3.1 no longer hold in our relaxed setting.

- Finally, we give a complete characterization of the round-complexity of quasi-polynomial time simulatable proofs and arguments.[3] More specifically, under cryptographic assumptions, we show that 2 rounds are necessary and sufficient for quasi-polynomial time simulatable *arguments*, while 3 rounds are necessary and sufficient for quasi-polynomial time simulatable *proofs*. Interestingly, the exact round-complexity of (non black-box) zero-knowledge proofs is still unknown.

3.1.4 Related Research

Resource-bounded Provers Recently, Dwork and Stockmeyer [65], investigated the possibility of 2-round zero-knowledge protocols in a model where the prover has bounded resources. Their approach is orthogonal to ours; whereas they consider weaker adversaries, we consider a weakening of the notion of zero-knowledge.

On a high-level, the intuition behind, and the structure of, our 2-round protocol is similar to that of [65]. However, since the security definitions are quite different, the techniques used to instantiate the intuition behind the protocol, are very different. Indeed, the results of Dwork and Stockmeyer are quite limited in the setting where the prover's running time is bounded, while we are able to prove security under standard type assumptions. We note, however, that this is due to the fact that the definition used in [65] is more restrictive than ours.

The Timing model Our on-line/off-line model is similar to the timing model introduced by Dwork, Naor and Sahai [25] in the context of concurrent zero-knowledge. We mention that the concurrent zero-knowledge protocol presented in [25] relies on both *time-out* and *delay* mechanism, whereas our protocol only

[2] In this section we emphasize that our protocols are simulatable in *strict* $n^{poly(logn)}$-time, as opposed to expected time. In the rest of the paper we do not emphasize this fact.

[3] We emphasize that this is not simply a black-box characterization (c.f. [35]).

relies on *time-outs*, which drastically improves the efficiency. The on-line/off-line model, however, relies on stronger assumptions than the timing model as it explicitly bounds the on-line running time of malicious parties, whereas the timing model only considers the running time of the honest parties.

Complexity leveraging Canetti et al have, in [14], used the technique of complexity leveraging. The proof of security of our 2-round protocol relies on the same technique.

Straight-line simulatability The application of straight-line simulatability (i.e. black-box simulatability without the use of rewinding) to protocol composition was demonstrated in [12]. Using our relaxed definition we are able to obtain similar effects, but without resorting to set-up assumptions, such as a Common Reference String.

3.1.5 Outline

In section 3.2 we formally define the new notions. Section 3.3 shows how straight-line concurrent $T(n)$-simulatable arguments can be used in advanced composition operations. In section 3.4 and 3.5 two constructions of straight-line concurrent quasi-polynomial time simulatable arguments are given. Finally, in section 3.6, we give a complete characterization of the round-complexity of quasi-polynomial time simulatable proofs and arguments.

3.2 Definitions of the New Notions

As argued in the introduction, for many applications, it is often sufficient to use interactive proofs (or arguments) that can be simulatable within some super-polynomial time $T(n)$.

3.2.1 $T(n)$-simulatable proofs

We start by defining $T(n)$-simulatable interactive proofs and thereafter provide some justifications for the definition. In order to obtain a notion that is robust under composition we restrict our attention to $T(n)$-simulatable proofs, where $T(n)$ is a class of function that is closed under composition with any polynomial.

Definition 28 *Let $T(n)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive proof (or argument) (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is $T(n)$-simulatable if there for every PPT machine V^* exists a probabilistic simulator S with running time bounded by $T(n)$ such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $n = |x|$)*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary* $y \in R_L(x)$
- $\{S(x,z)\}_{z\in\{0,1\}^*, x\in L}$

That is, for every probabilistic algorithm D *running in time polynomial in the length of its first input, every polynomial* p*, all sufficiently long* $x \in L$*, all* $y \in R_L(x)$ *and all auxiliary inputs* $z, z' \in \{0,1\}^*$ *it holds that*

$$|Pr[D(x,z',(\langle P(y), V^*(z)\rangle(x))) = 1] - Pr[D(x,z',S(x,z)) = 1]| < \frac{1}{p(|x|)}$$

Remark 4 *The standard definition of zero-knowledge can be simplified to only provide the distinguisher with the auxiliary input* z *of the verifier (instead of providing it with its own auxiliary input* z' *(see [32] pages 214-215 for a proof of this fact). We note that the proof of [32] no longer holds for* $T(n)$*-simulatable proofs.*

Essentially, the above definition is the same as the definition of zero-knowledge proofs, with the only exception that the simulator is allowed to run in time $T(n)$ instead of in polynomial time. Note that the output of the simulator is only required to be *polynomial-time* indistinguishable from a real execution, i.e. the distinguisher, as well as the distinguishing gap is only is polynomial. This feature makes the definition less robust than the definition of zero-knowledge. In particular, as we will see in the next section, standard "hybrid"-arguments [69] can no longer be used with the above definition. Looking ahead, in section 3.2.2 we therefore introduce a stronger notion called *strongly* $T(n)$*-simulatable proofs*, for which the simulator's output is required to be strongly $T(n)$-indistinguishable from a real execution.

Nevertheless, although standard proof techniques can not be applied when using definition 28, we argue that it is still a natural relaxation of $\mathcal{ZK}$:

1. Since we want to be protected against a verifier that attempts to extracts information that it will later try to use in a different protocol execution where the actual parties are only probabilistic polynomial-time machine, it seems natural to only require that the simulator's output is polynomial-time indistinguishable from a real executions.

2. Although a standard hybrid argument can not be used to show that Definition 28 is closed under sequential composition, we are able to show that this is the case for interactive proofs *with efficient prover* (which anyway are the interesting ones in cryptographic applications). In order to make the proof of this fact go through we need to rely on a slightly more refined argument than the one used to show that the notion of $\mathcal{ZK}$ proofs is closed under sequential composition (see [37]).

3. We show that the notion of $T(n)$-simulatability also encompasses the notion of $\mathcal{WI}$, namely a interactive proof is $\mathcal{WI}$ if and only if there exists a function $f(n)$ such that the interactive proof is $f(n)$-simulatable.

A Sequential Composition Lemma We show the robustness of Definition 28 by proving that the definition is closed under sequential composition, with respect to interactive proofs with *efficient provers*. Looking ahead, the proof of the sequential composition lemma will be used to facilitate the proof of concurrent security for one of our protocols.

Lemma 3 (Sequential Composition Lemma) *Let $T(n)$ be a class of functions that is closed under composition with any polynomial, and let (P, V) be a $T(n)$-simulatable interactive proof, with efficient prover, for the language $L \in \mathcal{NP}$. Let $Q(n)$ be a polynomial, and let (P_Q, V_Q) be an interactive proof that on common input $x \in \{0,1\}^n$ proceeds in $Q(n)$ phases, each on them consisting of executing the interactive proof (P, V) on common input x (each time with independent random coins). Then (P_Q, V_Q) is a $T(n)$-simulatable interactive proof.*

Proof: We start by noting that it follows directly from the construction that (P_Q, V_Q) is both sound and complete. Let us therefore turn to the $T(n)$-simulatability property. On a high level the proof follows the structure of the sequential composition lemma for $\mathcal{ZK}$ proofs of Goldreich and Oren [37]. We start by "partitioning" the malicious verifier V_Q^* into $Q(n)$ phases, each of which is the execution of a verifier for a "stand-alone" interactive proof (P, V), called V^*. The new stand-alone verifier V^* will communicate $Q(n)$ times with a real prover P, and we wish to show that V^* does not "learn" anything from these $Q(n)$ interactions. Towards this goal, we use the simulator S for V^* (which is guaranteed by the $T(n)$-simulatable property of (P, V)) and thereafter show that the output of $Q(n)$ execution of the simulator is indistinguishable of $Q(n)$ executions between a real prover and V^*. It is only at the last step of the proof that our proof differs from the proof of Goldreich and Oren. Whereas in the case of zero-knowledge proofs, the indistinguishability of $Q(n)$ executions of the simulator and $Q(n)$ real executions follows using a standard hybrid argument, the same does not hold in our case. The reason for this is that the argument of [37] relies on the fact that the distinguisher can incorporate (many) copies of the simulator. In the case of $T(n)$-simulatable proofs this is no longer the case since the simulator might run in super-polynomial time, whereas the distinguisher is only allowed to run in polynomial time. In order to overcome this issue, we instead rely on a careful orderings of the hybrid and on the fact that the (P, V) is an interactive proof with an efficient prover. We now proceed to a formal proof.

We rely on the following claim originally due to Goldreich and Oren [37], which shows that the execution of a "sequential" verifier V_Q^* can be partitioned into many sequential executions of a stand-alone verifier V^*.

Claim 1 (Claim 4.3.11.1 in [32]) *There exists a probabilistic polynomial-time machine V^* such that for every common input $x \in \{0,1\}^n$ and every auxiliary input z it holds that*

$$\langle P_Q, V_Q^*(z) \rangle(x) = Z_{Q(n)}$$

where $Z_0 = z$ and $Z_{i+1} = \langle P, V^*(Z_1)\rangle(x)$ for $i = 0, ..., Q(n) - 1$. Namely $Z_{Q(n)}$ is a random variable describing the output of V^* after $Q(n)$ successive interactions with P.

Let M be the simulator for the stand-alone verifier V^*. We define the simulator M_Q as a machine that on input x, z proceeds in $Q(n)$ phases. In the i'th phase M_Q computes $z_i = M(x, z_{i-1})$. After $Q(n)$ phases M_Q stops and outputs $z_{Q(n)}$. Note that the running time of M_Q is $Q(n)T(n) \in T(n)$.

It remains to show that the output of the simulator M_Q is indistinguishable from the output of the verifier V_Q in a real interaction with a prover.

Claim 2 *For every probabilistic machine D with running time polynomial in its first input, every polynomial $p(\cdot)$, all sufficiently long $x \in L$ and all $z, z' \in \{0,1\}^*$, we have*

$$|\Pr[D(x, z', \langle P_Q, V_Q^*(z)\rangle(x) = 1] - |\Pr[D(x, z', M_Q(x, z) = 1] < 1/p(|x|)$$

Proof Sketch: In order to prove the claim we define $Q(n)+1$ hybrids, $H_0, ..., H_{Q(n)}$. The j'th hybrid H_j is defined as the output of the following experiment:

- Let $z_0 = z$
- Run the simulator M_Q for j phases, i.e., for $i = 1$ to j let $z_i = M(x, z_{i-1})$.
- Thereafter let the honest prover communicate with the malicious verifier for the remaining $Q(n)-j$ phases, i.e., for $i = j$ to $Q(n)$ let $z_i = \langle P, V^*(z_{i-1})\rangle(x)$.
- Output $z_{Q(n)}$.

The claim consists of proving that hybrids H_0 and $H_{Q(n)}$ are computationally indistinguishable. Assume for contradiction that this is not the case. By a standard argument this implies that there are two consecutive hybrids distributions H_k, H_{k+1} that are distinguishable. Using an averaging argument it follows that there exists a z'_{k-1} such that the output of the following two experiments H'_k, H'_{k+1} are distinguishable.

- H'_k is defined as follows:
  1. let $z_k = \langle P, V^*(z'_{k-1})\rangle(x)$
  2. for $i = k+1$ to $Q(n)$ let $z_i = \langle P, V^*(z_{i-1})\rangle(x).(x)$.
  3. Output $z_{Q(n)}$.
- H'_{k+1} is defined as follows:
  1. let $z_{k+1} = M(z'_{k-1}, x)$
  2. for $i = k+1$ to $Q(n)$ let $z_i = \langle P, V^*(z_{i-1})\rangle(x)$.
  3. Output $z_{Q(n)}$.

Since Step 2 is defined in exactly the same way in both experiments, and since this step is efficiently computable given a witness $w \in R_L(x)$ (this follows from the efficient prover condition) it follows that following distributions are distinguishable (by a distinguisher that receives as auxiliary input the value z'_{k-1}, a witness $w \in R_L(x)$, and the auxiliary input needed to distinguish H'_k and H'_{k+1})

- $M(z'_{k-1}, x)$
- $\langle P, V^*(z'_{k-1})\rangle(x)$

which contradict the stand-alone $T(n)$-simulatability property of (P, V). ■

■

Witness Indistinguishability and $T(n)$-Simulatability We show that the definition of $T(n)$-simulatable proofs not only generalizes the notion of $\mathcal{ZK}$, it also encompasses the notion of $\mathcal{WI}$. Namely, an interactive proof is $\mathcal{WI}$ if and only if it is $f(n)$-simulatable for some function $f(n)$ (note that in particular $f(n)$ can be exponential).

Theorem 2 *Let (P, V) be an interactive proof (or argument) (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L. Then there exists a function $f(n)$ such that (P, V) is $f(n)$-simulatable if and only if (P, V) is $\mathcal{WI}$ for R_L.*

Proof: We show each implication separately.

Claim 3 *If (P, V) is $f(n)$-simulatable, then (P, V) is $\mathcal{WI}$.*

Proof: We wish to show that for every malicious verifier V^*, all sufficiently large instances $x \in L$, every witness pair $w_1, w_2 \in R_L(x)$ and every auxiliary input z to V^*, the following distributions are computationally indistinguishable:

- $\langle P(x, w_1), V^*(x, z)\rangle$
- $\langle P(x, w_2), V^*(x, z)\rangle$

Since (P, V) is $f(n)$-simulatable, there exist a simulator S such that the output of $S(x, z)$ is computationally indistinguishable from both $\langle P(x, w_1), V^*(x, z)\rangle$ and $\langle P(x, w_2), V^*(x, z)\rangle$. It follows using a standard hybrid argument that the distributions $\langle P(x, w_1), V^*(x, z)\rangle$ and $\langle P(x, w_2), V^*(x, z)\rangle$ are computationally indistinguishable. For completeness, we provide a formal proof.[4]

Suppose, for contradiction, that there exists a PPT machine V^*, a PPT distinguisher D and a non-negligible function $p(n)$ such that for infinitely many n

[4] Although in this particular instance, we are actually able to rely on "standard" hybrid arguments, in general such techniques can not always be used with $T(n)$-simulatable proofs (as was seen when demonstrating the sequential composition lemma). We therefore prefer to be extra careful and spell out the actual proof.

there exists an $x \in L \cap \{0,1\}^n$, two witnesses $w_1, w_2 \in R_L(x)$, and auxiliary inputs $z, z' \in \{0,1\}^*$ such that $D(z')$ distinguishes between the following distributions with distinguishing gap $p(n)$:

- $\{\langle P(w_1), V^*(z)\rangle)\}$
- $\{\langle P(w_2), V^*(z)\rangle)\}$

i.e.

$$|Pr[D(x, z', \langle P(w_1), V^*(z)\rangle(x))) = 1] \\ -Pr[D(x, z', \langle P(w_2), V^*(z)\rangle(x))) = 1]| > p(|x|)$$

Let S be the $f(n)$-simulator for (P, V). Now, consider the following expression:

$$|Pr[D(x, z', \langle P(w_1), V^*(z)\rangle(x))) = 1] - Pr[D(x, z', S(x, z)]| \\ +|Pr[D(x, z', S(x, z)] - Pr[D(x, z', \langle P(w^2), V^*(z)\rangle(x))) = 1]|$$

It follows from the fact that S is a simulator that the expression is negligible (since both the first and the second terms are negligible). On the other hand, by the triangle inequality, the expression is larger or equal to

$$|Pr[D(x, z', \langle P(w_1), V^*(z)\rangle(x))) = 1] \\ -Pr[D(x, z', \langle P(w_2), V^*(z)\rangle(x))) = 1]| > p(|x|)$$

The claim follows by contradiction. ■

Claim 4 *If (P, V) is $\mathcal{WI}$, then (P, V) is $f(n)$-simulatable.*

Proof: We show that there exists a black-box simulator S such that for every malicious V^*, all sufficiently large instances $x \in L$, every $w \in R_L(x)$ and every auxiliary input z to V^*, the following distributions are computationally indistinguishable:

- $\langle P(x, w), V^*(x, z)\rangle$
- $\langle S^{V^*(x,z)}(x, z)\rangle$

The simulator S simply finds a witness w' such that $w' \in R_L(x)$ (since L is in NP, this can be done in time $2^{poly(n)}$) and thereafter plays the role of the honest prover using this witness. It follows from the $\mathcal{WI}$ property of (P, V) that the output of the simulator is indistinguishable from the output of V^* in a real execution with a prover (since the simulator actually acts identically to the honest prover, except for the fact that it might use a different witness). ■

■

Remark 5 *We note that for the case of interactive proofs with* efficient provers *the above proof actually shows equivalence between $\mathcal{WI}$ and $2^{poly(n)}$-simulatability.*

3.2.2 Strongly $T(n)$-simulatable proofs

As we have seen when proving the sequential composition lemma (see Lemma 3) standard hybrid arguments can not be used with the notion of $T(n)$-simulatability. The reason for this is that in hybrid arguments one constructs a distinguisher that itself needs to execute the simulator. Since Definition 28 only requires that the output of the simulator is indistinguishable by a polynomial-time distinguisher, the distinguisher might not be able to execute the simulator (recall that the simulator might have a super-polynomial running time). In order to remedy this problem, we therefore introduce a more robust notion, called *strong $T(n)$-simulatability*. The definition of strong $T(n)$-simulatability requires the output of the simulator to be strongly $T(n)$-indistinguishable (instead of only polynomial-time indistinguishable as in Definition 28). To make the definition even stronger we furthermore allow the verifier to run in time $T(n)$. Looking ahead, this extra strengthening will allow us to prove a concurrent composition theorem for a certain type of strongly $T(n)$-simulatable proof (namely straight-line simulatable proofs, see Lemma 5).

Definition 29 (Strong $T(n)$-simulatability) *Let $T(n)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive proof (or argument) (P,V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is strongly $T(n)$-simulatable if there for every probabilistic machine V^* with running time bounded by $T(n)$ exists a probabilistic simulator S with running time bounded by $T(n)$ such that the following two ensembles are strongly $T(n)$-indistinguishable (when the distinguishing gap is a function in $n = |x|$)*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary $y \in R_L(x)$*
- $\{S(x,z)\}_{z\in\{0,1\}^*, x\in L}$

That is, for every probabilistic algorithm D running in time $T(\cdot)$ in the length of its first input, all sufficiently long $x \in L$, all $y \in R_L(x)$ and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$|Pr[D(x, z', (\langle P(y), V^*(z)\rangle(x))) = 1] - Pr[D(x, z', S(x,z)) = 1]| < \frac{1}{(T(|x|))}$$

Remark 6 *Note that since $T(n)$ is a class of functions that is closed under composition with any polynomial, the running time of the distinguisher (as well as the distinguishing gap) can be a $poly(f(n))$, where $f(n)$ is the running time of the simulator.*

Sequential Composition of Strongly Simulatable Proofs Whereas we had to restrict our attention to proofs with *efficient provers* in order to prove a sequential composition lemma for $T(n)$-simulatable proofs, for the case of strongly $T(n)$-simulatable proofs this is no longer needed. In fact, the proof of the sequential composition lemma for $\mathcal{ZK}$ of Goldreich and Oren [37] can be used unchanged.

Lemma 4 (Sequential Composition Lemma for Strong $T(n)$-Simulatability) *Let $T(n)$ be a class of functions that is closed under composition with any polynomial, and let (P, V) be a strongly $T(n)$-simulatable interactive proof for the language $L \in \mathcal{NP}$. Let $Q(n)$ be a polynomial, and let (P_Q, V_Q) be an interactive proof that on common input $x \in \{0,1\}^n$ proceeds in $Q(n)$ phases, each on them consisting of executing the interactive proof (P, V) on common input x (each time with independent random coins). Then (P_Q, V_Q) is a strongly $T(n)$-simulatable interactive proof.*

Proof: The proof of Goldreich and Oren [37] directly extends to yield the lemma. ■

Perfect $T(n)$-simulatability In analogy with the notion of perfect $\mathcal{ZK}$, we can further strengthen the notion of $T(n)$-simulatability to require that the output of the simulator is identically distributed to the output of the real execution. In out treatment, we furthermore let the verifier be a computationally unbounded machine.

Definition 30 (Perfect $T(n)$-simulatability) *Let $T(n)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive proof (or argument) (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is* perfectly $T(n)$-simulatable *if (P, V) satisfies Definition 28 and the two ensembles defined in Definition 28 (the simulated execution and the real execution) are identically distributed for every (computationally unbounded) machine V^*.*

We note that, trivially, perfect $T(n)$-simulatability implies strong $T(n)$-simulatability.

Security in the On-line/Off-line model We mention that although the main motivation behind the notion of strong $T(n)$-simulatability is to obtain a "better-behaved" variant of $T(n)$-simulatability, the definition itself also has direct applications.

In fact, in many natural situations it seems reasonable to assume that parties have certain "on-line" running time, but a longer "off-line" time.[5] One way of formalizing such an *on-line/off-line model* would be by modeling the parties as polynomial time machine during on-line communications, while machines with running time $T(n)$ when off-line.

We note that in such a setting, strongly $T(n)$-simulatable arguments are in fact *zero-knowledge*. In particular, strong $T(n)$-simulatability, means that the a verifier does not learn anything that it could not have learned by itself when being off-line. Note that the notion of simply $T(n)$-simulatability is not necessary to guarantee zero-knowledge in the on-line/off-line model. This follows from the fact

[5] In real-life protocol executions parties that do not respond withing a specified time would be "timed-out".

that $T(n)$-simulatability only guarantees that the output of the simulator is indistinguishable from the real execution, by an *on-line* distinguisher (and not by an off-line distinguisher).

3.2.3 Straight-line Simulation

In the concurrent setting it is often helpful to be able to perform simulation without rewinding, i.e. a so called *straight-line* simulation. Towards the goal of obtaining a definition which is robust also in the concurrent setting we restrict Definition 28 to straight-line simulators:

Definition 31 *Let $T(n)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive argument (proof) (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is straight-line $T(n)$-simulatable if there for every PPT machine V^* exists a probabilistic simulator S with running time bounded by $T(n)$ such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $n = |x|$)*

- $\{(\langle P(y), V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary $y \in R_L(x)$*
- $\{(\langle S, V^*(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$

We note that the above definition is very restrictive. In fact, the simulator is supposed to act a cheating prover, with its only advantage being the possibility of running in time $T(n)$, instead of in polynomial time. Trivially, there therefore do not exist any straight-line $T(n)$-simulatable *proofs* for non-trivial languages (this should be contrasted with straight-line simulatable interactive *arguments*, which we show do exist).

3.2.4 Concurrent Composition

In analogy with the notion of concurrent zero-knowledge, we further restrict Definition 31 to guarantee security under concurrent executions:

Definition 32 *Let $T(n)$ be a class of functions that is closed under composition with any polynomial. We say that an interactive argument (proof) (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is straight-line concurrent $T(n)$-simulatable if there for every PPT oracle machine A that is not allowed to restart or rewind the oracle it has access to, and every polynomial $g(n)$, there exists a probabilistic simulator $S(i, x)$ with running time bounded by $T(n)$ such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $n = |x|$)*

- $\{A^{P(x_1,y_1),P(x_2,y_2),..P(x_{g(n)},y_{g(n)})}(z, x_1, x_2, .., x_{g(n)})\}_{z\in\{0,1\}^*, x_1,x_2,..,x_{g(n)}\in L}$ *for arbitrary $y_i \in R_L(x_i)$*
- $\{A^{S(1,x_1),S(2,x_2),..S(g(n),x_{g(n)})}(z, x_1, x_2, .., x_{g(n)})\}_{z\in\{0,1\}^*, x_1,x_2,..,x_{g(n)}\in L}$

Looking ahead, in section 3.3 we formally show that the notion of straight-line concurrent $T(n)$-simulatability is very robust in the concurrent setting. More precisely, we show that protocols satisfying this notion not only retain their security properties when simultaneously executing multiple sessions of the *same* protocol (i.e. self-composition), but also if the protocols are concurrently composed with a large class of *other* protocols (i.e., general composition).[6]

A Concurrent (Self)-Composition Lemma Whereas we could only show that the notion of $T(n)$-simulatability is closed under sequential composition, we are able to give a concurrent composition lemma for *straight-line strongly $T(n)$-simulatable* proofs.

Lemma 5 (Concurrent Self-Composition Lemma) *Let $T(n)$ be a class of functions that is closed under composition with any polynomial, and let (P, V) be an interactive proof (argument) with efficient provers. If (P, V) is straight-line strongly $T(n)$-simulatable (or perfectly simulatable), then it is also straight-line concurrent $T(n)$-strongly simulatable (or perfectly simulatable).*

Proof of Lemma 5 Let S be the straight-line simulator for (P, V). Construct the straight-line concurrent simulator $S'(i, x) = S(x)$. Clearly, since the running time of S is bounded by $T(n)$, so is the running time of S'. We continue by showing the output of the simulator is correctly distributed, i.e., that for every PPT oracle machine A that is not allowed to restart or rewind the oracle it has access to, every polynomial $g(n)$, every $x_1, x_2, .., x_{g(n)} \in L$, every $y_i \in R_L(x_i)$, and every $z \in \{0,1\}^*$ the following distributions are strongly $T(n)$-indistinguishable:

1. $\texttt{real} = \{A^{P(x_1,y_1),P(x_2,y_2),..P(x_{g(n)},y_{g(n)})}(z, x_1, x_2, .., x_{g(n)})\}$
2. $\texttt{simulated} = \{A^{S'(1,x_1),S'(2,x_2),..S'(g(n),x_{g(n)})}(z, x_1, x_2, .., x_{g(n)})\}$

Since both the prover strategy P, and the simulator S' have a running time that is bounded by $T(n)$, it follows using a standard hybrid argument that the ensembles `real` and `simulated` are strongly $T(n)$-indistinguishable. For completeness we provide a sketch.

Construct the hybrid distributions $H_0, ..., H_{g(n)}$, where

- $H_i = \{A^{P(x_1,y_1),...,P(x_i,y_i),S'(i+1,x_{i+1},...S'(g(n),x_{g(n)})}(z, x_1, x_2, .., x_{g(n)})\}$

In other words, H_i is the output distribution of the adversary A having access to the real prover for sessions 1 to i, and access to the simulator for the remaining sessions. Note that H_0 is identically distributed to the distribution `real`, while $H_{g(n)}$ is identically distributed to the distribution `simulated`.

[6] We note that this is, for example, not true for concurrent zero-knowledge proofs. Such proofs only retain the zero-knowledge under concurrent *self-composition*, i.e., when concurrently running multiple executions of the *same* protocol. In fact, it is the straight-line simulatability that lets us achieve this strong property.

Suppose, for contradiction that the distributions `real` and `simulated` are distinguishable, i.e. that H_0 and $H_{g(n)}$ are distinguishable. Using the triangle inequality, it follows that there exists an index j such that the distributions H_j and H_{j+1} are distinguishable. We show that this contradicts the stand-alone strong $T(n)$-simulatability of (P, V), by constructing a stand-alone adversary A'. A' gets the witnesses $y_1, ..., y_{i-1}$, and z as its auxiliary input. A' incorporates A and proceeds as follows:

- A' internally emulates messages that are part of session 1 to $i-1$ for A by playing the role of the honest prover P. Note that since A' gets the witnesses $y_1, ..., y_{i-1}$ as part of its auxiliary input it can perform this task in polynomial time.
- A' internally emulates messages that are part of session $i+1$ to $g(n)$ for A by running the simulator S' to generate the interaction. Note that since A' is allowed to run in time $poly(T(n))$ it can indeed run the simulator S'.
- A' externally forwards messages that are part of session i.

Note that the output of A' after interaction with an honest prover is identically distributed to H_j. Likewise, the output of A' after interaction with the simulator S is identically distributed to H_{j+1}. But by our assumptions these distributions are distinguishable, which contradicts the fact that S is a straight-line simulator for (P, V). ■

Remark 7 *1. We note that the same proof can not be applied in the case of simply simulatable proofs, as the stand-alone verifier constructed in the hybrid argument has a running time of $poly(T(n))$.*

2. It is interesting to note that the notions of straight-line strong $T(n)$-simulatability is strictly stronger than the notion of straight-line concurrent $T(n)$-simulatability. In fact, the protocol in section 3.5 constitutes an evidence of this fact. See remark 11.

3.3 A Concurrent General Composition Theorem

In this section we formally show that the notion of straight-line concurrent $T(n)$-simulatability not only provides security under self-composition (such as the notion of concurrent $\mathcal{ZK}$); it also provides security in under a more general type of composition operation.[7] We formalize this feature through a composition theorem, which loosely speaking states that for a large class of natural protocols the security of these protocols is not affected when concurrently executed with multiple straight-line concurrent $T(n)$-simulatable arguments. In other words, we show that

[7] Note that since the notion of straight-line strong $T(n)$ is strictly stronger, the same result hold also for this notion.

straight-line concurrent $T(n)$-simulatable protocols not only retain their security properties when concurrently composed with themselves, but also when concurrently composed with a large class of other protocols.

We start by presenting the problem that we wish to address more carefully and thereafter turn to show the composition theorem.

The Problem of Concurrent General Composition Suppose that we have a cryptographic system (which we call an environment) that an adversary is trying to break. Suppose further that we are able to prove (or just strongly believe) that a stand-alone adversary is not be able to break the system; we say that such a system constitutes a *hard* environment. We would now like to identify a class of, so called "safe" protocols, which have the property that an adversary that is allowed to participate in (possibly many) safe protocol, will still not be able to break *any* hard environment. We separate three different "attack" scenarios:

1. **Sequential/Sequential** The attack proceeds in two stages. In the first stage the attacker is allowed *sequential* access to the "safe" protocols. When the first stage is completed the attacker then attempts to break the environment. It can be seen that in this scenario zero-knowledge proofs are "safe" (since the zero-knowledge property is closed under sequential composition [37]).

2. **Concurrent/Sequential** Also here the attack proceeds in two stages. In the first stage the attacker is allowed *concurrent* access to the "safe" protocols. When the first stage is completed the attacker then attempts to break the environment. Note that in the second stage only one protocol execution is taking place, namely the attacker is communicating with the environment. In this scenario *concurrent* zero-knowledge proofs constitute "safe" protocols.

3. **Concurrent/Concurrent** The most powerful attack scenario proceeds as follows. The attacker is allowed *concurrent* access to the "safe" protocols, and is *at the same time* allowed to communicate with the environment in order to attempt to break it.

In this section we attempt to identify a notion of "safe" protocols for the third, and most demanding, setting. Note that the notion of concurrent zero-knowledge is not sufficient to guarantee security in this scenario. This follows from the fact that the notion of concurrent zero-knowledge only guarantees security when there are many executions of the *same* protocol, while in our scenario the attacker might simultaneously participate in (at least) two different protocols, the "safe" protocols, and the protocol it is executing with the environment (which might be any arbitrary protocol).

On the impossibility of "safe" protocol In fact, it can be shown that the problem can not be resolved in its most general form. More precisely, it can be seen that there do not exist any proof of knowledge protocols that are "safe" for *any* hard environment (according to the most demanding attack). We provide a sketch.

Assume that there exists a "safe" proof of knowledge for a hard-instance language. Now, consider the environment consisting of the verifier of the "safe" protocol for a randomly generated (hard) instance. Clearly, a stand-alone adversary (that does not have the witness for this instance) will not be able to break such an environment. However, a so called man-in-the-middle adversary, that is simultaneously participating as a verifier in a different execution of the same interactive proof will, by simply forwarding messages between the environment and the prover it is communicating with, succeed in breaking the environment, which thus contradicts that the interactive proof was "safe" for any environment.

We conclude that it is impossible to find protocols that are "safe" for *every* environment. Nevertheless, it might still be possible to find protocols that are "safe" for a restricted class of environments.

"Safe" protocols for restricted environments Recently a solution to this problem was proposed by Canetti and Fischlin [13]. They show how to construct a zero-knowledge protocol in the Common Reference String (CRS) model, which is "safe" against any environment that is unrelated to the CRS string used by the "safe" protocols. In this section, we, instead, show an alternative solution to this problem, *without resorting to set-up assumptions*, such as that of a CRS string. More precisely, we identify a class of protocols that are "safe" against any environments that is hard for adversaries with running time $T(n)$.[8] In fact, we show that *straight-line concurrent $T(n)$-simulatable arguments* are "safe" for any environment that is hard for time $T(n)$.

The Composition Theorem

We start by formally defining the notion of an environment. We see an environment as a system that an adversary is trying to break. The environment outputs 1 if the adversary succeeds and 0 otherwise. Intuitively, we say that an environment is hard if an adversary can not make the environment output 1, i.e., break the environment. More formally,

Definition 33 *We say that an interactive PPT machine E, called environment, is hard for the language L, the generator Gen_L, and $T(n)$-adversaries, if for all interactive probabilistic machine A with running time bounded by $T(n)$, every $z \in \{0,1\}^{poly(n)}$*

$$Pr[x \leftarrow Gen_L, \langle A(z), E\rangle(x) = 1]$$

is negligible as a function of n, where Gen_L is a sampling machine that chooses an element $x \in L \cap \{0,1\}^n$ according to some distribution.

[8] For simplicity, think of $T(n)$ as being quasi-polynomial. This means means that instead of guaranteeing security against all environments (i.e. environments hard for polynomial-time), we guarantee security only against environment hard for quasi-polynomial time.

Our composition theorem states that a PPT adversary, that is allowed concurrent access to different provers, communicating *only* using straight-line concurrent $T(n)$-simulatable interactive arguments will not be able to break any environment that is hard for $T(n)$-adversaries (even if the adversary has simultaneous access to both the provers and the environment).

More formally,

Theorem 3 *Let $T(n) \subseteq n^{\omega(1)}$ be a class of functions that is closed under composition with any polynomial, and let (P,V) be a straight-line concurrent $T(n)$-simulatable interactive argument for the language L. Let p(n) be a polynomial, $Gen_{L'^{p(n)}}$ be a generator for $(L \cap \{0,1\}^n)^{p(n)}$, i.e., $Gen_{L'^{p(n)}}$ is a sampling machine that chooses an element $(x_1, x_2, ..x_{p(n)}) \in (L \cap \{0,1\}^n)^{p(n)}$ according to some distribution. Then, for every environment E that is hard for the language $L^{p(n)}$, the generator $Gen_{L^{p(n)}}$ and $T(n)$-adversaries, and every PPT oracle machine A (called the adversary), that can not restart or rewind the oracle it gets access to, and every auxiliary input $z \in \{0,1\}^{poly(n)}$, the following expression is negligible (as a function of n).*

$$Pr[\bar{x} = (x_1, x_2, .., x_{p(n)}) \leftarrow Gen_{L^{p}(n)}, y_i \in R_L(x_i),$$
$$\langle A^{P(x_1,y_1),P(x_2,y_2),..P(x_{g(n)},y_{g(n)})}(z), E\rangle(\bar{x}) = 1]$$

Intuitively, the theorem follows from the fact that a straight-line simulator is a "cheating" prover running in time $T(n)$. If an adversary would succeed in breaking a specific environment, then an adversary with access a straight-line simulator running in time $T(n)$, instead of the real provers, would succeed as well. More formally,

Proof of Theorem 3 Let Π, $T(n)$, L, $p(n)$, $Gen_{L^{p(n)}}$, E, and A be defined as in the theorem. Assume, for contradiction, that for infinitely many n, there exist a $z \in \{0,1\}^{poly(n)}$ such that

$$Pr[\bar{x} = (x_1, x_2, .., x_{p(n)}) \leftarrow Gen_{L^{p}(n)}, \langle A^{P(x_1),P(x_2),..,P(x_{p(n)})}(z), E\rangle(\bar{x}) = 1]$$

is non-negligible, i.e., the adversary A succeeds in breaking the environment, when receiving proofs of the statements $x_1, .., x_n$. Let $S(i,x)$ be the straight-line concurrent simulator for $p(n)$ concurrent sessions of Π. It follows from the fact that E is a hard for $T(n)$-adversaries (and since the class $T(n)$ is closed under composition with any polynomial) that the following expression is negligible,

$$Pr[\bar{x} = (x_1, x_2, .., x_{p(n)}) \leftarrow Gen_{L^{p}(n)}, \langle A^{S(1,x_1),S(2,x_2),..,S(p(n),x_{p(n)})}(z), E\rangle(\bar{x}) = 1]$$

In other words, the adversary A does not succeed in breaking the environment when receiving simulated proofs, instead of real proofs. Thus, intuitively this fact contradicts the straight-line concurrent $T(n)$-simulatability of the interactive proof (P,V). In order to formalize this intuition we need to show that for infinitely many n there exist instances $x_1, ..., x_{p(n)}$ for which A succeeds in breaking the environment with non-negligible probability when receiving real proofs of these instances, but fails (i.e. only succeeds with negligible probability) when receiving simulated proofs. We call such instances `good`.

We start by showing that the existence of `good` instances for infinitely many n contradicts the fact that S is a straight-line concurrent $T(n)$-simulator for (P, V). Towards this goal, we define a new (concurrent) malicious verifier A' which is obtained by combining the machine A and E, and outputting what E would have output. Note that since both A and E are polynomial-time, A' is so as well. By our assumptions it follows that for infinitely many n there exists instances $\bar{x} = x_1, ..., x_{p(n)}$ for which the following holds:

- $A(\bar{x})$ outputs 1 with non-negligible probability after receiving real proofs of the statements $x_1, ..., x_{p(n)}$
- $A(\bar{x})$ output 1 only with negligible probability when receiving simulated proofs of the same statements.

We conclude that S is not a valid straight-line concurrent simulator for (P, V), which contradicts our assumptions.

It thus only remains to show that there exist `good` instances for infinitely many n. Assume, for contradiction, that this is not the case. Since for infinitely many n it holds that, with non-negligible probability the, instance $\bar{x} = (x_1, x_2, .., x_{p(n)}) \leftarrow Gen_{L^{p}(n)}$ has the property that

$$Pr[\langle A^{P(x_1), P(x_2), .., P(x_{p(n)})}(z), E\rangle(\bar{x}) = 1]$$

is non-negligible, it must also hold that for infinitely many n with non-negligible probability the instance $\bar{x} = (x_1, x_2, .., x_{p(n)}) \leftarrow Gen_{L^{p}(n)}$ has the property that

$$Pr[\langle A^{S(1,x_1), S(2,x_2), .., S(p(n), x_{p(n)})}(z), E\rangle(\bar{x}) = 1]$$

is non-negligible, which contradicts the fact that E is hard for $T(n)$-adversaries. ■

Remark 8 *We note that since the environment is a polynomial time machine, it is sufficient that the arguments that the adversary is allowed to participate in are simply simulatable and not strongly simulatable, i.e., that the simulator's output is only required to be polynomial time indistinguishable from a real interaction (instead of $T(n)$-indistinguishable).*

The theorem shows that straight-line $T(n)$-simulation is a sufficient condition for security when integrating a sub-protocol in an environment that is hard for $T(n)$-adversaries. This yields an efficient way of constructing protocols with strong security properties by the use of telescopic composition of protocols, i.e., using protocols that are successively harder and harder. Indeed, the key to the theorem is the fact that the interactive arguments, that the adversary is receiving, are *easy* for "break" in time $T(n)$ (since they are *straight-line* $T(n)$-simulatable) while the environment is hard for $T(n)$-adversaries.

3.4 An Efficient Perfectly Simulatable Argument

In this section we show how to construct an efficient protocol satisfying the requirements needed for the composition theorem. More precisely, we construct a constant-round straight-line $n^{poly(logn)}$-perfectly simulatable argument. The protocol is thus also zero-knowledge in the on-line/off-line model (see section 3.2.2). We also mention that the protocol can be constructed through an efficient transformation from any 3-round special-sound public-coin honest-verifier perfect zero-knowledge (HVPZK) argument.

On a high-level, the protocol builds on the Feige-Lapidot-Shamir construction [28]: The protocol consists of a $\mathcal{WI}$ proof (argument) of knowledge of the fact that *either* the prover has a witness to the statement to be proved, *or* it has a, so called, "fake" witness to some other (bogus) statement. This is done is such a way that it is hard for a (possibly malicious) prover to find the "fake" witness, but the simulator can (using its extra power) find such a witness.

Our instantiation of the above paradigm is as follows: The verifier starts by sending a random image $c = f(r)$ through a one-way function f with subexponential security. The prover thereafter proves that he either has a witness to the statement x (to be proved) or that he has a pre-image to c (for the function f). The size of c is chosen in such a way that a polynomial time malicious prover will not be able to find a pre-image to c, but one *can* can be found in quasi-polynomial time by performing an exhaustive search. Now, intuitively, the soundness of the protocol follows from the proof of knowledge property. The simulator, on the other hand, is able to find a pre-image and can thus use it as a "fake" witness. We proceed to a more formal description of the protocol and thereafter discuss how to implement it efficiently.

The Protocol Let f be function that is one-way for time 2^{n^κ}. Furthermore, assume that f has an efficiently recognizable range (i.e., given an input y it is "easy" to check whether there exist a value x such that $f(x) = y$; note that, for example, permutations have this property). Let the witness relation $R_{L'}$, where $(x, y) \in R_{L'}$ if $f(x) = y$, characterize the language L'.

Let the language $L \in \mathcal{NP}$, and $k = \frac{1}{\kappa} + 1$. Consider the following protocol for proving that $x \in L$:

Protocol Π - A 4 Round Perfectly Simulatable Argument for $\mathcal{NP}$

Common Input: an instance x of a language L with witness relation R_L, 1^n: security parameter.

Stage 1:

V uniformly chooses $r \in \{0,1\}^{log^k n}$.

V $\rightarrow$ P: $c = f(r)$.

Stage 2:

P checks whether the value c is in the range of f. If it is not, P halts, outputting reject.

P $\leftrightarrow$ V: a witness independent argument of knowledge of the statement

either there exists a value r' s.t $c = f(r')$

or $x \in L$

The argument of knowledge is with respect to the witness relation $R_{L \vee L'}(c, x) = \{(r', w) | r' \in R_{L'}(c) \vee w \in R_L(x)\}$.

We start by showing that Π is stand-alone perfectly quasi-polynomial time simulatable.

Proposition 1 *The protocol Π is a straight-line $n^{O(log^k n)}$-perfectly simulatable argument of knowledge.*

Proof of Proposition 1 Completeness follows from the completeness of the witness independent argument used in Stage 2.

Soundness/Proof of knowledge We continue to show that Π is an proof of knowledge; this directly implies that the protocol is sound.

The claim follows from the fact that the argument system used in Stage 2 is an proof of knowledge, and the fact that a PPT adversary only finds a pre-image to c (for f) with negligible probability. More formally, we construct an extractor machine E for every malicious prover P^* for the protocol Π. E internally incorporates P^* and proceeds as follows. E generates the first verifier message c honestly and feeds it to P^*. It then invokes the extractor E' for the argument system in stage 2 of protocol Π. E finally outputs whatever E' outputs. By the proof of knowledge property of the argument system in stage 2, it follows that with probability negligibly close to the success probability of P^*, the output of E will either be a witness to the statement proved, or the pre-image of c.

Since f is a one-way for subexponential circuits, probabilistic non-uniform adversaries with running time bounded by $2^{(log^k n)^\kappa} = n^{log^\kappa n}$ only have a negligible probability of finding a pre-image to c. Thus, E (which only has a polynomial running time) will only output a pre-image to c with negligible probability.[9] We conclude that E outputs a witness to the statement proved with probability negligibly close to the success probability of the malicious prover.

Zero-knowledge Let us turn to zero-knowledge. Consider a straight-line simulator S that proceeds as the honest prover until the witness independent argument in stage 2 of the protocol is reached, i.e., S receives the values c and checks that c indeed has a pre-image. S thereafter performs an exhaustive search to find a pre-image r of the value c for the function f (i.e., a value r' such that $f(r) = c$). To perform this task S tries all possible values $r'' \in \{0,1\}^{log^k n}$ and checks if $f(r'') = c$. This thus takes time $poly(2^{log^k n})$, since the time it takes to evaluate the function f is a polynomial in n. After having found a value r' such that $f(r') = c$, S uses r' as a witness in the witness independent argument (instead of using a real witness to the statement x, as the honest prover would do).

Clearly the running time of S is bounded by $n^{O(log^k n)}$. We proceed to show that the output of the simulator is identically distributed to the output of the verifier in a real execution with an honest prover. Note that the only difference between a real execution and a simulated execution is in the choice of witness used in stage 2 of the protocol. It thus follows from the witness independence property of the argument system in stage 2 that the output of the simulated execution is identically distributed to the output of the real execution. ■

Remark 9 *1. Note that the above proof relies on the fact that Witness Independence is defined for unbounded verifiers.*

2. Also note that in order to construct this, and the subsequent protocols in this chapter which rely on subexponential hardness assumption, the "hardness" constant κ of the one-way function needs to be known.

Concurrent simulatability follows directly, since the protocol is strongly simulatable.

Proposition 2 *The protocol Π is a straight-line concurrent $n^{O(log^k n)}$-perfectly simulatable argument.*

Proof of Proposition 2 Follows directly by combining Proposition 1 and Lemma 5. ■

[9] To prove this formally, consider a different extractor E'' that gets the values c as input instead of generating it honestly. If E'' is fed an honestly generated value c, the output of E will be identical to the output of E''. This means that if E succeeds in finding a pre-image to the values c with non-negligible probability, then E'' inverts the one-way function f with non-negligible probability.

We mention that constant-round witness independent arguments of knowledge can be constructed based on the existence of constant-round perfectly hiding commitments (consider for example Blum's proof for the Hamiltonian Cycle problem [8], using perfectly hiding commitments). We thus obtain the following theorem.

Theorem 4 *Assume the existence of one-way functions secure for subexponential circuits, with efficient recognizable range, and the existence of constant-round perfectly hiding commitments. Then there exist a constant-round interactive argument of knowledge that is straight-line concurrent $n^{poly(log n)}$-perfectly simulatable.*

Remark 10

1. *We note that if we use 2-round perfectly hiding commitments (such commitment schemes exist under the assumption of claw-free collections [34]), then the resulting protocol uses only 4 communication rounds.*
2. *Using a standard "trick" the efficiently recognizable range condition on the function f can be removed by letting the verifier perform a zero-knowledge (witness hiding) proof showing that the value c has a pre-image. This solution, however, requires a higher round-complexity.*

Efficient Implementation In this section we show how to give an efficient instantiation of the protocol. In fact, we show how to implement a 4-round version of our protocol, through an efficient transformation from any 3-round (or 4-round) special-sound public-coin honest verifier perfect zero-knowledge (HVPZK) proof. Assume that we have a 3-round special-sound public-coin HVPZK argument for the language L. Choose the function f and thus the language L' with witness relations and $R_{L'}$ such that there exist an efficient 3-round special-sound public-coin HVPZK proof for the language L with witness relation R'_L (examples of such protocols are the Guillou-Quisquater scheme [41] for the RSA function, and the Schnorr scheme [63] for the discrete logarithm).

Now, to implement the witness independent proof we start by noting that HVPZK arguments are witness independent [19]. We can thereafter combine the argument system for L and the argument system for L' using the efficient OR-transformation of [19] yielding a special-sound public-coin HVPZK proof for $L \vee L'$ with the witness relation $R_{L \vee L'}$.[10]

3.5 A Two Round Simulatable Argument

In this section we construct a 2-round straight-line concurrent $n^{poly(log n)}$-simulatable argument for $\mathcal{NP}$. The protocol shows that all the impossibility results regarding the zero-knowledge definitions mentioned in section 3.1 can be overcome if relaxing

[10] Since the transformation in [19] uses that the second messages of the two protocols have the same length, we need to run several parallel versions of the protocol for L'. The resulting argument then uses less communication than the argument for L plus the (parallelized) argument for L'.

the definition to allow the simulator to run in quasi-polynomial time instead of polynomial time.

On a high level, the idea behind the protocol is very similar to the protocol described in section 3.4 with the main difference that a zap (i.e., a two-round $\mathcal{WI}$ proof) is used instead of a three-round witness indistinguishable argument of knowledge. Since the "compressed" protocol, using a zap, no longer is a proof of knowledge, we rely on the complexity leveraging technique of [14] to ensure the soundness of the protocol. This is done by the use of, so called, "extractable" non-interactive commitments. We start by defining extractable commitments and show how such commitment schemes can be constructed. We thereafter present the full protocol.

3.5.1 Extractable Commitments under General Assumptions

Roughly speaking, we say that a commitment scheme is $T(n)$-extractable, if the value committed to can be extracted in time $T(n)$ (although the commitment scheme is hiding for polynomial time). For simplicity we only state a definition for non-interactive perfectly-binding commitment schemes.

Definition 34 *We say that a non-interactive perfectly-binding commitment scheme Com is $T(n)$-extractable if there exists a probabilistic extractor machine E, with running time bounded by $T(n)$, such that for all messages $c = Com(x;r)$, $E(c) = (x,r)$ with overwhelming probability.*

We show how to construct a non-interactive $n^{poly(logn)}$-extractable commitment scheme, based on the existence of one-to-one one-way functions secure for subexponential circuits. More specifically, we construct a $n^{poly(logn)}$-extractable bit commitment scheme using a modified version of Blum's commitments scheme [8]. The idea behind the construction is to create commitments that are "large" enough to guarantee the hiding property against polynomial time adversaries, but "small" enough for a quasi-polynomial time adversary to be able to perform an exhaustive search and thus extracting the committed value. We obtain such commitment schemes by "scaling" down the security parameter in Blum's commitment scheme.

The Construction Let $f : \{0,1\}^* \rightarrow \{0,1\}^*$ be a function, and let $b : \{0,1\}^* \rightarrow \{0,1\}$ be a predicate.

Protocol *Com* **- A Non-interactive Extractable Commitment Scheme**

Commit Phase:

- To commit to bit $v \in \{0,1\}$, the sender uniformly selects $s \in \{0,1\}^{log^k n}$ and sends the pair $(f(s), b(s) \oplus v)$.

Reveal Phase:

- The sender reveals v and s.
- The receiver accepts if $f(s) = \alpha$ and $b(s) \oplus v = \beta$ where (α, β) is the receiver's view of the commit phase.

Proposition 3 *Let $f : \{0,1\}^* \rightarrow \{0,1\}^*$ be a one-to-one function that is one-way for subexponential circuits, and let $b : \{0,1\}^* \rightarrow \{0,1\}$ be a hard-core predicate for subexponential circuits for f, i.e., there exists a κ such that b is a hardcore for probabilistic non-uniform adversaries with running time bounded by 2^{n^κ}. Then for $k = \frac{1}{\kappa} + 1$, the protocol Com constitutes a $n^{O(log^k n)}$-extractable bit-commitment scheme.*

Proof of Proposition 3 We start by showing that the protocol indeed constitutes a bit-commitment scheme. The hiding property follows from the fact that b is a hard-core predicate for subexponential circuits. That is, since s is a string of length $log^k n$, all probabilistic non-uniform adversaries with running time bounded by $2^{(log^k n)^\kappa} = 2^{log^{1+\kappa} n} = n^{log^\kappa n}$ have negligible probability of predicting $b(s)$ given $f(s)$ with probability that is non-negligibly higher than $\frac{1}{2}$. (In fact every $n^{log^\kappa n}$ adversary has a probability smaller than $\frac{1}{2} + n^{log^\kappa n}$ of predicting $b(s)$.) Since $n^{log^\kappa n}$ is asymptotically greater than any polynomial, this thus also holds for PPT non-uniform adversaries.

The binding property follows trivially from the one-to-one property of f.

Now, let us turn to extractability. We show how to construct an extractor machine E, with running time bounded by $n^{O(log^k n)}$. Upon receiving a value $c = (\alpha, \beta)$, E proceeds as follows:

- E performs an exhaustive search to find the pre-image s of α for the function f (i.e., a values s such that $\alpha = f(s)$). To perform this task E tries all possible values $s' \in \{0,1\}^{log^k n}$ and checks if $f(s') = \alpha$. This thus takes time $poly(2^{log^k n})$, since the time it takes to evaluate the function f is a polynomial in n.
- E thereafter computes the bit $v = \beta \oplus b(s)$.

Clearly, the running time of E is bounded by $n^{log^k n}$. Furthermore, since f is a one-to-one function there only exists one possible decommitment for every commitment c. Thus, the value extracted by E will be the correct value with probability 1. ■

3.5.2 The Protocol

Having constructed extractable commitments, we are now ready to present the two-round protocol.

Suppose that $f : \{0,1\}^n \rightarrow \{0,1\}^n$ is a one-to-one one-way function secure against adversaries running in time $2^{n^{\kappa}}$. Let Com be a commitment scheme extractable in time $n^{log^{k'} n}$ and let $k = \frac{1}{\kappa} + \frac{2k'}{\kappa}$. Consider the following protocol:

Protocol Π - A 2 Round Quasi-Poly Simulatable Argument for $\mathcal{NP}$

Common Input: an instance x of a language L with witness relation R_L, 1^n: security parameter.

Stage 1:

V uniformly chooses $r \in \{0,1\}^{log^k n}$, $B \in \{0,1\}^{poly(n)}$

V $\rightarrow$ P: $b = f(r)$, B

Stage 2:

P $\rightarrow$ V: $c_1 = Com(0^n), c_2 = Com(w)$, a zap using B as randomness, showing the statement

either there exists r' s.t $c_1 = Com(r')$ and $b = f(r')$
or there exists w s.t. $c_2 = Com(w)$ and $w \in R_L(x))$

We start by showing that Π is an interactive argument that is stand-alone straight-line simulatable.

Proposition 4 *The protocol* Π *is an interactive argument that is straight-line* $n^{O(log^k n)}$*-simulatable.*

Proof of Proposition 4 Completeness follows directly from the completeness of the zap.

Soundness To prove soundness of the protocol we rely on the complexity leveraging technique of [14]. Assume, for contradiction, that there exist a malicious prover

P^* that succeeds in convincing the honest verifier of a false statement x with non-negligible probability. Assume further (this assumption is currently unjustified, but will soon be removed) that the prover only succeeds in constructing accepting zap proofs for true statements. This means that with non-negligible probability it holds that

1. either $c_1 = Com(r')$ and $b = f(r')$
2. or $c_2 = Com(w)$ and $(x, w) \in R_L$

Since the statement x is false (i.e., $x \notin L$) the first condition must be fulfilled with non-negligible probability. Now, since the commitment scheme Com is $n^{O(log^{k'} n)}$-extractable, the value r' can be extracted in time $n^{O(log^{k'} n)}$. By combining the malicious prover and the extractor for Com we can thus construct a machine A (with running time bounded by $n^{O(log^{k'} n)}$) which on input a value $b = f(r)$, for a randomly chosen $r \in \{0,1\}^{log^k n}$, succeeds in outputting $f^{-1}(b)$ with non-negligible probability, contradicting the fact that f is one-way for subexponential circuits (since f is one-way for subexponential circuits, f is hard to invert for time $2^{(log^k n)^\kappa} = n^{log^{2k'} n} > n^{O(log^{k'} n)}$ for inputs of the size of $|r|$ (i.e., $log^k n$)).

Recall that in order to reach contradiction we made the unjustified assumption that P^* never succeeds in constructing accepting zaps of false statements. In reality, P^* might, however, be able to construct an accepting zap of a false statement. Fortunately, due to the soundness of the zap, this only happens with negligible probability. We conclude that the above argument thus still holds with overwhelming probability.

Zero-knowledge Let us turn to zero-knowledge. Consider a straight-line simulator S that upon receiving V^*'s first message (b, B) performs an exhaustive search to find a value r' such that $f(r') = b$. (Note that since f is one-to-one, there exists such a value exists for every value $b \in \{0,1\}^{log^k n}$.) S thereafter computes $c_1 = Com(r')$, $c_2 = Com(0^n)$ and thereafter produces an accepting zap proof with respect to B (using r' as witness).

It follows using the same argument as in the proof of Proposition 1 that the running time of S is bounded by $n^{log^k n}$. It remains to show that the output of the simulator is indistinguishable from the output of the verifier V^* in a real execution with an honest prover. Towards this goal, we define the following experiments.

- **Simulated execution** Let E_0 denote the output of the verifier $V^*(z)$ after interacting with the straight-line simulator S. i.e.,

$$E_0 = \langle S, V^*(z) \rangle (x)$$

- **Simulated execution with commitment to real witness** Let S' be a hybrid simulator which gets the witness w to the statement x. S' proceeds exactly as S, but instead of letting $c_2 = Com(0^n)$, S' lets $c_2 = Com(w)$. (Note that S' still uses the witness r' when constructing the zap proof.) Let E_1 denote the output of the verifier $V^*(z)$ after interacting with the straight-line simulator S', i.e.,

$$E_1 = \langle S'(w), V^*(z) \rangle (x)$$

- **Real execution with commitment to "fake" witness** Let S'' be a hybrid simulator that proceeds exactly as S', but instead of using the "fake" witness r' when constructing the zap proof (as S' does), S'' uses the real witness. Note that the only difference between S'' and a real prover is that S'' commits to both the real and the "fake" witness, while the prover only commits to the real witness. Let E_2 denote the output of the verifier $V^*(z)$ after interacting with the straight-line simulator S'', i.e.,

$$E_1 = \langle S''(w), V^*(z) \rangle (x)$$

- **Real execution** Let E_0 denote the output of the verifier $V^*(z)$ in a real execution with the prover, i.e.,

$$E_0 = \langle P(w), V^*(z) \rangle (x)$$

We proceed to show that the outputs of each of these experiments are indistinguishable for every large enough $x \in L$, $y \in R_L(x)$ and every $z \in \{0,1\}^*$. Without loss of generality we restrict our attention to only deterministic verifiers.

Claim 5 *E_0 is indistinguishable from E_1*

Proof Note that the only difference between the view of V^* in the executions that generate these distributions is the value that c_2 is a commitment to. Thus intuitively it follows from the hiding property of Com that these distributions are indistinguishable. In order to make this argument formal, we, however, need to make a slightly more refined argument. In fact, note that since the running times of S and S' are quasi-polynomial, a naive argument would result in machine that violates the hiding property of the commitment scheme in quasi-polynomial time (which would not be enough for our goal, since we actually need a commitment scheme that is *extractable* in less time than the running time of the simulator). We show how to use the fact that the commitment scheme is hiding for *non-uniform* adversaries to circumvent this problem. Namely, since the machine V^* is deterministic, the first message (b, B) sent by V^* is fixed. Let $\tilde{r}'$ be a value such that $f(\tilde{r}') = b$, and let $\tilde{S}$ be a machine, which gets $\tilde{r}'$ and w as auxiliary input, and proceeds just as S, except that instead of performing an exhaustive search to find the value r', $\tilde{S}$ just lets $r' = \tilde{r}'$. Note that the output of $\tilde{S}$ in interaction with $V^*(z, x)$ is identically distributed to the output of S in the same interaction. However, while the running time of S is super-polynomial, $\tilde{S}$ runs in polynomial time. We use the same method to obtain a polynomial-time variant $\tilde{S}'$ of S', and can now rely on the hiding property of Com to conclude that distributions E_0 and E_1 are indistinguishable. ■

Claim 6 *E_1 is indistinguishable from E_2*

Proof Note that the only difference in the executions that generate these distributions is the choice of the witness used when constructing the zap. It thus follows from the witness indistinguishability property of the zap that the these distributions are indistinguishable. ■

Claim 7 *E_2 is indistinguishable from E_3*

Proof Note that the only difference between the view of V^* in the executions that generate these distributions is the value that c_1 is a commitment to. It thus follows, using the same argument as in the proof of Claim 6 (which relies on the hiding property of the commitment scheme) that the distributions E_2 and E_3 are indistinguishable. ■

We conclude, by the triangle inequality, that the distributions E_0 (the simulated execution) and E_3 (the real execution) are indistinguishable. ■

Remark 11 *We note that in the soundness proof of the protocol we actually show a stronger property, namely that a a witness to the statement proved can be extracted in time quasi-polynomial time. This fact shows that the protocol is* not *strongly quasi-polynomial time simulatable.*

We continue by showing that Π is straight-line concurrent $n^{O(log^k n)}$-simulatable:

Lemma 6 *The protocol Π is straight-line concurrent $n^{O(log^k n)}$-simulatable.*

Proof of Lemma 6 Let S be the straight-line (stand-alone) simulator for Π (guaranteed by Proposition 4). We show that $S'(i, x) = S(x)$ is a straight-line concurrent simulator for Π.

We start by noting that since the protocol Π only consists of two rounds we only need to consider two different schedulings: parallel repetitions and sequential repetitions. In order words, a concurrent scheduling is always a sequential repetition of parallel repetitions of the protocol Π.

Dealing with Parallel Repetitions For simplicity, we start by considering only parallel repetitions. Let $g(n)$ be a polynomial, P be the honest prover, and let $A_{\mathtt{para}}$ be a probabilistic polynomial-time adversary that only uses a parallel scheduling. It follows using a hybrid argument, which relies on the stand-alone $T(n)$-simulatability of (P, V), that $S'(i, x)$ is a straight-line simulator for the adversary $A_{\mathtt{para}}$, i.e., that the following ensembles are indistinguishable:

- $\{A_{\texttt{para}}^{P(x_1,y_1),P(x_2,y_2),..P(x_{g(n)},y_{g(n)})}(z,x_1,x_2,..,x_{g(n)})\}_{z\in\{0,1\}^*,x_1,x_2,..,x_{g(n)}\in L}$ for arbitrary $y_i \in R_L(x_i)$
- $\{A_{\texttt{para}}^{S(x_1),S(x_2),..S(x_{g(n)})}(z,x_1,x_2,..,x_{g(n)})\}_{z\in\{0,1\}^*,x_1,x_2,..,x_{g(n)}\in L}$

More precisely, define the hybrid distribution

$$H_i = A_{\texttt{para}}^{P(x_1,y_1),...,P(x_i,y_i),S(x_{i+1}),...,S(x_{g(n)})}(z,x_1,x_2,...,x_{g(n)})$$

Assume, for contradiction, that H_0 (i.e. the output of $A_{\texttt{para}}$ in the real execution) is distinguishable from $H_{g(n)}$ (i.e., the output of $A_{\texttt{para}}$ in the simulated execution). Without loss of generality we restrict our attention to a deterministic adversary $A_{\texttt{para}}$. This means that for each set of instances $\bar{x} = (x_1,...,x_{g(n)})$ and auxiliary input z, the message $(b_1,B_1),..,(b_{g(n)},B_{g(n)})$ sent by $A_{\texttt{para}}(z,\bar{x})$ is determined. For each such message there is thus a string $\bar{r} = (r_1,...,r_{g(n)}$ such that for $1 \le i \le g(n)$, $b_i = f(r_i)$. Now consider a polynomial-time simulator S' which receives the string $\bar{r}$ as auxiliary input and proceeds just as S, but instead of performing an exhaustive search to find a pre-image, S' simply uses the value obtained as auxiliary input in order to perform it simulation. It follows that

$$H_i = A_{\texttt{para}}^{P(x_1,y_1),...,P(x_i,y_i),S'(x_{i+1},\bar{r}),...,S'(x_{g(n)},\bar{r})}(z,x_1,x_2,...,x_{g(n)})$$

By our assumption that H_0 and $H_{g(n)}$ are distinguishable, it follows using the triangle inequality that there must exist an index j such that such that the distributions H_j and H_{j+1} are distinguishable. Since both P and S' are polynomial-time, we thus contradict the stand-alone $T(n)$-simulatability of (P,V).

Dealing with the Full Scheduling Let us now turn to the "full" scheduling, which might contain sequential repetitions of the parallel repetitions. Since the prover strategy P for Π is efficient, we can use the same argument as in the proof of the sequential composition lemma (i.e., Lemma 3) to show that the straight-line simulator S' also works for adversaries A that receive (many) sequential repetitions of parallel repetitions of proofs using Π.

Finally, note that clearly the simulator $S'(i,x)$ runs in time $n^{O(log^k n)}$. ■

We thus obtain the following theorem:

Theorem 5 *Assume the existence of a one-to-one function $f : \{0,1\}^n \to \{0,1\}^n$ that is one-way functions for subexponential circuits, and the existence of zaps. Then, there exists a two round interactive argument that is straight-line concurrent $n^{poly(logn)}$-simulatable.*

Remark 12 *1. Zaps can be constructed based on the existence of non-interactive zero-knowledge proofs in the Common Reference String model, which in turn can be based on the existence of trapdoor permutations [66].*

2. *We mention that since the protocol is not* strongly *quasi-polynomial time simulatable (see Remark 11) it is not zero-knowledge in the on-line/off-line model. However, it does satisfy the requirements needed for the composition theorem in section 3.3.*

3.6 A Characterization of the Round-complexity

In this section we give a complete characterization of the round-complexity of quasi-polynomial time simulatable protocols.[11] It can be summarized as follows: Unless $\mathcal{N}P \in \mathcal{D}TIME(n^{poly(logn)})$,

- Two rounds are sufficient and necessary for quasi-polynomial time simulatable interactive *arguments* for $\mathcal{NP}$ (assuming the existence of one-to-one one-way functions secure for subexponential circuits, and the existence of zaps).
- Three rounds are sufficient and necessary for quasi-polynomial time simulatable interactive *proofs* for $\mathcal{NP}$ (assuming the existence of one-way functions).

3.6.1 On the Round-complexity of Simulatable Arguments

We show the impossibility of one-round $n^{poly(logn)}$-simulatable proofs for languages that are not decidable in quasi-polynomial time. This, in particular, means that the protocol in section 3.5.2 is round-optimal.

Theorem 6 *Assume there exists a one-round interactive proof for the language L that is $n^{poly(logn)}$-simulatable. Then L is decidable in quasi-polynomial time.*

Proof of Theorem 6 Assume that there exist a $n^{poly(logn)}$-time simulator S for an interactive argument for the language L. Let D be a machine that first runs S and thereafter runs the honest verifier strategy on the output of S. We show that D decides the language L:

Claim 8 *When $x \in L$, D outputs 1 with probability at least 1/2.*

Proof of Claim 8 Assume, for contradiction, that D outputs 1 with probability smaller than 1/2. This would imply that the honest verifier strategy distinguishes between the output of the honest prover and the simulator. The claim follows by contradiction. ■

Claim 9 *When $x \notin L$, D outputs 1 with negligible probability.*

[11] Interestingly the round-complexity of zero-knowledge protocols with negligible soundness error is still unknown. In fact, it is still unknown whether there exist 3-round zero-knowledge proofs/arguments for $\mathcal{NP}$.

Proof of Claim 9 Assume, for contradiction, that S convinces the honest verifier with non-negligible probability. This means that for a non-negligible part of random tapes r ,the machine S with random tape fixed to r will convince the honest verifier with non-negligible probability. Let z_r denote the output of S when run with the random tape r. Using an averaging argument, it follows that there exist an r' such that the honest verifier accepts $z_{r'}$ with non-negligible probability.

Now consider the non-uniform cheating prover P^* that simply outputs its auxiliary input. Then $P^*(z_r)$ will convince the honest verifier with non-negligible probability, which contradicts the soundness of the interactive argument. ■

Since the running time of S is bounded by $n^{poly(logn)}$, we conclude that L is decidable in quasi-polynomial time. ■

Remark 13 *We note that the proof of theorem 6 makes use of the fact that the soundness of the argument system holds against non-uniform adversaries. Recent results by Barak and Pass [3] show that if the soundness condition of the interactive argument is weakened to hold only against uniform adversaries, then one-round quasi-polynomial time simulatable arguments can be constructed under certain general (although non-standard) complexity theoretic assumptions.*

3.6.2 On the Round-complexity of Simulatable Proofs

Changing perspective and considering proofs instead of arguments, we show the impossibility of two-round $n^{poly(logn)}$-simulatable proofs for languages that are not decidable in quasi-polynomial time:

Theorem 7 *Assume there exist a two-round interactive proof for the language L that is $n^{poly(logn)}$-simulatable. Then, L is decidable in quasi-polynomial time.*

Proof of Theorem 6 Recall the proof of the impossibility result for non-trivial two-round auxiliary-input zero-knowledge of Goldreich-Oren [37]. They show how that the simulator S for the cheating verifier that simply forwards its auxiliary input as its message, can be used to decide the language. More precisely, on input an instance x and a honestly generated first verifier message, S outputs a message that will be accepted by the honest verifier if $x \in L$ (by the zero-knowledge property), while it will be rejected with high probability if $x \notin L$ (by the soundness). S, combined with the honest verifier thus decide the language L. We note that the same transformation can be used also in the case of quasi-polynomial simulatable proofs. Since in this setting the simulator S runs in quasi-polynomial time, the deciding machine obtained will also run in quasi-polynomial time. ■

Remark 14 *We note that the above proof actually gives a stronger result than stated. Namely, it shows the impossibility of "non-trivial" two-round interactive arguments* that are sound for quasi-polynomial time.

We mention that the lower bound can be matched. Consider the 3-round interactive proof consisting of $log^2 n$ parallel repetitions of the graph hamiltonicity protocol of [7]. The protocol has both negligible soundness error, and is simulatable in quasi-polynomial time, by simple rewinding (see [14] for a proof).

Theorem 8 *Assume the existence of one-way functions. Then, there exists a 3-round $n^{poly(logn)}$ simulatable proof for $\mathcal{NP}$.*

3.7 Extensions

If assuming one-to-one one-way functions secure for *exponential-sized* circuits (instead of assuming security for sub-exponential sized circuits as we do), then our protocols can be modified in a straight-forward way to become $n^{\omega(1)}$-simulatable. In fact, if assuming provers that are computationally bounded below a specific polynomial $f(n)$, then our protocols can be modified to become simulatable in time $g(n)$, where $g(n) > f(n)$ is another polynomial. It would be interesting to extend this analysis to exact security.

3.8 Subsequent Work

One-message Weak Zero-Knowledge Barak and Pass show that if weakening the soundness condition of interactive arguments to hold only for *uniform* machines, then there exists a one-message (i.e., non-interactive) quasi-polynomial time simulatable argument system for $\mathcal{NP}$, under certain general (although non-standard) complexity theoretic assumptions [3]. Their protocol can be seen as a derandomized version of our two-round protocol.

Universal Composability with Super-polynomial Time Simulators Recent work by Prabhakaran and Sahai extend our notions to the framework for Universal Composability [58]. In particular, Prabhakaran and Sahai show how to obtain universally composable protocols with super-polynomial time simulators, under certain (new and non-standard) assumptions. We mention that both the security definitions and the techniques used to instantiate them rely (and extend) on our work.

Applications of Quasi-polynomial Time Simulatable Arguments

1. Pass and Rosen use the notion of quasi-polynomial time simulatable arguments, and in particular one of the protocols constructed in this chapter, as a tool to prove security according to standard definitions of polynomial-time simulatability. More precisely, straight-line concurrent quasi-polynomial time simulatable arguments are used as a component to obtain a constant-round protocol for bounded-concurrent secure two-party computation [56].

2. di Crescenzo, Persiano and Visconti rely on the notion of straight-line $T(n)$-simulatability, and on protocols very similar to ours in order to obtain *Resettable Zero Knowledge protocols with Concurrent Soundness in the Bare Public-Key Model* [20]. We note that the use of these protocols is a crucial (and main) part of their construction.

3.9 Open Problems

In this work we have taken a initial step towards understanding the potential of using super-polynomial time simulation as a relaxation of the standard notion of security. Our focus has been on zero-knowledge interactive proofs. An interesting open problem is to extend our techniques also to other cryptographic problems. A prime candidate would be notion of non-malleability: It is possible to obtain an efficient protocol for the tasks of non-malleable zero-knowledge and non-malleable commitments, using a relaxed security definition where the simulator is allowed to be a super-polynomial (quasi-polynomial) machine?

3.10 Acknowledgments

I wish to thank Johan Håstad for his invaluable help and comments. Special thanks to Boaz Barak for suggesting the use of complexity leveraging in a way similar to [14] to obtain two-round protocols. I am also very grateful to Shafi Goldwasser, Alon Rosen and Yael Tauman-Kalai for helpful comments. Finally, I would like to thank the anonymous EuroCrypt referees.

Part II

Results in Shared Object Models

Chapter 4

Deniable Zero-Knowledge

Abstract

We revisit the definitions of zero-knowledge in the Common Reference String (CRS) model and the Random Oracle (RO) model. We argue that even though these definitions syntactically mimic the standard zero-knowledge definition, they loose some of its spirit. In particular, we show that there exist a specific natural security property that is not captured by these definitions. This is the property of *deniability*. We formally define the notion of *deniable zero-knowledge* in these models and investigate the possibility of achieving it. Our results are different for the two models:

- Concerning the CRS model, we rule out the possibility of achieving deniable zero-knowledge protocols in "natural" settings where such protocols cannot already be achieved in plain model.
- In the RO model, on the other hand, we construct an efficient 2-round deniable zero-knowledge argument of knowledge, that preserves both the zero-knowledge property and the proof of knowledge property under concurrent executions (concurrent zero-knowledge and concurrent proof-of knowledge).

4.1 Introduction

Zero-knowledge proofs, i.e., interactive proofs that yield no other knowledge than the validity of the assertion proved, were introduced by Goldwasser, Micali and Rackoff [38] in 1982. Intuitively, the verifier of a zero-knowledge proof should not be able to *do* anything it could not have *done* before the interaction. Knowledge, thus, in this context means the ability to perform a task. The intuition is formalized through a simulation definition: We say that a protocol is zero-knowledge if there exists a simulator (that does not have access to a prover) that can simulate a malicious verifier's output after interaction with a prover. The existence of such a simulator implies that if an adversary succeeds in a task after having communicated with a prover, then the adversary could just as well have reached the same results without the help of the prover, by first running the simulator. This feature has made zero-knowledge a very powerful and useful tool for proving the security of cryptographic protocols.

For some applications, such as signature schemes [31] [64], voting systems, non-interactive zero-knowledge [9] [37], concurrent zero-knowledge [25], [16] etc., it however seems hard, or is even impossible, to achieve efficient and secure schemes in the standard model. Stronger models, such as the Common Reference String (CRS) model [9], where a random string is accessible to the players, or the Random Oracle (RO) model [6], where a random function is accessible through oracle calls to the players, were therefore introduced to handle even those applications. Recently the CRS model has been extensively used in interactive settings to obtain secure protocols in the framework for Universal Composability (e.g. [12] [13] [17]).

We note that an important part of the intuition behind zero-knowledge is lost in these two models in a multi-party scenario, *if the CRS string or the random oracle may be reused.* An easy way of seeing this is simply by noting that non-interactive zero-knowledge proofs are possible in both these model. A player having received a non-interactive proof of an assertion, it could not have proved before the interaction, can definitely do something new: it can simply send the same proof to someone else. This fact may seem a bit counter-intuitive since the intuition tells us that the simulation paradigm should take care of this. We note, however, that the simulator is much "stronger" in these models than in the plain model. As it is, the simulator is allowed to *choose* the CRS string, or random oracle, and this fact jeopardizes the zero-knowledge intuition. In fact, one could say that the zero-knowledge definition in these models only guarantees that the verifier does not learn anything, *provided that he was able to choose the CRS or random oracle.* Since the verifier clearly does not have the position to do so (in the standard formulations of the CRS and RO models), we conclude that the zero-knowledge property in these model only guarantees that the verifier will not be able to do anything *that is unrelated to the CRS or random oracle*, it could not have done before.

In the non-interactive setting, this problem has lead to the definition of *non-malleable non-interactive zero-knowledge* [62], and very recently *robust non-interactive zero-knowledge* [23]. In this paper we examine the problem in the more general interactive setting.

Deniable Zero-knowledge. In many applications of interactive protocols (e.g. undeniable signatures [18], or deniable authentication [25]) it is essential that the transcript of the interaction does not yield any evidence of the interaction. We say that such protocols are *deniable.* We use the standard simulation paradigm to formalize this notion:

Definition 35 *[Informal meta-definition] A protocol is* deniable *if it is zero-knowledge and the zero-knowledge simulator can be run by the verifier.*[1]

[1] *Strictly speaking, the simulator is an algorithm and can therefore always be run by the*

The standard definition of zero-knowledge in the plain model certainly satisfies deniability, however this is *no longer* the case with the definitions of zero-knowledge in the CRS/RO models. This stems from the fact that in the real world the shared object in the model, i.e., the CRS string or the random oracle, is fixed once and for all at start-up. When proving security, however, the simulator in these models is allowed to choose (or program) this shared object in anyway it pleases as long as it "looks" ok. Thus, even though there exists a simulator for a protocol, there is no guarantee that a player can actually simulate a transcript using a certain predefined shared object. Non-interactive proofs of a statement x are trivially proofs of an interaction with a party that can prove the assertion of the statement x, or else the soundness condition of the proof would be broken.

Indeed, the idea behind the simulation paradigm, and the reason for its widespread applicability, is that a verifier should be able to run the simulator by himself instead of interacting with a prover. The standard definitions of zero-knowledge in the CRS and RO models have not retained this spirit (since the simulator in these model is allowed to choose the shared object, which evidently the verifier is not allowed to do), but only syntactically mimic the original zero-knowledge definition.

In the following we give formal definitions of deniable zero-knowledge in the CRS (see section 4.2) and RO (see section 4.3) models and investigate the possibility of achieving protocols satisfying the definitions.

When Does Deniability Matter. For some applications zero-knowledge and deniability is the goal (e.g. deniable authentication [25]). For these applications the standard definitions of zero-knowledge in the CRS/RO models clearly are not sufficient, since they do not guarantee deniability.[2]

The issue of deniability also arises when a zero-knowledge protocol is used as a sub-protocol in a larger context where the CRS string or random oracle may be reused. In such a scenario it is no longer clear what security properties are guaranteed by the standard definitions of zero-knowledge in the CRS/RO models. More technically, general protocol composition becomes problematic since the simulator cannot be run when a specific CRS string or random oracle already has been selected.

Nevertheless, we mention that when "plugging-in" zero-knowledge protocols in the CRS/RO models into certain specific protocols, the standard definitions (that do not guarantee deniability) can in some cases be sufficient. For example in the construction of encryption schemes secure against chosen-ciphertext attacks [53], zero-knowledge protocols that do not satisfy deniability have been successfully used as sub-protocols.[3] (Looking ahead, the notion "unreplayability" introduced in sec-

---

verifier. What we mean here is that the output of the verifier when running this simulator algorithm should be "correctly" distributed.

[2]We mention that it was pointed out in an early version of [24] that a the concurrent zero-knowledge protocol in the CRS model of [21] is not sufficient for the task of deniable authentication.

[3]We mention that in the more complicated case of encryption schemes secure against *adaptive* chosen-cipher text attacks, the standard definition of zero-knowledge in the CRS model is not

tion 4.1.1 is another example where zero-knowledge definitions that do not satisfy deniability can be sufficient).

Implications on the Framework for Universal Composability. Recently, Canetti introduced a new framework for the construction of cryptographic protocols, called the framework for Universal Composability (UC) [12]. The framework for UC suggest the following paradigm for the design of cryptographic protocols.

1. Design an "idealized" functionality, $\mathcal{F}$.

2. Construct a protocol Π and prove that it implements the idealized functionality $\mathcal{F}$.

3. Design more complicated protocol using calls to idealized functionalities $\mathcal{F}_1, \mathcal{F}_2, ...$

4. Finally substitute calls to the idealized functionalities $\mathcal{F}_1, \mathcal{F}_2, ...,$ with protocols $\Pi_1, \Pi_2, ...$ that implement them and rely on a *general composition theorem* to show that the resulting protocol is as secure as the protocol which relied on the idealized functionalities.

In short, the UC framework allows for a modular design of cryptographic protocols, which facilitates the design of secure solutions for more complicated application, e.g. [13] [17]. Furthermore, protocols proven secure in the UC framework enjoy the property of remaining secure even when *concurrently* executed with any other, arbitrary, protocol.

The "ideal" zero-knowledge functionality was first defined in [12] and has later been used in several subsequent works. Due to the impossibility of implementing the ideal zero-knowledge functionality in the plain model [12], the functionality was implemented in the CRS model [13, 23]. We note that the implementation of [23] is non-interactive, i.e., only a single message is send. Their protocol is, thus, not deniable and therefore constitutes an evidence that in the framework for UC, the ideal zero-knowledge functionality *does not* capture the concerns for deniability.

The example given shows the non-triviality of the task of defining ideal functionalities in the UC framework. At a first glance it seemed like the definition given of the ideal zero-knowledge functionality would satisfy deniability. Closer inspection of the framework shows, however, that *the concern for transferability/deniability is not taken into account in the framework for UC when introducing shared objects*, such as the CRS string. This can be seen as follows: The UC framework only guarantees security if a CRS string is not reused. A transferability/deniability attack, however, relies on the fact that an honest-party reuses a CRS that has been used in a different execution. In other words, such attacks are not ruled-out by the composition theorem of [12], since they involve honest-parties deviating from their prescribed protocols by *reusing* a CRS string.

---

sufficient, but needs to be strengthened to guarantee *simulation-soundness.* [62]

4.1.1 Results Concerning the CRS Model

There could have been hope that the CRS model might be used to implement deniable $\mathcal{ZK}$ protocols in settings where the plain model is not sufficient. We show that in natural settings, where the usage of the CRS model seems meaningful, the demand for deniability makes the CRS model collapse down to the plain model:

- We show that known black-box impossibility result concerning $\mathcal{ZK}$ in the plain model also hold in the CRS model, with respect to deniable $\mathcal{ZK}$. That is, we show the impossibility of non-trivial black-box deniable $\mathcal{ZK}$ arguments in the CRS model with either of the following extra requirements:
  - straight-line simulatability (i.e., non-rewinding simulatability)
  - non-interactive
  - constant-round concurrent $\mathcal{ZK}$
  - three-round
  - constant-round strict polynomial-time simulatable
  - constant-round public-coin

Achieving a Weaker Form of Deniability. Although our results rule out the possibility of "interesting" deniable $\mathcal{ZK}$ protocols in many natural settings, we show that a limited form of deniability can be achieved in the CRS model by restricting the communication to a certain class of pre-specified protocols where the CRS string may be reused. Very loosely speaking, we say that a class of protocols is closed under *unreplayability* if an adversary cannot prove anything using a protocol in the class, after having interacted with a prover using a protocol in the class, that it could not have done before interacting with the prover. We show that a natural class of protocols is closed under unreplayability in the CRS model : If C is a class of interactive proofs (or arguments) *of knowledge* that are zero-knowledge in the CRS model, then C is closed under unreplayability. This result shows that restricting the communication to only zero-knowledge *arguments of knowledge* eliminates the concern for deniability in the CRS model.

4.1.2 Results Concerning the RO Model

While the results in the CRS model were mostly negative in nature, the situation in the RO model is rather different. Indeed we are able to construct "interesting" deniable $\mathcal{ZK}$ protocols.

More precisely, we show that 2 rounds are necessary and sufficient to construct deniable $\mathcal{ZK}$ arguments for $\mathcal{NP}$ in the RO model. In fact, we construct an efficient 2-round deniable $\mathcal{ZK}$ argument for $\mathcal{NP}$ in the RO model that is both straight-line simulatable and witness extractable. This implies that both simulation of polynomially many concurrent executions (concurrent zero-knowledge) and simultaneous extraction of polynomially many witnesses under concurrent executions (concurrent proof of knowledge) can be performed. As far as we know it was previously unknown how to simultaneously extract witnesses from polynomially many proofs in the RO model (let alone the question of deniability).

4.1.3 Other Models

We mention briefly that there are other models that are stronger than the plain model, such as the timing model of [25], or the on-line/off-line model of [54], that do not suffer from problems with deniability. We also note that in a public-key model, methods similar to those of designated verifiers [44] can be used to successfully implement non-trivial $\mathcal{ZK}$ protocols that are deniable. Indeed, the method of designated verifier shows how to convert $\mathcal{ZK}$ protocols that are not deniable into $\mathcal{ZK}$ protocols in a stronger model (namely the public-key model) that satisfy deniability.

4.1.4 Technical Contribution

Although this chapter is mostly conceptual in nature, we believe that some of the techniques used in the proofs might be of independent interest.

Straight-line Extractable Commitments We define and construct *straight-line extractable* commitment schemes in the RO model. Such commitment schemes have the property that the value committed to can be extracted in polynomial time if given access to a list of all the queries to the random oracle made by the committer. We present very efficient (practical) constructions of both statistically binding and statistically hiding non-interactive straight-line extractable commitments in the RO model.

Straight-line Witness Extractable Proofs Using straight-line extractable commitments we construct efficient non-interactive $\mathcal{ZK}$ arguments in the RO model which have the property that a witness to the statement proved can be extracted in polynomial time if given access to a list of all the queries made to the random oracle by the prover. We say that such a proof is *straight-line witness extractable*.

We mention that the straight-line extraction feature implies two strong properties that were (as far as we know) previously unattained in the RO model:

- **Simultaneous extraction of polynomially many witnesses**. Previous methods to extract witnesses in the RO model [64] relied on rewinding and could therefore not be used to extract witnesses under concurrent executions.
- **Tight security reductions for non-interactive proofs of knowledge**. Standard extraction techniques for non-interactive proofs of knowledge in the

RO model [64] result in "loose" security reductions (see [39] for a discussion).[4] Using straight-line extraction, on the other hand, we obtain proofs of knowledges with a linear (and thus optimal) security reduction.

4.1.5 Related Work

Criticism of the RO Model Since truly random functions which short descriptions do not exist, Bellare and Rogaway suggest, as a heuristic step, to instantiate the RO, in schemes proven secure in the RO model, with a hashfunction [6]. In recent years, this heuristic step has meet serious criticism. In particular it has been shown that there exists encryption schemes that are provably secure in the RO model, yet are insecure for every instantiation of the RO with hashfunction [15]. Other examples of such separation results include [40, 51]

Although all these "counter-examples" are quite artificial, they indicate that security proofs for "natural" protocols in the RO model can not "easily" be turned into proofs in the standard model (without a RO). We do not take a stand in the on-going debate (see for example [45]) on whether the RO model could/should be used as a heuristic in order to obtain practical protocol. We merely point out that, *even if we believe that the RO heuristic works for some class of "natural" protocols* the security guarantees obtained from proofs in the random oracle model are weaker than those guaranteed by the security definitions in the standard model. In particular, commonly used protocols in the RO model are not deniable. (Our work shows that there, on the other hand, do exist protocols in the RO model that are deniable.)

The Non-programmable RO Model In a recent work (done independently of ours), Nielsen shows the existence of an efficient scheme for the task of *non-committing encryption* in the RO, while such schemes do not exist if the simulator is not allowed to *program*, or choose, the RO [51]. We note that the definition of the non-programmable RO model of Nielsen bears resemblance to our definition of deniable $\mathcal{ZK}$ in the RO model. However, whereas Nielsen simply introduces this model an an intermediary step in order to show a separation between the RO model and the standard model, we provide a strong *justification* for why it is interesting to investigate schemes that are proven secure without programming the RO.

4.1.6 Overview

In Section 4.2 we present our results concerning the possibility of deniable $\mathcal{ZK}$ protocols in the CRS model. Section 4.3 contains our results on the RO model. Finally, in Section 4.4 we define the notion of unreplayability, discuss the possibility of achieving it and compare it to the notion of deniability.

[4] Roughly, in order to break the underlying assumption the "cheating prover" has to be run $O(q)$ times, where q is the running time of the cheating prover, thus resulting in a total running time of $O(q^2)$.

4.2 On Deniable Zero-Knowledge Proofs in the CRS Model

In this section we investigate the possibility of obtaining deniable $\mathcal{ZK}$ protocols in the CRS model. We start by formally defining the notion of deniable $\mathcal{ZK}$ in the CRS model. Recall that in order to obtain a deniable zero-knowledge protocol we require that a real-world verifier should be able to perform a simulation which produces a transcript that is indistinguishable from the real transcript, *even* if the distinguisher gets access to the actual Common Reference String. More formally,

Definition 36 *We say that an interactive proof (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is deniable zero-knowledge in the CRS model if there for every PPT machine V^* exists a probabilistic expected polynomial time simulator S such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$)*

- $\{(r, \langle P(y_x), V^*(z)\rangle(x, r))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary* $y_x \in R_L(x)$
- $\{(r, S(x, z, r)\}_{z\in\{0,1\}^*, x\in L}$

where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$.
That is, for every probabilistic algorithm D running in time polynomial in the length of its first input, every polynomial p, all sufficiently long $x \in L$, all $y_x \in R_L(x)$ and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$\begin{aligned} &|Pr[D(x, z', r, \langle P(y_x), V^*(z)\rangle(x, r))) = 1] \\ &-Pr[D(x, z', r, S(x, z, r)) = 1]| < \frac{1}{p(|x|)} \end{aligned}$$

where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$.

Remark 15 *We note that, in analogy with the standard definition of $\mathcal{ZK}$ in the CRS model, Definition 36, is not necessarily closed under sequential composition.*

Note that the difference between the standard $\mathcal{ZK}$ definition (see Definition 23) and Definition 36 is that whereas in the standard definition of $\mathcal{ZK}$ in the CRS model the simulator is allowed to *choose* (or program) the CRS string, the definition of deniable $\mathcal{ZK}$ requires the simulator to succeed for almost every randomly generated CRS string, without obtaining any trapdoor information about the string.

Black-box deniable zero-knowledge We also provide black-box definition, i.e., a definition which requires the existence of a simulator that uses the verifier as a black-box.

Definition 37 *We say that an interactive proof (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is black-box deniable zero-knowledge in the CRS model* *if there for every polynomial $p(n)$ exists a probabilistic expected polynomial time oracle machine S such that for every probabilistic polynomial-time interactive machine V^* that uses at most $p(n)$ random coins, the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$)*

- $\{(r, \langle P(y_x), V^*(z)\rangle(x,r))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary* $y_x \in R_L(x)$
- $\{(r, S^{V^*}(x,z,r)\}_{z\in\{0,1\}^*, x\in L}$

where r is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)}$.

4.2.1 On the Impossibility of "Non-trivial" Deniable ZK Protocols

We show that known black-box impossibility result concerning $\mathcal{ZK}$ in the plain model also hold in the CRS model with respect to deniable $\mathcal{ZK}$. That is we show that for known settings where it seems interesting to resort to the CRS model the demand for deniability makes the CRS model collapse down to the plain model.

We start by showing the impossibility of deniable $\mathcal{ZK}$ proofs that are proven secure using black-box technique *without the use of rewinding.*

Theorem 9 *Assume there exists a straight-line (i.e., non-rewinding) black-box simulatable deniable zero-knowledge proof (or argument) for the language L in the CRS model. Then $L \in \mathcal{BPP}$.*

Proof Assume that there exist a straight-line black-box simulator S for the honest verifier of Π running in expected time $p(n)$. Let S' be a machine that runs S for $10p(n)$ steps and the aborts if S has not output anything. It follows, by the Markov's inequality, that S aborts with probability less than $1/10$. Now, consider the machine D that on input x uniformly chooses a CRS string r and thereafter runs $S'(x,r)$ against the honest verifier V^* (on input x and r). We show that D decides the language L:

- if $x \in L$, unless S' aborts, the simulator's output is indistinguishable from an interaction between the honest prover and the honest verifier. D therefore outputs 1 with high probability. (by the union bound the probability can be bounded from below by at most negligible probability less than the completeness bound of Π $-1/10$).
- if $x \notin L$ then, by the soundness of the interactive proof S' will only succeed in convincing the honest verifier with small probability.

Thus L is in $\mathcal{BPP}$ ■

We note that Theorem 9 has important implications for the framework for Universal Composability. More specifically, since UC protocols require straight-line black-box simulation [12], Theorem 9 rules out the possibility of constructing deniable UC $\mathcal{ZK}$ protocols in the CRS model.

As a sanity check to the definition we also note the impossibility of non-interactive $\mathcal{ZK}$ arguments for non-trivial languages,

Theorem 10 *Assume there exists a non-interactive deniable zero-knowledge argument for the language L in the CRS model. Then $L \in \mathcal{BPP}$.*

Proof Follows directly from Theorem 9 since non-interactive arguments need to be black-box straight-line simulatable. ■

Indeed, non-interactive proofs in the CRS model are the most obvious violation of deniability, since they can be passed on.

Goldreich-Krawczyk Reductions

In 1990, Goldreich and Krawczyk [35] presented the first black-box lower bounds for zero-knowledge proofs. More precisely, they show that non-trivial languages (i.e., languages L such that $L \notin \mathcal{BPP}$) cannot have black-box $\mathcal{ZK}$ arguments that either have less than four communication rounds, or are constant-round and public-coin.

The method of Goldreich-Krawczyk has later been used to show black-box impossibility results for constant-round concurrent $\mathcal{ZK}$ [16] and for constant-round zero-knowledge arguments that are simulatable in strict polynomial-time (as opposed to expected polynomial-time) [4].

On a high-level, the Goldreich-Krawczyk method is a constructive reduction from a probabilistic polynomial-time machine deciding the language L to a simulator of the zero-knowledge argument. That is, the existence of a simulator implies the existence of a probabilistic polynomial-time machine deciding the language, which in turn implies that the language is in $\mathcal{BPP}$. Since the reduction is black-box and constructive, the same reduction can be used for protocols that are deniable zero-knowledge in the CRS model. We construct a machine deciding the language, by simply first choosing a random string and thereafter running the deciding machine obtained in the Goldreich-Krawczyk reduction, feeding it the random string as a CRS string. Careful examination of the proofs of [35], [16], and [4] thus gives:

Theorem 11 *Let Π be a interactive argument for the language L in the CRS model. If one of the following conditions are fulfilled, then $L \in \mathcal{BPP}$.*

- Π *is constant-round black-box concurrent deniable* $\mathcal{ZK}$
- Π *is constant-round public-coin black-box deniable* $\mathcal{ZK}$
- Π *is 3-round black-box deniable* $\mathcal{ZK}$
- Π *is black-box deniable* $\mathcal{ZK}$ *with a strict polynomial-time simulator.*

4.2.2 Conclusions and Directions for Future Research

We have shown that for currently known settings, the CRS model cannot be used to implement deniable black-box zero-knowledge protocols for languages in $\mathcal{NP}$, that cannot already be implemented in the plain model. Looking ahead, in section 4.1.1 we, nevertheless, show that a limited form of deniability (called *unreplayability*) can be achieved by restricting the communication of honest-parties to a certain class of protocols.

Concerning the framework for Universal Composability [12], we have shown that the ideal $\mathcal{ZK}$ functionality (defined in [12] and [13]) does not capture the concerns for deniability. Our results furthermore rule out the possibility of constructing UC $\mathcal{ZK}$ protocols which are deniable (i.e., have property that when implemented with a real CRS, the UC simulator actually can be executed by the verifier).

We note, on the other hand, that if instead resorting to a public-key model, methods similar to those of designated verifier [44] could possibly be used to achieve universally composable deniable zero-knowledge.

Open Problems

An interesting open problem is to find a type of deniable zero-knowledge protocol that can be achieved in the CRS but not in the plain model. Since most of our results only apply in the black-box setting, a direction would be to investigate the non-black-box setting.

4.3 On Deniable Zero-knowledge Proofs in the RO Model

As in the CRS model, in order to obtain deniable $\mathcal{ZK}$ proofs and arguments in the RO model, we resort to a weaker simulation model, where the simulator should produce a distribution which is indistinguishable from a real execution, even when the distinguisher gets access to the actual random oracle. More formally,

Definition 38 *We say that an interactive proof (P, V) for the language $L \in \mathcal{NP}$, with witness relation R_L, is* deniable zero-knowledge in the RO model *if for every PPT verifier V^* there exists an expected polynomial time probabilistic simulator S such that the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$):*

- $\{(RO, \langle P^{RO}(y_x), V^{*RO}(z)\rangle(x))\}_{z\in\{0,1\}^*, x\in L}$ *for arbitrary $y_x \in R_L(x)$*
- $\{RO, S^{RO}(z, x))\}_{z\in\{0,1\}^*, x\in L}$

where $RO : \{0,1\}^{poly(|x|)} \to \{0,1\}^{poly(|x|)}$ is a uniformly distributed random variable.
That is, for every probabilistic algorithm D running in time polynomial in the length

of its first input, every polynomial p, all sufficiently long $x \in L$, all $y_x \in R_L(x)$ and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$|Pr[D^{RO}(x, z', \langle P^{RO}(y_x), V^{*RO}(z)\rangle(x))) = 1] \\ -Pr[D^{RO}(x, z', S^{RO}(x, z)) = 1]| < \frac{1}{p(|x|)}$$

where $RO : \{0,1\}^{poly(|x|)} \rightarrow \{0,1\}^{poly(|x|)}$ is a uniformly distributed random variable.

Note that according to the standard $\mathcal{ZK}$ definition in the RO model (see Definition 24), the $\mathcal{ZK}$ simulator has two advantages over a plain model $\mathcal{ZK}$ simulator. Namely,

- The simulator can see what values parties query the oracle on.
- The simulator can answer these queries in whatever way it chooses, as long as the answers "look" random.

The definition of deniable $\mathcal{ZK}$ in the RO model restricts the power of the simulator and *only* allows it to see on what values the parties query the oracle (thus out of the two advantages only the first remains). This is due to the fact that in the definition of deniable $\mathcal{ZK}$ in the RO model the distinguisher is given access to the random oracle and can thus verify if the simulator has answered the oracle queries in accordance to the pre-specified oracle. We, however, use this first advantage in a novel fashion, and show that it alone is an extremely powerful. More specifically, we employ the random oracle to construct commitment schemes where the simulator, gaining access to all oracle calls, will be able to extract the committed values, without rewinding the committer.

Lower bounds As a sanity check to the definition we start by showing the impossibility of non-interactive deniable $\mathcal{ZK}$ arguments for non-trivial languages.

Theorem 12 *Assume there exists a one-round deniable zero-knowledge argument for the language $L \in \mathcal{NP}$ in the RO model. Then, $L \in \mathcal{BPP}$.*

Proof Let R_L be a witness relation for the language $L \in \mathcal{NP}$. Suppose that there exists a one-round deniable zero-knowledge argument for L in the RO model. Then there exists a simulator S for the honest verifier such that for large enough x the following ensembles are indistinguishable by a distinguisher having access to RO:

- $\{P^{RO}(x, y_x)\}_{x \in L}$ for arbitrary $y_x \in R_L(x)$
- $\{S^{RO}(x)\}_{x \in L}$

where RO is a random variable uniformly distributed in $\{0,1\}^{poly(|x|)} \to \{0,1\}^{poly(|x|)}$.

Using the same argument as in the proof of Theorem 9, it follows the language L can be decided by S with the same soundness bound as that of Π's and with a completeness bound that is only negligibly different. It remains to show that S, which is an expected polynomial time machine can be turned into a strict polynomial time machine that decides the language. This follows using a standard argument by applying Markov's inequality and the union bound, which concludes so $L \in \mathcal{BPP}$. ■

Upper bounds On the positive side we show that 2 rounds are necessary to construct efficient and "robust" deniable $\mathcal{ZK}$ protocols for $\mathcal{NP}$. In fact we construct a protocol that is both concurrent zero-knowledge and concurrent proof of knowledge. We furthermore show that the protocol can be constructed through an efficient transformation from any special-sound honest-verifier zero-knowledge (HVZK) public-coin proof. We start by briefly outlining the construction, and thereafter proceed to a more formal treatment.

Outline of the Construction of 2-round Deniable ZK Arguments

On a very high level the protocol follows the paradigm of Feige-Shamir [30]. The verifier start by sending a "challenge" and a $\mathcal{WH}$ argument of knowledge of the answer to the challenge, to the prover. The prover thereafter shows using a $\mathcal{WI}$ argument of knowledge that either it possesses a witness for the statement it wishes to prove or that it possesses the answer to the challenge.

The difficulty in constructing such a protocol lies in the fact that each of these two steps must be implemented in a single message.[5] We outline the construction of these proof systems below.

Extractable Commitments Our main technical ingredient is the notion of straight-line extractable commitments in the RO model (see section 4.3.1). On a high level, these are commitments where the value committed to can be extracted in polynomial-time without the use of rewinding techniques. We construct such commitment schemes by letting the committer use the random oracle to commit. It follows from the random properties of the oracle that the committer, in order to succeed in opening a commitment, must have applied the oracle on it. This means that by simply observing all the random oracle queries made by committer, the committed values can be extracted without rewinding the committer.

[5]It is actually sufficient that the first step is implemented using a single message. The second step could conceivable be implemented using 2 rounds (see [54]). Nevertheless, our construction implements both steps using one-round solutions.

One-round witness extractable proofs Having established the powerful tool of straight-line extractable commitments, in section 4.3.2 we construct a one-round straight-line witness extractable $\mathcal{ZK}$ argument for $\mathcal{NP}$ in the RO model. Straight-line witness extraction means that a witness to the statement proved can be extracted without rewinding the prover. On a high-level, we do this by implementing the commitment scheme in the GMW protocol [36] with a straight-line extractable commitment scheme and thereafter applying the Fiat-Shamir transformation [31] [6] to "collapse" down the GMW protocol to a one-round (i.e., non-interactive) zero-knowledge argument in the RO model (see Lemma 5).

Finally, Lemma 1 and Lemma 2 can be applied to show that the one-round $\mathcal{ZK}$ protocol obtained is both $\mathcal{WH}$ and $\mathcal{WI}$ in the RO model.

In order to obtain an efficient protocol, in Lemma 10, we show how to transform any special-sound HVZK public-coin proof (so called Σ-protocols) into a one-round, $\mathcal{WH}$ and $\mathcal{WI}$, straight-line witness extractable argument in the RO model. Essentially, this is done by first transforming the Σ-proof into a cut-and-choose proof, and thereafter applying the same transformation as was done for the GMW protocol.

Putting it all together In section 4.3.3 we finally put everything together to obtain the 2-round deniable $\mathcal{ZK}$ argument for NP (see Theorem 13). In order to obtain an efficient protocol, we here rely on the OR transformation of [19] to implement the second message of the protocol.

We mention that some technical problems related to the malleability [24] of the commitments arise in the security proof. Nevertheless, since we have access to a random oracle these problems can be resolved in a rather straightforward manner.

4.3.1 Straight-line Extractable Commitments

We construct efficient commitment schemes with strong properties, which are proven secure without allowing the simulator to "choose" (or program) the random oracle. We start by defining the notion of straight-line extractable commitments schemes in the RO model. For simplicity we only state the definition for non-interactive commitment schemes.

Definition 39 *Let a (C, R) be a non-interactive commitment scheme for strings of length n. We say that (C, R) is straight-line extractable in the RO model if there exists a polynomial time (deterministic) extractor machine E such that for every PPT malicious committer C^* and every auxiliary input z to C^*, the following holds:*

- *Let c denote the output of $C^{*RO}(z)$ during the commit phase, and let l be a list of all the random oracle queries performed by $C^*(z)$ during and before the commit phase, including their answers.*

- *If $C^{*RO}(z)$ succeeds in decommitting to x with non-negligible probability during the reveal phase, then $E(c, l) = x$ with overwhelming probability (over the choice of RO).*

Remark 16 *Note that the extractor E is not given access to the random oracle, but instead receives both a list of the queries to the random oracle, as well as the answers to those queries.*

When having access to a random oracle it is easy to construct efficient commitment schemes that are straight-line extractable. Let l be a super-logarithmic polynomially bounded function, i.e., $\omega(log(n)) \leq l(n) \leq poly(n)$, and $RO : \{0,1\}^{2n} \rightarrow \{0,1\}^{l(n)}$ be a random oracle. Consider the following commitment scheme:

Protocol Com **- A Non-interactive Extractable Commitment Scheme**

Commit Phase:

- To commit to value $v \in \{0,1\}^n$, the sender uniformly selects $s \in \{0,1\}^n$ and sends the value $c = RO(v, s)$.

Reveal Phase:

- The sender reveals v and s.
- The receiver accepts if $c = RO(v, s)$ where c is the receiver's view of the commit phase.

Lemma 7 *$SLCom$ is a straight-line extractable non-interactive commitment scheme in the RO model.*

Proof

Computational Hiding The computational hiding property follows from the random properties of the the random oracle. Note that an adversary will only be able to distinguish commitments to two values if he is able to query the random oracle on a value that maps to the commitment. However, an adversary using $T(n)$ oracle queries only has a probability of (at most) $\frac{T(n)}{2^{l(n)}}$ of finding such a value. We conclude that a polynomially bounded adversary only has a negligible probability of distinguishing between two commitment.

Computational Binding In order to break the computational binding property of the commitment scheme an adversary needs to find a collision in the RO. An adversary using $T(n)$ oracle queries will fail in doing so with the following probability:

$$(1-\frac{1}{2^{l(n)}-1})(1-\frac{1}{2^{l(n)}-2})..(1-\frac{1}{2^{l(n)}-(T(n)-1)})>$$
$$(1-\frac{1}{2^{l(n)}-T(n)})^{T(n)}\approx 1-\frac{T(n)}{2^{l(n)}-T(n)}>1-2^{-l(n)/2}$$

which is negligible if $T(n)$ is a polynomial.

Straight-line Extraction Upon receiving the value c and the list l the extractor E proceeds as follows. E goes through the list $l=\{(x_i,r_i),c_i\}_{i=1..poly(n)}$ and checks if there is an index i such that $c_i=c$. If so it returns x_i, and otherwise nothing. We show that if there exist a committer C^* that succeeds in revealing to a value x with non-negligible probability, then except with negligible probability, he must have applied the RO to the value x during the commit phase. In fact, a cheating that has not applied the RO random oracle on the value committed to, has a probability of at most $\frac{T(n)}{2^{l(n)}}$, where $T(n)$ is the number of oracle calls during the decommit phase, of decommitting. ■

We note that SLCom can be used either as a statistically binding or a statistically hiding commitment scheme, depending on the parameter l.

Lemma 8 *If $l(n)=4n$ then SLCom is a statistically binding non-interactive commitment scheme in the RO model.*

Proof The probability that all the 2^{2n} different random oracle queries yield different answers is:

$$(1-\frac{1}{2^{4n}-1})(1-\frac{1}{2^{4n}-2})..(1-\frac{1}{2^{4n}-(2^{2n}-1)})>$$
$$(1-\frac{1}{2^{4n}-2^{2n}})^{2^{2n}}\approx 1-\frac{2^{2n}}{2^{4n}-2^{2n}}>1-2^{-n}$$

Thus, except with negligible probability (over the choice of RO) the commitment scheme is perfectly binding. ■

Lemma 9 *If $l(n)=n/8$ then SLCom is a statistically hiding non-interactive commitment scheme in the RO model.*

Proof We want to show that if r_0,r_1 are random variables uniformly distributed in $\{0,1\}^n$, then for all $x_1,x_2\in\{0,1\}^n$, $\Delta(x_1,x_2)=$

$$\frac{1}{2}\sum_{c\in\{0,1\}^{n/8}}|Pr[r\in^R\{0,1\}^n,RO(x_1,r)=c]-Pr[r\in^R\{0,1\}^n,RO(x_2,r)=c]|$$

is negligible with overwhelming probability (over the choice of RO). Consider the random variable $X_c^x = Pr[r \in^R \{0,1\}^n, RO(x,r) = c]$. (It is a random variable since RO is a random variable). We start by computing the variant of X_c^x: Let $X_{c,r}^x$ denote the random variable such that $X_{c,r}^x = 1$, if $RO(x,r) = c$, and 0 otherwise. Then $X_c^x = \frac{1}{2^n} \sum_r X_{c,r}^x$. Thus

$$V(X_c^x) = (\frac{1}{2^n})^2 V(\sum_r X_{c,r}^x) = \frac{1}{2^{2n}} \sum_i V(X_{c,r}^x)$$

since $X_{c,r}^x$ are independent for different r. Now

$$V(X_{c,r}^x) = E((X_{c,r}^x)^2) - E(X_{c,r}^x)^2 = E(X_{c,r}^x) - E(X_{c,r}^x)^2 = 2^{-n/8} - 2^{-n/4}$$

which gives that

$$V(X_c^x) = 2^{-n}(2^{-n/8} - 2^{-n/4}) < 2^{-9n/8}$$

We can now apply Chebyshev's Inequality, yielding

$$Pr(|X_c^x - E(X_c^x)| > 2^{-n/4}) < 2^{n/2} V(X_c^x) < 2^{-5n/8}$$

Now the probability that for none of the $c \in \{0,1\}^{n/8}$, $|X_c^x - E(X_c^x)| > 2^{-n/4}$ is

$$(1 - 2^{-5n/8})^{2^{n/8}} \approx 1 - 2^{-n/2}$$

Thus with overwhelming probability, since none of the terms in the sum for $\Delta(x_1, x_2)$ will be greater than $2^{-n/4}$,

$$\Delta(x_1, x_2) < 2^{n/8} 2^{-n/4} = 2^{-n/8}$$

which concludes the proposition. ■

Extractable Commitments with Oracle Restricted to a Prefix.

In the sequel we will construct a 2-party protocol where both parties use a commitment schemes. A problem that often arises in such scenarios is that one of the parties might be able to construct a commitment to a value that *depends* on the value committed to by the other party. When this is the case, we say that a party is able to *maul* a commitment that he is receiving. Commitment schemes for which this kind of attack is not possible are called *non-malleable* [24]. In order to obtain multiple commitments that are non-malleable with respect to each other we generalize the notion of straight-line extractability. We say that a commitment scheme in the RO model is *straight-line extractable with oracle restricted to the prefix s* if the commitment scheme is straight-line extractable and there exists an extractor that succeeds in extracting witnesses using only oracle queries that begin with the prefix s. We will in the following let different parties use different prefixes allowing for individual extraction of the committed values, and thus assuring the non-malleability of the commitments.

We note that SLCom can be changed in a straight-forward manner to become straight-line extractable with oracle restricted to the prefix s, by simply concatenating the string s to the oracle queries, i.e., $RO(s, x, r)$ becomes a commitment to the string x, where $RO : \{0,1\}^{2n+|s|} \rightarrow \{0,1\}^{l(n)}$.

4.3.2 Straight-line Witness Extractable Proofs

All previously known proofs of knowledge in the RO model (e.g. [64]) relied on rewinding and could therefore not be applied to simultaneously extract polynomially many witnesses. We introduce a stronger notion of proofs of knowledge in the RO model, namely proofs where witnesses can be extracted without rewinding the prover. More formally,

Definition 40 *We say that an interactive proof (P, V) for the language $L \in \mathcal{NP}$, with the witness relation R_L, is straight-line witness extractable in the RO model if there exists a PPT witness extractor machine E such that for every PPT machine P^*, every $x \in L$ and every auxiliary input $y \in \{0,1\}^{poly(|x|)}$, the following holds: Let* view *denote the view of V in the interaction $(P^{*RO}(x, y), V(x))$, and let l be a list of all oracle queries posed by $P^{*RO}(x, y)$ and $V(x)$, including the answers to the queries. Then, except with non-negligible probability (over the random coins of P, V and RO), it holds that if $\langle P^*(x, y), V(x)\rangle = 1$ then $E(\text{view}, l) \in R_L(x)$*

We show two constructions to achieve efficient straight-line witness extractable arguments in the RO model. First, we show how the GMW [36] protocol for proving the existence of a 3 coloring to a graph directly can be turned into a straight-line witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$, one-round argument in the RO model, by applying the Fiat-Shamir transformation [31] to "collapse" it down to one round, and using straight-line extractable commitments. Secondly, we show how to transform any three-round special-sound HVZK public-coin proof into a straight-line witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$, one-round argument. The second construction is of interest as it allows us to construct efficient protocols without going through Cook's transformation.

An Argument System for Graph-3-Coloring.

We recall the three-round zero-knowledge protocol of GMW (Goldreich, Micali, Wigderson) [36]:

Protocol Π (GMW's Graph 3-coloring proof)

Common Input: a directed graph $G = (V_G, E_G)$, with $n = |V_G|$

Auxiliary input to the prover: a 3-coloring of G, $c_0, c_1, .., c_n \in \{1, 2, 3\}$.

P uniformly chooses a permutation π over 1,2,3.

P $\rightarrow$ V: Commits to $\pi(c_0), \pi(c_1), .., \pi(c_n)$ using any statistically binding commitment scheme.

V $\rightarrow$ P: Uniformly selects an edge $(i, j) \in E$.

P $\rightarrow$ V: Reveals c_i, c_j. V checks that c_i and c_j are different colors.

As is shown in [6], the protocol can be "collapsed" down to a one-round $\mathcal{ZK}$ argument, Π', in the RO model by running $t = 2n * |E_G|$ parallel versions of the protocol and applying the RO to all the t first messages, to "simulate" the honest verifier. This transformation is called the Fiat-Shamir transformation [31].

Protocol Π' - "Collapsing" Π into a one-round protocol

P $\rightarrow$ V: $a' = a'_1, a'_2, .., a'_t$, $c' = c'_1, c'_2, .., c'_t$.

V checks that for all $1 \leq i \leq t$, $(a'_i, RO(a')_i, c'_i)$ is an accepting execution of the protocol Π, where $RO(a')_i$ signifies the i'th part of the random oracle's reply, such that each part has the appropriate size of the verifiers challenge in protocol Π.

Since Π' is zero-knowledge in the RO model it follows, by Lemma 1 and 2, that Π' is also $\mathcal{WH}$ and $\mathcal{WI}$. Now, if the commitment scheme chosen has the property of being straight-line extractable, the resulting protocol is straight-line witness extractable.

Proposition 5 *If the protocol Π' is instantiated with a straight-line extractable commitment scheme, then the resulting protocol is straight-line witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$ in the RO model.*

Proof It only remains to show that Π' is straight-line witness extractable. Let E be the straight-line extractor to the commitments scheme used. We construct an extractor E' for the protocol Π'. E' proceeds as follows, on input a view v and a list l of oracle queries.

- E' uses E to extract all the values committed to in the view v (i.e., all the graph colorings committed to by the prover). Formally, for each commitment in the view v, E' feeds the view of the commitment as well as the list l to E. If E fails in extracting a value, E' uses the color "1" for that vertex.

- If the honest verifier V would reject the view v, E' halts, outputting nothing. Otherwise, E returns the first correct graph coloring that it finds. If none are found it outputs `fail`.

We show that E' is a valid straight-line extractor for the protocol Π'. Assume, for contradiction, that there exists a polynomial $p(n)$ such that for infinitely many n there exists values $x \in \{0,1\}^n \cap L$ and $y \in \{0,1\}^{poly(n)}$ for which

$$\Pr[(P^{*RO}(x,y), V^{RO}(x)) = 1 \wedge E(\text{view}, l) \notin R_L(x)] > p(n)$$

where view denotes the view of V in the interaction $(P^{*RO}(x,y), V^{RO}(x))$, and l is a list of all oracle queries posed by $P^{*RO}(x,y)$, including their answers.

Now, consider the malicious prover P^{**} that proceeds just as P^*, except that before outputting the proof, P^{**} checks if E succeeds in extracting the witness (note that since the protocol is non-interactive E only needs P^{**}'s oracle calls to perform its extraction). If E succeeds in the extraction then P^{**} outputs $\perp$ and otherwise it outputs what P^* would have output. It follows that:

- $\Pr[E(\text{view}', l') \in R_L(x)] = 0$, where view$'$ denotes the view of V in the interaction $(P^{**RO}(x,y), V^{RO}(x))$, and l' is a list of all oracle queries posed by $P^{**RO}(x,y)$, including their answers.

- $\Pr[(P^{**RO}(x,y), V^{RO}(x)) = 1 > p(n)$

Using an argument of Goldreich and Krawczyk [34] it can be seen that there thus exists a prover P'^{**RO} for the "non-collapsed" protocol (i.e. a parallelized version of the 3-round GMW protocol), such that with probability $O(p(n)/T(n))$, where $T(n)$ is a bound on the number of oracle queries posed by the prover, P'^{**RO} succeeds in convincing the honest verifier, without E' being able to extract the graph coloring. (Roughly, P'^{*RO} is constructed as follows. P'^{*RO} picks a random number i between 1 and $T(n)$, and thereafter runs P^{**RO}. When P^{**RO} queries the oracle the i'th time, P^{**RO} externally forwards the query to the outside verifier, instead of querying the random oracle. It can be seen that with probability $O(1/T(n))$, P'^{**RO} picks the "challenge" that is used in the proof output by P^{**RO}.) This means that, with a certain polynomial probability, P'^{**RO} is able to convince the honest verifier of the 3-round protocol. Furthermore it holds that E always fails in extracting a witness from P'^{**RO}. Also note that it follows from the extractability property of the commitment scheme that, except with negligible probability, P'^{**RO} will only be able to open up the commitments sent in the first round to the values extracted by E. This means that, except with negligible probability, the probability that the honest verifier of the 3-round protocol accepts the proof is

$$(1 - \frac{1}{|E_G|})^{2n|E_G|} \approx \frac{1}{e^{2n}} < \frac{1}{2^{2n}}$$

for large n, which contradict the fact that P'^{**RO} succeeds with polynomial probability. ∎

Remark 17 *We note that the above protocol has a soundness error of 2^{-n}. If we only care about having a negligible soundness error, it is actually enough to only consider $\omega(logn)$ parallel repetitions of the protocol of GMW.*

A Transformation from HVZK Protocols.

Suppose $\Pi = (a, b, c)$ is a three round special-sound HVZK public-coin proof for the language $L \in \mathcal{NP}$. In order to achieve a one-round witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$ argument for L we transform the protocol Π into a cut-and-choose protocol Π' and thereafter use the same transformation as was done in the case of the proof of Graph-3-Coloring. Consider the following protocol:

Protocol Π' : A cut and choose variant of Π

P $\rightarrow$ V: a, two different random numbers $b_0, b_1 \in B$, commitments to c_0, and c_1 where c_i is the answer to the query b_i with a as first message in the protocol Π

V $\rightarrow$ P: chooses q randomly from $\{0, 1\}$

P $\rightarrow$ V: Decommits to c_q

V checks that (a, b_q, c_q) is a consistent execution of the protocol Π.

Now, let Π'' be the protocol obtained after applying the Fiat-Shamir transformation on Π', i.e., running $2n$ versions of the protocol in parallel, and simulating the verifier's challenge by applying the random oracle to the first message:

Protocol Π'' : A "collapsed" version of Π'

P $\rightarrow$ V: $a' = a'_1, a'_2, .., a'_t$, $c' = c'_1, c'_2, .., c'_t$

V checks that for all $1 \leq i \leq 2n$, $(a'_i, RO(a')_i, c'_i)$ is an accepting execution of the protocol Π', where $RO(a')_i$ signifies the i'th bit of the random oracle's reply.

Lemma 10 *If the protocol Π'' is instantiated with a straight-line extractable commitment scheme, then the resulting protocol is a straight-line witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$ argument for L in the RO model.*

Proof Completeness of the protocol Π' follows directly from the completeness of Π. Secondly, the special soundness condition tells us that answers to two different queries b_1 and b_2 to the same first message a, yields a witness to the assertion being proved. Therefore, the protocol Π' has the same form as the GMW graph 3-coloring proof, in the sense that the first message contains a commitment to a witness of the statement proved. The same proof as in Section 4.3.2 can thus be used to show that Π'' is a witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$, one-round argument for the language L. (Since Π' has soundness error $1/2$ only $2n$ parallel repetitions are sufficient to get a non-interactive proof with soundness error 2^{-n}.) ■

Remark 18 1. *We note that if we only care about obtaining a proof with negligible soundness error, it is enough to consider only $\omega(logn)$ parallel repetitions, instead of $2n$.*

2. *Note that our transformation actually turns the 3-round HVZK protocol into a, so called, "cut-and-choose" protocol. It is an interesting open question to find a transformation that does not entrain a similar overhead.*

Witness Extraction by an Oracle Restricted to a Prefix.

As with the commitments schemes, the above mentioned protocols can easily be turned into arguments that are witness extractable by an oracle restricted to a certain prefix, by using commitment schemes that are straight-line witness extractable by an oracle restricted to the prefix.

4.3.3 Deniable Concurrent Zero-knowledge Proof of Knowledge

In this section we use the witness extractable, $\mathcal{WH}$ and $\mathcal{WI}$, one-round arguments in a way similar to the Feige-Shamir construction [30] to construct a 2-round straight-line simulatable deniable $\mathcal{ZK}$ argument of knowledge for $\mathcal{NP}$ in the RO model.

Since the protocol is straight-line simulatable it is also deniable concurrent zero-knowledge:

Theorem 13 *Assume the existence of one-way functions. Then, there exists a two-round deniable concurrent zero-knowledge argument for languages in $\mathcal{NP}$ in the RO model. Furthermore the argument is both straight-line witness extractable, and straight-line simulatable.*

Proof Let $f : \{0,1\}^n \rightarrow \{0,1\}^{poly(n)}$ be a one-way function, and let the witness relation $R_{L'}$, where $(x,y) \in R_{L'}$ if $f(x) = y$, characterize the language L'. Let $RO : \{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$ be a random oracle, and the language $L \in \mathcal{NP}$. Consider the following protocol for proving that $x \in L$:

Protocol SLZK - A Two-round Straight-line Simulatable ZK Argument

Common Input: an instance x, security parameter 1^n.

V chooses a random number $r \in \{0,1\}^n$.

V $\rightarrow$ P: $c = f(r)$, a one-round $\mathcal{WH}$ WH, straight-line witness extractable, by oracle restricted to prefix "0", argument of the statement

"$\exists r'$ s.t $c = f(r')$" for the witness relation $R_{L'}$.

P $\rightarrow$ V: a one-round $\mathcal{WI}$, straight-line witness extractable, by oracle restricted to prefix "1", argument of the statement

"$\exists r'$ s.t $c = f(r') \vee x \in L$" for the witness relation $R_{L \vee L'}(c,x) = \{(r',w) | r' \in R_{L'}(c) \vee w \in R_L(x)\}$.

Completeness of the protocol is clear.

Soundness In order to prove soundness, we start by noting that the prover sends an argument that is straight-line witness extractable by oracle restricted to prefix "1". But since the honest verifier has not used the oracle with prefix "1", a witness can be extracted using only the prover's oracle queries. If a malicious prover succeeds in convincing the honest verifier, he must thus have either an r' s.t $c = f(r')$ or a witness for $x \in L$. We will show that the prover needs to have a witness for x: Let the probability ensemble U be uniform on $\{0,1\}^n$, and let $X = f(U)$ be a probability ensemble for the language L'. Then since f is a one-way function, X is a hard instance ensemble. Now, if the prover, after having received the verifier's first message was able to a find a witness to a randomly chosen instance in the hard-instance ensemble X, this would violate the witness hiding property of the verifier's message. The claim that the prover must have a witness for x follows. The protocol is thus straight-line witness extractable for the statement $x \in L$. Soundness follows automatically.

Zero-knowledge We show that the protocol is straight-line simulatable. We construct a simulator that proceeds as follows. The simulator simply extracts r from the verifier's first message and then uses it as a "fake" witness to send its proof. If the simulator fails in extracting the witness from a proof that the honest verifier would had accepted, it outputs `fail`. We proceed to show that the simulators output is indistinguishable from the honest prover's. Towards this goal we define a hybrid simulator S' which receives the real witness. S' proceeds just as S, but instead of sending a proof using the fake witness S' uses the real witness. It follows from the witness indistinguishability property of the second message that the output of S and S' are indistinguishable. Now, note that the only difference between S' and the honest prover is that S' outputs `fail` sometimes, while the honest prover would complete a proof. Recall that S' only outputs `fail` when the extraction failed for a proof that the honest verifier would accept. It follows from the definition of straight-line witness extractability that the extraction only fails with negligible probability. We conclude the output of the honest prover and S' are indistinguishable, which in turn means that the output of the honest prover and S are indistinguishable. ■

Remark 19 1. *We note that since the protocol is straight-line witness extractable it is also witness extractable under concurrent executions, i.e., witnesses to all concurrent executions can be simultaneously extracted. Indeed, this feature is of great importance in, for example, identification schemes. We note that it was previously unknown how to simultaneously extract witnesses from polynomially many proofs in the RO model.*

2. *Note that even though we have access to a random oracle we need to rely on the existence of one-way functions since our protocol uses the one-way function in a non-black box way (by applying Cook's transformation on the function).*

Efficient implementation We show how to give an efficient instantiation of the protocol SLZK. Let $f : \{0,1\}^n \to \{0,1\}^{poly(n)}$ be a one-way function, and let Π' be a special-sound HVZK public-coin argument for proving the knowledge of a pre-image to f. Such argument systems exists for every one-way function, by reducing the one-way function to an instance of the graph hamiltonicity problem, using Cook's theorem, and thereafter using Blum's protocol [7]. We emphasize, however, that if a specific one-way function is used, the HVZK argument can be tailored for the function to get an efficient implementation. Examples of such protocols are the Guillou-Quisquater scheme [41] for the RSA function, and the Schnorr scheme [63] for the discrete logarithm.

To implement the first message of the protocol, we use the transformation described in Section 4.3.2 to turn Π', i.e., the special-sound HVZK zero-knowledge argument for L', into the needed one-round argument for L'.
The second message is implemented as follows: Assuming that we have a special-sound HVZK public-coin argument for L, we can use the efficient OR transformation in [19] to yield a special-sound HVZK public-coin argument for $L \vee L'$ and the witness relation $R_{L \vee L'}$.[6] We can thereafter apply the transformation in section 4.3.2. (Note that although the protocol is constructed through a quite efficient transformation from any special-sound HVZK argument, the transformation turns the HVZK protocol into a "cut-and-choose" protocol, which induces a blow up in communication complexity.)

4.3.4 An efficient 4-round proof

In this section we show the existence of a 4-round concurrent zero-knowledge proof in the RO which is more efficient considering communication complexity than the 2-round protocol described in Section 4.3.3. In fact, we show a transformation from any public-coin HVZK proofs to a concurrent $\mathcal{ZK}$ proof in the RO, which only induces a small *additive* overhead in communication complexity. We furthermore note that whereas the transformation in Section 4.3.3 turned the HVZK proof into an argument, the transformation of this section does not do so.

Theorem 14 *Assume there exists a three-round HVZK public-coin zero-knowledge proof $\Pi = (a, b, c)$ for the language $L \in \mathcal{NP}$. Then there exists a four-round concurrent deniable zero-knowledge proof Π' for the L in the RO model. Furthermore, the protocol Π' uses only an extra communication complexity of $2|b|$.*

Proof Sketch: Suppose $\Pi = (a, b, c)$ is a three-round honest-verifier public-coin zero-knowledge proof (or argument) for the language L:

Protocol Π:
P $\rightarrow$ V: a
V $\rightarrow$ P: a random value $b \in B$
P $\rightarrow$ V: c
where $B \in \{0, 1\}^{poly(n)}$

We construct a four-round protocol Π' that proceeds in two phases:

- In phase one, the prover and the verifier perform a coin-tossing, using straight-line witness extractable commitments (the use of straight-line extractable commitments makes it possible for the simulator to "force" the outcome of the coin-tossing to any value).

[6] The resulting argument uses less communication than the argument for L plus the argument for L'.

- In phase two, the prover and the verifier execute the protocol Π using the outcome of the coin-tossing as the verifier's challenge. It is thus essential the that outcome of the coin-tossing only is revealed at the moment that the verifier is supposed to send its challenge.

More precisely,

Protocol Π' - transforming Π into a concurrent ZK proof

V $\rightarrow$ P: Commits to random number $r_1 \in B$, using a statistically hiding straight-line extractable commitment scheme.

P $\rightarrow$ V: a the first message in protocol Π, a random value $r_2 \in B$

V $\rightarrow$ P: decommits to r_1

P $\rightarrow$ V: c, such that $(a, r_1 \oplus r_2, c)$ would be an accepting execution of protocol Π

We give a brief sketch of the security of the protocol. Completeness of the protocol is easy to show. Soundness follows from the fact that: 1. the coin-tossing scheme is unconditionally secure against a cheating prover, since the verifier uses a statistically hiding commitment scheme, 2. the protocol Π is sound. A formal proof proceeds by transforming a cheating prover P'^* for Π' into one for Π, P^*. Roughly, P^* is constructed by internally incorporating P'^* and proceeding as follows:

- Start by feeding P'^* a commitment to random value.
- Upon receiving back (a, r_2) from P'^*, send a to the outside verifier.
- When receiving back a challenge b from the outside verifier, open up the commitment sent to P'^* to a value r_1 such that $b = r_1 \oplus r_2$. Note that since the protocol uses a statistically hiding commitment scheme, with overwhelming probability there exist a way of opening the commitment to any value.

Note that the machine P^* needs to run in exponential time in order to ”equivocate” the commitment. Furthermore, note that the views of P'^* in a real execution (with the honest verifier) and in the “emulated” execution by P^* is statistically close. It thus follows that that P^* succeeds in convincing the honest verifier of the protocol Π with roughly the same probability as P'^* succeeds in Π', contradicting the (unconditional) soundness of Π.

Zero-knowledge Since Π is HVZK there exists a simulator S that is able to simulate a transcript of an honest prover with verifier messages that look random. We construct a simulator S' for the protocol Π' that proceeds as follows.

- S' first runs S to produce a transcript (a, b, c)
- S' then starts communicating with the malicious verifier. Using the straight-line extractability property of the commitment scheme used by the verifier, S' is able to performs a coin-tossing such that the verifier's challenge equals b, and can thus start by sending the message a and answer the challenge send by the verifier, using c.

Note that output of the simulator S' will be identical to that of S if the malicious verifier V^* does not abort the protocol. On the other hand, if V^* does abort the protocol, then S' acts as the honest prover (and also aborts). We show that V^* abort with (roughly) the same probability in the simulated and in the real execution. In fact, this follows from the fact that the first message (sent by the prover) in simulated execution is indistinguishable from the first message sent in the real execution (due to the HVZK property of Π). (Note that if Π was *statistical* HVZK, then these distributions would be statistically close).

We finally note that the simulator S' is straight-line and the protocol Π' is thus concurrent deniable $\mathcal{ZK}$ in the RO model.

Overhead In Section 4.3.1 we showed the existence of straight-line witness extractable statistically hiding commitment schemes, in the RO model, where the commit phase use the same communication complexity as the strings committed to, and reveal phase uses the twice the communication complexity. If using such a commitment scheme, the overall communication complexity of the protocol is $|b| + (|a| + |b|) + (2|b|) + |c| = (|a| + |b| + |c|) + 2|b|$. Thus, Π' only uses an extra communication complexity of $2|b|$. ■

Remark 20 *Note that if* Π *is a statistical HVZK proof, then the resulting protocol* Π' *is a statistical deniable* $\mathcal{ZK}$ *proof in the RO model.*

Remark 21 *Interestingly the above protocol does not rely on any other assumption than the random oracle. This is in contrast with the 2-round protocol of Section 4.3.3 which relies on the existence of one-way functions.*

Open problems

The most urgent open problem is to find a more efficient construction of one-round witness extractable arguments that do not rely on cut-and-choose techniques. Secondly, our 2-round protocol relies on the existence of one-way functions, while our 4-round protocol does not. We wonder if it is possible to construct 2-round straight-line simulatable deniable $\mathcal{ZK}$ protocols without any further assumptions than the random oracle.

4.4 On Unreplayabable Zero-Knowledge Protocols

In many settings it is reasonable to assume that honest parties only communicate using a certain class of pre-specified protocol. In this section we investigate the possibility of constructing $\mathcal{ZK}$ protocols, in shared object models, that retain the spirit of "standard" $\mathcal{ZK}$ proofs, under the above assumption on the behavior of honest parties. In other words, we investigate the possibility of $\mathcal{ZK}$ protocols in shared object models that have the property that a verifier of a $\mathcal{ZK}$ proof will not be able to do anything it could not have before the interaction, *using a certain class of protocols.*

More precisely, we say that a class C of protocols is *closed under unreplayability* if an adversary, that has interacted with a prover using a protocol in C and the shared object R, will not be able to prove any statement x using a protocol in C with the shared object R, unless he could not have already done so without the help of the prover. Note that the adversary is allowed to interact with a prover showing the statement x, but is not necessarily restricted to provers showing x. The only restriction on the communication of the adversary is that in both interactions *only protocols in C* are used.

Note that the definition of unreplayability *does not* necessarily imply that a verifier of an unreplayable proof system will not "learn" anything from the protocol execution (which is what the notion of deniable $\mathcal{ZK}$ satisfies). Rather, it means that the verifier will not be able to "convey" this information using a "valid" protocol. Unreplayability therefore only achieves our goals of deniability in a limited way (an example of this is given in section 4.4.3).

4.4.1 Definition of Unreplayability and Some Consequences

Since unreplayability is a notion related to witness hiding, we use a similar definition. We, thus, start by generalizing the notion of a hard instance ensemble for a relation R_L, to the notion of a *hard instance ensemble for an interactive proof (P, V)*:

Definition 41 *Let (P, V) be an interactive proof for the language $L \in \mathcal{NP}$ and $X = \{X_n\}_{n\in \mathrm{N}}$ a probability ensemble such that X_n ranges over $L \cap \{0,1\}^n$. We say that X is* hard *for (P, V) if for every expected polynomial time probabilistic algorithm F, all $z \in \{0,1\}^{poly(n)}$, the probability*

$$Pr[\langle F(X_n, z), V(X_n)\rangle = accept]$$

is negligible (as a function of n).

Unreplayability can now be defined in analogy with witness hiding:

Definition 42 *Suppose R is a random variable uniformly distributed in either $1^{poly(n)} \to \{0,1\}$ or $\{0,1\}^{poly(n)} \to \{0,1\}^{poly(n)}$, $C = \{(\Pi_1, L_1), (\Pi_2, L_2), ..\}$ is a class of tuples of zero-knowledge proof (or argument) protocols, in the CRS or*

random oracle model, and languages. A replay attack *against the class C and the protocol Π, for the language L, where $(\Pi, L) \in C$, using R as shared object, is a PPT oracle machine A such that for all $x \in L_i$, every $z \in \{0,1\}^*$, and the oracle O_R, $A^{R,O_R}(x,z) \to P_x^*$, where P_x^* is the code for a polynomial-sized circuit (which will act as a cheating prover). The oracle O_R is defined as follows:*
For all $(\Pi', L') \in C$, A can ask O_R to prove any statement y using the protocol Π' initiated with R as shared object. If $y \in L'$ then O_R follows the protocol Π', and if not it just answers "not in the language". Secondly, O_R can only be invoked once, and may not be rewound.

Remark 22 *1. The oracle O_R may only be invoked once since the notion of $\mathcal{ZK}$ is not necessarily closed under sequential composition in the CRS, or random oracle model.*

2. We do not allow the adversary simultaneous access to the oracle and the verifier, since then a man-in-the-middle attacker would always succeed in such a scenario.

Definition 43 *Let R and C and O_R be as in the previous definition. We say that a* replay attack *A* succeeds *on the class C using R as shared object, if there exists a hard instance ensemble $\{X_n\}_{n\in\mathrm{N}}$ for the interactive proof Π such that for infinitely many n there exists a $z \in \{0,1\}^{poly(n)}$ such that*

$$Pr[A^{R,O_R}(X_n, z) \to P_{X_n}^{*R}, \langle P_{X_n}^{*R}, V^R(X_n)\rangle = accept]$$

is non-negligible, where V is a PPT honest verifier following the protocol Π.

Definition 44 *A class of tuples of zero-knowledge proof (or argument) protocols and languages C is* closed under replayability *if it resists all replay attacks against the class C and every protocol Π and language L such that $(\Pi, L) \in C$.*

Definition 45 *We say that a zero-knowledge proof (or argument) Π for the language L is* unreplayable, *if the class $C = \{(\Pi, L)\}$ is closed under unreplayability.*

Remark 23 *We note that all proofs for languages in $\mathcal{BPP}$ are trivially unreplayable, since those protocol do not have any hard instance ensembles.*

We end this section by noting a relationship between hard instance ensembles for witness relations, and hard instance ensembles for interactive proofs.

Lemma 11 *Suppose that $\{X_n\}_{n\in\mathrm{N}}$ is a hard instance ensemble for the interactive proof (or argument) (P, V), for the language $L \in \mathcal{NP}$, with efficient prover for the witness relation R_L for L. Then $\{X_n\}_{n\in\mathrm{N}}$ is a hard instance ensemble for R_L.*

Proof Let R_L be a witness relation for language $L \in \mathcal{NP}$, (P, V) an interactive proof for L, with efficient prover for the witness relation R_L, and $X = \{X_n\}_{n \in \mathrm{N}}$ a hard instance ensemble for the interactive proof (P, V). Suppose that X is not a hard instance ensemble for R_L, i.e. there exists an expected polynomial time probabilistic witness finding algorithm F, such that for infinitely many n, there exists a $z \in \{0,1\}^{poly(n)}$ such that

$$Pr[F(X_n, z) \in R_L(X_n)]$$

is non-negligible (as a function of n). It follows from the efficient prover condition of the interactive proof (P, V), that for infinitely many n there exists a $z \in \{0,1\}^{poly(n)}$ such that

$$Pr[\langle P(F(X_n, z)), V\rangle(X_n) = accept]$$

is non-negligible (as a function of n). Since P is a polynomial-time machine (due to the efficient prover condition), and since F is expected polynomial-time, their composition is thus an expected polynomial-time machine, which contradicts that X is a hard instance ensemble for (P, V). ■

4.4.2 On the Existence of Unreplayable Zero-knowledge

Following our intuition, non-interactive zero-knowledge arguments for $\mathcal{NP}$ are *not* unreplayable unless they do not contain any hard-instance ensembles:

Theorem 15 *Let Π be a one-round proof (or argument) for the language L in a model with shared objects. If Π is unreplayable, then there does not exist any hard instance ensembles for Π.*

Proof Suppose, for contradiction that there exists an unreplayable proof Π for the language L, which has the hard-instance ensemble X. Now, considering the following replay attack A for X: A, when asked to prove an instance x, simply asks the oracle for a proof of x. Upon receiving back the proof π from O, A outputs the code of a circuit P_x^* that simply outputs π. Note that P_x^* will thus succeed in producing an accepting proof of x with the same probability as the honest prover. It follows that the replay attack A succeeds on the protocol Π and the ensemble A, which contradicts the unreplayability property of Π. ■

There is, however, a natural class of zero-knowledge protocols that is closed under unreplayability:

Theorem 16 *Let $C = (\Pi_1, L_1), (\Pi_2, L_2), ..$ be a class of tuples of zero-knowledge proofs (or arguments) with efficient provers, in a model with shared objects, and languages, such that, for all i, Π_i is a proof system for L_i. If, for all i, Π_i is a proof (or argument) of knowledge in a model with shared objects, then C is unreplayable.*

Proof Assume that C is a class of arguments of knowledge, as specified in the theorem. Assume further that there exists a replay attack on the class C and the argument Π for the language L, where $(\Pi, L) \in C$, i.e there exists an adversary A, and a hard instance subset $X = \{X_n\}_{n \in \mathrm{N}}$ for Π such that for infinitely many n there exists a $z \in \{0,1\}^{poly(n)}$ such that

$$Pr[A^{R,O_R}(X_n, z) \to P^{*R}_{X_n}, \langle P^{*R}_{X_n}, V^R(X_n)\rangle = accept]$$

is non-negligible, where R and V are defined as in Definition 43. Since P^*_x communicates with V by means of a proof of knowledge, there exists an expected polynomial time probabilistic extractor machine E, and a witness relation R_L for L such that for all $x \in L$: $Pr[E(P^{*R}_x, x) \in R_L(x)]$ is non-negligible if $Pr[\langle P^{*R}_x, V^R(x)\rangle = accept]$ is non-negligible. This means that for infinitely many n there exists a $z \in \{0,1\}^{poly(n)}$ such that

$$Pr[A^{R,O_R}(X_n, z) \to P^{*R}_{X_n}, E(P^{*R}_{X_n}, x) \in R_L(x)]$$

is non-negligible. But since A communicates with O by means of a $\mathcal{ZK}$ protocol in a model with shared objects, there exists a PPT simulator machine S such that for infinitely many n there exists a $z \in \{0,1\}^{poly(n)}$ such that

$$Pr[S(X_n, z) \to (\sigma, P^{*\sigma}_{X_n}), E(P^{*\sigma}_{X_n}, X_n) \in R_L(x)]$$

is non-negligible, which shows that X is not a hard instance ensemble for R_L. This in turn contradicts the fact that X is a hard instance ensemble for the argument with efficient prover Π, by using lemma 11. ■

Remark 24 *Since we have assumed a fixed witness relation R_L for each language $L \in \mathcal{NP}$, it is implicit in the statement of the theorem that both the proof of knowledge and the efficient prover refer to the same witness relation.*

The conclusion of this section is, thus, that if honest-parties only communicate using arguments of knowledge that are zero knowledge in a model with shared objects, then the intuitive interpretation of zero-knowledge is preserved. When proving security of a fixed protocol this property is often sufficient. Nevertheless, in other, more complicated settings, where we want to guarantee security without any restrictions on the protocols, we need to resort fully deniable protocols, satisfying meta-definition 35.

4.4.3 Deniability is Stronger than Unreplayability

It could be tempting to believe that unreplayability for a protocol is a sufficient requirement for the protocol to be deniable. We show that deniability is a strictly stronger requirement.

Below we show the existence of a protocol that is unreplayable, but provably not deniable. In fact, a transcript of an interaction yields a proof of the assertion:

Protocol Σ

Common Input: a directed graph $G = (V_G, E_G)$, with $n = |V_G|$

Auxiliary input to the prover: a 3-coloring of G, $c_0, c_1, .., c_n \in \{1, 2, 3\}$.

P $\rightarrow$ V: Commits to $n = V \cdot E$ colorings of G using an equivocal bit-commitment scheme, Com'.

V $\rightarrow$ P: $(a_1, b_1), (a_2, b_2), .., (a_n, b_n)$ where a_i, b_i are the indexes of two random adjacent vertices, for $1 \leq i \leq n$.

P $\rightarrow$ V: For $1 \leq i \leq n$, P opens up the commitments for vertices a_i, b_i.

V checks that the vertices in each pair have different colors.

The bit-commitment scheme used is defined as follows:

Protocol Com' - A Non-interactive Equivocal (and Extractable) Commitment Scheme

Commit Phase:

- To commit to value $v \in \{0, 1\}$, the sender uniformly selects $s \in \{0, 1\}^n$ and sends the value $c = RO(r) + v$.

Reveal Phase:

- The sender reveals v and s.
- The receiver accepts if $c = RO(s) + v$ where c is the receiver's view of the commit phase.

It can be seen that the bit-commitment scheme Com' indeed is computationally hiding and binding, and that a simulator (that is allowed to chooses the random oracle) can open up the commitments both ways (i.e to both a 0 and a 1). In fact, this means that the protocol Σ, which clearly is complete, also is $\mathcal{ZK}$, since the simulator can just commit to a random coloring and thereafter open it up in such a way that the verifier will accept. We also note that Σ is a proof of knowledge (by simple rewinding). See [21] for proofs of the $\mathcal{ZK}$ and proof of knowledge properties for a very similar protocol (the protocol of [21] is essentially the same as Σ, except for the particular commitment scheme used).

Now since Σ is both $\mathcal{ZK}$ in the RO model, and a proof of knowledge in the RO model, it is also unreplayable. However, consider the malicious verifier V^* that, instead of randomly choosing the vertices to be opened up, applies the random oracle, RO to x and the commitments sent by P in the first round of the protocol. The transcript of the protocol is then a non-interactive ($\mathcal{ZK}$) argument (NIZK) in the random oracle model. The verifier can therefore clearly show something that it could not before, which certainly contradicts our intuition of a zero-knowledge proof. (More formally, suppose, of contradiction, that Σ is deniable $\mathcal{ZK}$. Let S be the simulator for the above-mentioned verifier V^*. It follows, from the $\mathcal{ZK}$ property, that for instances $x \in L$, S outputs an accepting NIZK argument. On the other hand, for instance $x \notin L$, the output of S will only be an accepting NIZK with negligible probability (by the soundness of the NIZK protocol [6]). S can thus be used to decide the language L, i.e., $NP = BPP$.)

4.5 Acknowledgments

First, I wish to thank Johan Håstad for his invaluable help and comments. I am also very grateful to Ran Canetti for long and helpful discussions. Thanks also to Shafi Goldwasser, Tal Rabin, Alon Rosen, Victor Shoup and the anonymous referees for helpful comments.

Bibliography

[1] B. Barak. How to go Beyond the Black-Box Simulation Barrier. In *42nd FOCS*, pages 106–115, 2001.

[2] L. Babai and S. Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *JCSS*, Vol. 36, pages 254–276, 1988.

[3] B. Barak and R. Pass. On the Possibility of One-Message Weak Zero-Knowledge. In *1st TCC*, pages 121–132, 2004.

[4] B. Barak and Y. Lindell. Strict Polynomial-Time in Simulation and Extraction. In *34th STOC*, pages 484–493, 2002.

[5] M. Bellare, R. Impagliazzo and M. Naor. Does Parallel Repetition Lower the Error in Computationally Sound Protocols? In *38th FOCS*, pages 374–383, 1997.

[6] M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *1st ACM Conf. on Computer and Communications Security*, pages 62–73, 1993.

[7] M. Blum. How to prove a Theorem So No One Else Can Claim It. *Proc. of the International Congress of Mathematicians,* Berkeley, California, USA, pages 1444–1451, 1986.

[8] M. Blum. Coin Flipping by Telephone. In *Crypto81*, ECE Report 82-04, ECE Dept., UCSB, pages 11–15, 1982

[9] M. Blum, P. Feldman and S. Micali. Non-Interactive Zero-Knowledge and Its Applications. In *20th STOC*, pages 103–112, 1988

[10] G. Brassard, D. Chaum and C. Crépeau. Minimum Disclosure Proofs of Knowledge. *JCSS*, Vol. 37, No. 2, pages 156–189, 1988. Preliminary version by Brassard and Crépeau in *27th FOCS*, 1986.

[11] R. Canetti. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[12] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *34th STOC*, pages 494–503, 2002.

[13] R. Canetti and M. Fischlin. Universally Composable Commitments. In *Crypto2001*, Springer LNCS 2139, pages 19–40, 2001.

[14] R. Canetti, O. Goldreich, S. Goldwasser and S. Micali. Resettable Zero-Knowledge. In *32nd STOC*, pages 235–244, 2000.

[15] R. Canetti, O. Goldreich and S. Halevi. The Random Oracle Methodology, Revisited. In *30th STOC*, pages 209–218, 1998

[16] R. Canetti, J. Kilian, E. Petrank and A. Rosen. Black-Box Concurrent Zero-Knowledge Requires (almost) Logarithmically Many Rounds. *SIAM Jour. on Computing*, Vol. 32(1), pages 1–47, 2002.

[17] R. Canetti, Y. Lindell, R. Ostrovsky and A. Sahai. Universally Composable Two-Party and Multi-Party Computation. In *34th STOC*, pages 494–503,2002.

[18] D. Chaum and H. van Antwerpen. Undeniable Signatures. In *Crypto89*, Springer LNCS 435, pages. 212–216, 1989.

[19] R. Cramer, I. Damgård and B. Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In *Crypto94*, Springer LNCS 839, pages. 174–187, 1994.

[20] G. di Crescenzo, G. Persiano and I. Visconti Constant-Round Resettable Zero Knowledge with Concurrent Soundness in the Bare Public-Key Model. In *Crypto04*, Springer LNCS 3152, pages. 237–253, 2004.

[21] I. Damgård. Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In *EuroCrypt2000*, LNCS 1807, pages 418–430, 2000.

[22] I. Damgård, T. Pedersen and B. Pfitzmann. On the Existence of Statistically Hiding Bit Commitment Schemes and Fail-Stop Signatures. In *Crypto93*, Springer-Verlag LNCS Vol. 773, pages 250–265, 1993.

[23] A. De Santis, G. Di Crescenzo, R. Ostrovsky, G. Persiano and A. Sahai. Robust Non-interactive Zero Knowledge. In *Crypto2001*, Springer LNCS 2139, pages 566–598, 2001.

[24] D. Dolev, C. Dwork and M. Naor. Non-Malleable Cryptography. *SIAM Jour. on Computing*, Vol. 30(2), pages 391–437, 2000.

[25] C. Dwork, M. Naor and A. Sahai. Concurrent Zero-Knowledge. In *30th STOC*, pages 409–418, 1998.

[26] C. Dwork and A. Sahai. Concurrent Zero-Knowledge: Reducing the Need for Timing Constraints. In *Crypto98*, Springer LNCS 1462 , pages 442–457, 1998.

[27] U. Feige. Ph.D. thesis, Alternative Models for Zero Knowledge Interactive Proofs. Weizmann Institute of Science, 1990.

[28] U. Feige, D. Lapidot and A. Shamir. Multiple Noninteractive Zero Knowledge Proofs under General Assumptions. *Siam Jour. on Computing 1999*, Vol. 29(1), pages 1–28.

[29] U. Feige and A. Shamir. Witness Indistinguishability and Witness Hiding Protocols. In *22nd STOC*, pages 416–426, 1990.

[30] U. Feige and A. Shamir. Zero Knowledge Proofs of Knowledge in Two Rounds. In *Crypto89*, Springer LNCS 435, pages. 526–544, 1989.

[31] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto86*, Springer LNCS 263, pages 181–187, 1987

[32] O. Goldreich. *Foundations of Cryptography – Basic Tools.* Cambridge University Press, 2001.

[33] O. Goldreich. Zero-knowledge twenty years after their invention. Weizmann Institute, 2002.

[34] O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Jour. of Cryptology*, Vol. 9, No. 2, pages 167–189, 1996.

[35] O. Goldreich and H. Krawczyk. On the Composition of Zero-Knowledge Proof Systems. *SIAM Jour. on Computing*, Vol. 25(1), pages 169–192, 1996.

[36] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *JACM*, Vol. 38(1), pp. 691–729, 1991.

[37] O. Goldreich and Y. Oren. Definitions and Properties of Zero-Knowledge Proof Systems. *Jour. of Cryptology*, Vol. 7, No. 1, pages 1–32, 1994.

[38] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Jour. on Computing*, Vol. 18(1), pp. 186–208, 1989.

[39] E. Goh and S. Jarecki. A Signature Scheme as Secure as the Diffie-Hellman Problem. In *EuroCrypt2003*, Springer LNCS 2656, pages 401–415, 2003.

[40] S. Goldwasser and Y. Tauman. On the (In)security of the Fiat-Shamir Paradigm In *44th FOCS*, pages 102–112, 2003.

[41] L.C. Guillou and J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing Both Transmission and Memory. In *EuroCrypt88*, Springer LNCS 330, pages 123–128, 1988.

[42] S. Hada and T. Tanaka. On the Existence of 3-Round Zero-Knowledge Protocols. In *Crypto98*, Springer LNCS 1462, pages 408–423, 1998.

[43] J. Håstad, R. Impagliazzo, L.A. Levin and M. Luby. Construction of Pseudorandom Generator from any One-Way Function. *SIAM Jour. on Computing*, Vol. 28 (4), pages 1364–1396, 1999.

[44] M. Jakobsson, K. Sako and R. Impagliazzo. Designated Verifier Proofs and Their Applications. In *EuroCrypt96*, Springer LNCS 1070, pages 143–154.

[45] N. Koblitz and A. Menezes. Another Look at "Provable Security". ePrint Archive, 2004/152, 2004.

[46] J. Kilian and E. Petrank. Concurrent and Resettable Zero-Knowledge in Polylogarithmic Rounds. In *33rd STOC*, pages 560–569, 2001.

[47] J. Kilian, E. Petrank and C. Rackoff. Lower Bounds for Zero-Knowledge on the Internet. In *39th FOCS*, pages 484–492, 1998.

[48] Y. Lindell. General Composition and Universal Composability in Secure Multi-Party Computation. In *44th FOCS*, pages 394–403, 2003

[49] Y. Lindell. Lower Bounds for Concurrent Self Composition. In *1st TCC*, Springer LNCS 2951, pages 203–222, 2004.

[50] M. Naor. Bit Commitment using Pseudorandomness. *Jour. of Cryptology*, Vol. 4, pages 151–158, 1991.

[51] J. Nielsen. Separating Random Oracle Proofs from Complexity Theoretic Proofs: The Non-Committing Encryption Case. In *Crypto2002*, Springer LNCS 2442, pages 111–126, 2002.

[52] M. Naor, R. Ostrovsky, R. Venkatesan and M. Yung. Perfect Zero-Knowledge Arguments for NP Using any One-Way Permutation. *Jour. of Cryptology*, Vol. 11, pages 87–108, 1998.

[53] M. Naor and M. Yung. Universal One-Way Hash Functions and their Cryptographic Applications. In *21st STOC*, pages 33–43, 1989.

[54] R. Pass. Simulation in Quasi-polynomial Time and its Application to Protocol Composition. In *EuroCrypt2003*, Springer LNCS 2656, pages 160–176, 2003.

[55] R. Pass. On Deniability in the Common Reference String and Random Oracle Models. In *Crypto 2003*, Springer LNCS 2729, pages 216–337, 2003.

[56] R. Pass and A. Rosen. Bounded-Concurrent Two-Party Computation in Constant Number of Rounds. In *44th FOCS*, pages 404–413, 2003

[57] M. Prabhakaran, A. Rosen and A. Sahai. Concurrent Zero-Knowledge with Logarithmic Round Complexity. In *43rd FOCS*, pages 366–375, 2002.

[58] M. Prabhakaran and A. Sahai. New Notions of Security: Achieving Universal Composability without Trusted Setup In *36th STOC*, pages 242–251, 2004.

[59] R. Richardson and J. Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. In *EuroCrypt99*, Springer LNCS 1592, pages 415–431, 1999.

[60] A. Rosen. A note on the round-complexity of Concurrent Zero-Knowledge. In *Crypto2000*, Springer LNCS 1880, pages 451–468, 2000.

[61] C. Rackoff and D. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Crypto91*, Springer LNCS 576, pages 433–444, 1991.

[62] A. Sahai. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. In *40th FOCS*, pages 543–553, 1999.

[63] C.P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *Crypto89*, Springer LNCS 435, pages 235–251, 1989.

[64] J. Stern and D. Pointcheval. Security Arguments for Digital Signatures and Blind Signatures. *Jour. of Cryptology*, Vol. 13, No. 3, pages 361–396, 2000.

[65] C. Dwork and L. J. Stockmeyer. 2-round zero knowledge and proof auditors. In *34th STOC*, pages 322–331, 2002.

[66] C. Dwork and M. Naor. Zaps and Their Applications. In *40th FOCS*, pages 283–293, 2000.

[67] C. Dwork and M. Naor, Pricing via Processing or Combatting Junk Mail. In *Crypto92*, Springer LNCS 740, pages 139–147, 1992.

[68] O. Goldreich, L. A. Levin. A Hard-Core Predicate for all One-Way Functions. In *21st STOC*, pages 25–32, 1989.

[69] S. Goldwasser, S. Micali. Probabilistic Encryption. JCSS 28(2), pages 270–299, 1984.

Appendix A

An Alternative Definition of Witness Indistinguishability

We note that the definition of $\mathcal{WI}$ in models with shared objects can be slightly weakened in the context of *interactive* proofs. Recall that Definition 25 requires that the outputs of the prover using two different witnesses are indistinguishable, by a distinguisher *that has access that the shared object.* We show that it is actually sufficient to only consider distinguishers that do not have access to the shared object.

Consider the following alternative definition, where the distinguisher does not have access to the shared object:

Definition 46 ($\mathcal{WI}$ with Shared Objects - Alternative definition) *Let (P, V) be an interactive proof in a model with shared objects for the language $L \in \mathcal{NP}$, and R_L be a fixed witness relation for L. We say that (P, V) is witness indistinguishable for R_L if for every PPT machine V^* and every two sequences $W^1 = \{w_x^1\}_{x \in L}$ and $W^2 = \{w_x^2\}_{x \in L}$, such that $w_x^1, w_x^2 \in R_L(x)$, the following two ensembles are computationally indistinguishable (when the distinguishing gap is a function in $|x|$):*

- $\{\langle P^R(w_x^1), V^{*R}(z)\rangle(x)\}_{x \in L, z \in \{0,1\}^*}$
- $\{\langle P^R(w_x^2), V^{*R}(z)\rangle(x)\}_{x \in L, z \in \{0,1\}^*}$

where R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$. That is, for every probabilistic algorithm D running in time polynomial in the length of its first input, every polynomial p, all sufficiently long $x \in L$, and all auxiliary inputs $z, z' \in \{0,1\}^*$ it holds that

$$
\begin{aligned}
&|Pr[D(x, z', \langle P^R(w_x^1), V^{*R}(z)\rangle(x))) = 1] \\
&-Pr[D(x, z', \langle P^R(w_x^2), V^{*R}(z)\rangle(x))) = 1]| < \frac{1}{p(|x|)}
\end{aligned}
$$

where R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$

Lemma 12 *If Π is a $\mathcal{WI}$ proof (or argument) in a model with shared objects, according to the alternative definition, then Π is $\mathcal{WI}$ in a model with shared objects, according to the standard definition.*

Proof We show that the distinguisher does not need to have access to the shared object, as the malicious verifier (which obviously has access to the shared object) can perform the distinguisher's operations. More formally, suppose, for contradiction, that Π is $\mathcal{WI}$ according to the alternative definition, but not according the standard definition, i.e there exists a PPT machine V^*, a PPT distinguisher D, and a positive polynomial p, such that for infinitely many n, there exists an $x \in L \cap \{0,1\}^n$, two witnesses $w^1, w^2 \in R_L(x)$ and auxiliary inputs $z, z' \in \{0,1\}^*$ such that

$$\begin{aligned} &|Pr[D^R(x, z', \langle P^R(w_x^1), V^{*R}(z)\rangle(x))) = 1] \\ -&Pr[D^R(x, z', \langle P^R(w_x^2), V^{*R}(z)\rangle(x))) = 1]| > p(|x|) \end{aligned}$$

where R is a random variable uniformly distributed in either $1^{poly(n)} \rightarrow \{0,1\}$ or $\{0,1\}^{poly(n)} \rightarrow \{0,1\}^{poly(n)}$. We construct a new distinguisher D' which does not have access to the shared object, and a new verifier V'^*.

- D' is defined as follows: $D'(x^*, z^*, out) = out$,
- V'^* proceeds as follows. V'^* first acts as V^* and thereafter internally executes $D^R(x, z', out')$, where out' is $V^*(z)$'s output. In order for V^* to be able to execute $D^R(x, z', out')$ and $V^*(z)$, V'^* receives $z^* = (z, z')$ as auxiliary input.

It follows that

$$\begin{aligned} &|Pr[D'(x, \cdot, \langle P^R(w_x^1), V'^{*R}(z^*)\rangle(x))) = 1] \\ -&Pr[D'(x, \cdot, \langle P^R(w_x^2), V'^{*R}(z^*)\rangle(x))) = 1]| > p(|x|) \end{aligned}$$

which contradict that Π is $\mathcal{WI}$ according to the alternative definition. ■