Resettably-Sound Zero-Knowledge and its Applications

Boaz Barak Oded Goldreich[*] Shafi Goldwasser[†] Yehuda Lindell
Department of Computer Science
Weizmann Institute of Science, ISRAEL
{boaz,oded,shafi,lindell}@wisdom.weizmann.ac.il

Abstract

Resettably-sound proofs and arguments maintain soundness even when the prover can reset the verifier to use the same random coins in repeated executions of the protocol. We show that resettably-sound zero-knowledge arguments for $\mathcal{NP}$ exist if collision-free hash functions exist. In contrast, resettably-sound zero-knowledge proofs are possible only for languages in $\mathcal{P}/\mathrm{poly}$.

We present two applications of resettably-sound zero-knowledge arguments. First, we construct resettable zero-knowledge arguments of knowledge for $\mathcal{NP}$, using a natural relaxation of the definition of arguments (and proofs) of knowledge. We note that, under the standard definition of proof of knowledge, it is impossible to obtain resettable zero-knowledge arguments of knowledge for languages outside $\mathcal{BPP}$. Second, we construct a constant-round resettable zero-knowledge argument for $\mathcal{NP}$ in the public-key model, under the assumption that collision-free hash functions exist. This improves upon the sub-exponential hardness assumption required by previous constructions.

We emphasize that our results use non-black-box zero-knowledge simulations. Indeed, we show that some of the results are impossible *to achieve using black-box simulations. In particular, only languages in $\mathcal{BPP}$ have resettably-sound arguments that are zero-knowledge with respect to black-box simulation.*

1 Introduction

Having gained a reasonable understanding of the security of cryptographic schemes and protocols as stand-alone, cryptographic research is moving towards the study of stronger notions of security. Examples include the effect of executing several instances of the same protocol concurrently (e.g., the malleability of an individual protocol [10]) as well as the effect of executing the protocol concurrently to any other activity (or set of protocols) [6]. Another example of a stronger notion of security, which is of theoretical and practical interest, is the security of protocols under a "resetting" attack. In a resetting attack, a party (being attacked) can be forced to execute a protocol several times while using the same random tape and without the ability to coordinate between these executions (as he may not even be aware of all the executions taking place). The theoretical interest in this notion stems from the fact that randomness plays a pivotal role in cryptographic protocols, and thus the question of whether one needs fresh (and independent) randomness in each invocation of a cryptographic protocol is natural. The practical importance is due to the fact that in many settings it is impossible or too costly to generate fresh randomness on the fly. Moreover, when parties in a cryptographic procotol are implemented by devices which cannot reliably keep state (e.g., smart cards), being maliciously "reset" to a prior state could be a real threat.

1.1 Resettable Provers

Resettability of players in a cryptographic protocol was first addressed by Canetti *et al.* in [7] who considered what happens to the security of zero-knowledge interactive proofs and arguments when the verifier can reset the prover to use the same random tape in multiple executions. Protocols which remain zero-knowledge against such a verifier, are called resettable zero-knowledge (rZK) protocols. Put differently, the question of prover resettability, is whether zero-knowledge is achievable when the prover cannot use fresh randomness in new interactions, but rather is restricted to (re-)using a fixed number of coins.

Resettability implies security under concurrent executions: any rZK protocol constitutes a concurrent zero-knowledge protocol. The opposite direction does not hold, and indeed it was not a-priori clear whether (non-trivial) rZK protocols exist. The main result of Canetti *et al.* answers this question affirmatively, under some standard complexity assumptions. Specifically, assuming the existence of

[*] Supported by the MINERVA Foundation.
[†] Supported in part by NTT grant

perfectly hiding and computationally binding commitment schemes, resettable zero-knowledge interactive proofs for $\mathcal{NP}$ using polynomially many rounds do exist [7].[1]

In order to obtain a constant-round rZK protocol, Canetti *et al.* introduced a weak public-key model and used a strong intractability assumption - the existence of a perfectly hiding and computationally binding commitment scheme that cannot be broken by sub-exponential size circuits, and not merely by polynomial-size ones. In this model and under that assumption, they were able to construct a constant-round rZK argument system for $\mathcal{NP}$ [7].

On the negative side, [7] point out that resettable zero-knowledge *proofs of knowledge* are impossible to achieve for non-trivial languages, ruling out the use of the Fiat-Shamir [15] paradigm of identification protocols based on proofs of knowledge when the provers may be resettable. This impossibility extends to resettable zero-knowledge arguments of knowledge and to resettable witness indistinguishable proofs and arguments of knowledge. All these negative results are with respect to the standard definition of proofs of knowledge (cf. [4]), in which the extractor of knowledge is limited to oracle access to the prover (for detailed discussion see Section 1.3.2).

1.2 Resettable Verifiers

In a similar fashion, one may consider what happens to the soundness of (zero-knowledge) interactive proofs and arguments when the prover can reset the verifier to use the same random tape in multiple concurrent executions.

Informally, we say that an interactive proof or argument achieves *resettable soundness* if a prover cannot convince a verifier of an incorrect statement with non-negligible probability, even when the prover can reset the verifier to use the same random tape in multiple concurrent executions. The verifier resettability question can be recast as whether soundness can be achieved, when the verifier is restricted to (re-)using a fixed number of coins rather than using fresh coins in every interaction.

Resettable-soundness *in the public key model* was already defined and studied by Micali and Reyzin [25]. They showed that the existing rZK protocols in the public-key model (i.e., [7, 24]) are not resettably-sound (i.e., do not maintain soundness when the verifier can be reset). Furthermore, they demonstrated the non-robustness of soundness *in the public key model* by considering several natural notions of soundness (i.e., one-time soundness, sequential soundness, concurrent soundness, and resettable soundness), and showing separations between these notions.

[1]The number of rounds was recently improved to poly-logarithmic [23]. Interestingly, poly-logarithmic many rounds are necessary for any protocol that can be shown to be concurrent zero-knowledge via a black-box simulator [8].

1.3 Our contributions

In this paper we study resettable-soundness in the standard model, rather than in the public-key model considered in [7, 25]. As was the case for resettable zero-knowledge, it is not clear a-priori whether non-trivial resettably-sound zero-knowledge protocols exist at all.

Indeed, the situation here is worse: we show that resettably-sound zero-knowledge *proofs* exist only for $\mathcal{P}/\mathrm{poly}$.[2] Furthermore, if one is restricted to showing zero-knowledge via a black-box simulator, resettably-sound zero-knowledge *arguments* exist only for BPP-languages. Thus, our study would have come to an end, if it were not for the recent result of Barak [1] in which for the first time, zero-knowledge arguments are constructed using non-black-box simulators. This opens the door to hope to get around impossibility results regarding zero-knowledge proved via black-box simulators.

Indeed, we construct resettably-sound zero-knowledge arguments for $\mathcal{NP}$, using Barak's construction. In turn, using resettably-sound zero-knowledge arguments for $\mathcal{NP}$, we obtain two main applications:

1. Resettable zero-knowledge arguments of knowledge for $\mathcal{NP}$, using a relaxed and yet natural definition of arguments (and proofs) of knowledge (see Section 1.3.2).

2. Constant-round resettable zero-knowledge arguments for $\mathcal{NP}$, in the public-key model, under weaker assumptions than known previously: instead of assuming sub-exponential hardness, we only assume super-polynomial hardness.

All our positive results inherit Barak's [1] intractability assumption – the existence of collision-free hash functions [2]. As the existence of collision-free hash functions implies the existence of one-way functions, we use the latter freely. Our protocols also inherit from [1] non-black-box demonstrations of various properties.

We proceed to give details on our main result and its applications.

1.3.1 Main result

Our main result is a constant-round resettably-sound zero-knowledge argument for $\mathcal{NP}$, assuming the existence of collision-free hash functions. This is achieved by showing how to transform any constant-round public-coin zero-knowledge interactive argument into a constant-round resettably-sound zero-knowledge argument for the same

[2]We also show that resettably sound proofs – without the zero-knowledge requirement – are possible only for languages in $\mathcal{NP}/\mathrm{poly}$.

language, and then applying the transformation to the recent construction [1] of a constant-round public-coin zero-knowledge argument of knowledge for $\mathcal{NP}$.

Recall, that until recently, this transformation would have been useless as no constant-round public-coin zero-knowledge arguments were known for languages outside of $\mathcal{BPP}$. Indeed, Goldreich and Krawczyk proved that only languages in $\mathcal{BPP}$ have constant-round public-coin arguments and proofs that are black-box zero-knowledge [19]. Naturally, the [1] construction of a constant-round public-coin zero-knowledge argument of knowledge for $\mathcal{NP}$ must (and does) use a non-black-box simulator. Thus, we obtain:

Theorem 1.1 *If there exist collision-free hash functions, then any NP-language has a* (constant-round) *resettably-sound zero-knowledge argument. Furthermore, these protocols are arguments of knowledge.*

Using Theorem 1.1 we obtain the following applications.

1.3.2 Application 1: Resettable-ZK Arguments of Knowledge

The standard definition of an argument (or proof) of knowledge requires the knowledge-extractor to use the prover's strategy as a black-box. Furthermore, in a resetting attack on the prover, the verifier has this very same capability during the execution of the protocol . Loosely speaking, then, if the extractor can extract anything (e.g., an NP-witness) from the prover, then so can a cheating verifier mounting a resetting attack. Thus, under the standard definition (cf. [4]), resettable zero-knowledge arguments of knowledge exist only for $\mathcal{BPP}$.

We adopt a relaxation of the definition of an argument (or proof) of knowledge in which the knowledge-extractor is given the prover's program as auxiliary input (rather than given only black-box access to it). The knowledge-extractor, now, is (at least syntactically) more powerful than the cheating verifier during a resetting attack in which the latter has in essence black-box access to the prover's strategy. The relaxed definition appeared originally in Feige and Shamir [14] (which differs from the definition in [12]; see discussion in [4]), and suffices for all practical applications of arguments of knowledge.

The standard definition, allowing only oracle access to the prover's strategy, was used in the literature in the past as in principle it allows the consideration of prover strategies which are not in $\mathcal{P}/\mathrm{poly}$ (an irrelevant consideration for practical applications and for arguments in particular). Moreover, it was generally believed, that one cannot benefit from non-black-box access to the prover's code, and thus restricting access to the prover poses no limitation.

Henceforth we will use "proof of knowledge" to refer to the relaxed definition.

Using Theorem 1.1, we construct resettable zero-knowledge and resettable witness-indistinguishable arguments of knowledge for $\mathcal{NP}$. Our construction is based on a modification of a well-known design principle underlying protocols such as those in [18, 27, 7]: In these protocols, the verifier starts by committing to its queries, then the prover sends some information, and the verifier decommits to the abovementioned queries, which the prover is now supposed to answer. Such protocols usually fail to yield proofs of knowledge, as the typical way in which a knowledge-extractor works is by obtaining answers to several different queries regarding the same piece of information, but this way is blocked when the queries are committed to before the information is presented. Our modification is to replace the action of decommitment by merely revealing the committed values and proving in zero-knowledge that the revealed values are indeed those committed to. Toward this end, the verifier needs to employ a zero-knowledge proof (or argument), in which the prover plays the role of the verifier, which is why in our setting (in which the main prover is resettable) this subprotocol has to be resettably-sound. Here is where we use Theorem 1.1, which supplies us with a resettably-sound zero-knowledge argument for $\mathcal{NP}$. Thus, we obtain:

Theorem 1.2 *If there exist collision-free hash functions, then there exists*

1. *A constant-round resettable witness-indistinguishable argument of knowledge for $\mathcal{NP}$.*
2. *A poly-logarithmic round resettable zero-knowledge argument of knowledge for $\mathcal{NP}$.*

All applications of the notion of a proof of knowledge, including the Fiat-Shamir paradigm of building identification protocols from zero-knowledge (and witness indistinguishable) proofs of knowledge [15], are thus salvaged for resettable provers. This holds also with respect to constant-round protocols in the public-key model; see Theorem 1.3 (below). Of course, for any particular proof of knowledge, one needs to explicitly prove being **resettable** zero-knowledge (or witness indistinguishable).

1.3.3 Application 2: rZK in the Public-Key Model under weaker assumptions

Current protocols that achieve constant-round resettable zero-knowledge arguments for $\mathcal{NP}$, *in the public-key model*, assume a sub-exponential lower bound on the size of circuits attempting to break commitment schemes.

In contrast, using Theorem 1.1 we construct constant-round resettable zero-knowledge arguments for $\mathcal{NP}$ in the public-key model, relying only on the existence of collision-free hash functions. Thus, we replace a sub-exponential hardness assumption by a standard hardness assumption of collision-free hashing secure against all polynomial time adversaries. Furthermore, both the protocol and its analysis are conceptually simpler than the corresponding constructions presented in [7, 24].

Moreover, the constant-round protocol constructed is also an argument of knowledge (in the relaxed sense discussed in previous section).

Theorem 1.3 *If there exist collison-free hash functions, then there exists a constant-round resettable zero-knowledge argument of knowledge for $\mathcal{NP}$* in the public-key model.

1.4 Simultaneous resettability

A natural question that arises is whether it is possible to simultaneously protect both the prover and the verifier from resetting attacks. That is:

Open Problem 1.4 *Do languages outside of $\mathcal{BPP}$ have resettably-sound arguments that are resettable zero-knowledge.*

Some hope for an affirmative resolution of the above question is provided by the fact that some level of resettable-security for both parties does seem to be achievable.[3] That is:

Theorem 1.5 (implicit in [11]): *Assuming the existence of trapdoor permutations, any NP-language has a resettably-sound proof that is resettable witness-indistinguishable.*

Theorem 1.5 follows from the following facts regarding ZAPs (as defined by Dwork and Naor [11]). Loosely speaking, ZAPs are two-round public-coin witness-indistinguishable proofs. Thus, by definition, ZAPs are resettably-sound (because even in a single session the prover obtains all the verifier's coins before sending its own message). On the other hand, as noted in [11], any ZAP can be made resettable witness-indistinguishable (by using pseudorandom functions as in the transformation of [7]). Using a main result of [11], by which ZAPs for $\mathcal{NP}$ can be constructed based on any non-interactive zero-knowledge proofs for $\mathcal{NP}$, which in turn can be constructed based on trapdoor permutations (cf. [13, 5]), Theorem 1.5 follows.

We conjecture that resettably-sound resettable *zero-knowledge* arguments for $\mathcal{NP}$ do exist, and have made some progress towards establishing this.

[3]The evidence provided by Theorem 1.5 (towards an affirmative resolution of the above question) is admittingly not very strong. In general, zero-knowledge seems a significantly stronger notion of security than witness-indistinguishability. Furthermore, Theorem 1.5 yields resettably-sound proofs, whereas (as mentioned above) there is no hope of obtaining resettably-sound zero-knowledge proofs (rather than arguments) for languages outside $\mathcal{P}/\mathrm{poly}$.

1.5 Organization

In Section 2 we present the definition of resettable-soundness and show the existence of resettably-sound zero-knowledge arguments of knowledge. We further discuss the limitations of resettably-sound proofs and the triviality of resettably-sound zero-knowledge with black-box simulation. Due to space limitations, we present only an overview of the applications of resettably-sound zero-knowledge arguments. Full descriptions of the protocols and their proofs are presented in the full version of this work [3].

2 Resettable-Soundness

In this section we define and study various notions of resettable-soundness. Specifically, we define resettably-sound proofs and arguments, and justify our focus on the latter (where soundness holds only with respect to polynomial-size cheating provers, rather than for arbitrary cheating provers).

2.1 Definitions

We adopt the formalism of resettable zero-knowledge (cf. [7]), with the understanding that here the adversary plays the role of the prover and has the power to reset the verifier (or invoke it several times on the same sequence of coins).[4]

Given a specified verifier program V and a common input x, we consider polynomially-many interactions with the residual deterministic verifier strategy $V_{x,r}$ determined by uniformly selecting and fixing V's coins, denoted r. That is, r is uniformly selected and fixed once and for all, and the adversary may invoke and interact with $V_{x,r}$ many times. Each such interaction is called a session. Thus, the adversary and $V_{x,r}$ engage in polynomially-many sessions; but whereas $V_{x,r}$'s actions in the current session are oblivious of other sessions (since $V_{x,r}$ mimics the "single session strategy" V), the actions of the adversary may depend on other sessions. Typically, $x \notin L$ and the aim of the adversary, or cheating prover, is to convince $V_{x,r}$ to accept x in one of these sessions. (In the context of resettable zero-knowledge, the adversary is called a cheating verifier and its aim is to "extract knowledge" from the prover by possibly resetting it.)

We consider two variants of the model. In the first (and main) variant, a session must be terminated (either completed or aborted) before a new session can be initiated by the adversary. In the second (interleaving) variant, this restriction is not made and so the adversary may concurrently

[4]In contrast, in the context of resettable zero-knowledge, the adversary plays the role of the verifier and has the power to reset the prover.

initiate and interact with $V_{x,r}$ in many sessions. A suitable formalism must be introduced in order to support these concurrent executions. (For simplicity, say that the adversary prepends a session-ID to each message it sends, and a distinct copy of $V_{x,r}$ handles all messages prepended by each fixed ID.) Note that in both variants, the adversary may repeat in the current session the same messages sent in a prior session, resulting in an identical prefix of an interaction (since the verifier's randomness is fixed). Furthermore, by deviating in the next message, the adversary may obtain two different continuations of the same prefix of an interaction. Viewed in other terms, the adversary may "effectively rewind" (or "reset") the verifier to any point in a prior interaction, and carry-on a new continuation (of this interaction prefix) from this point.

For sake of simplicity, we will present only the definition of the main (non-interleaving) model. We can afford to focus on the non-interleaving model because the argument given in [7] by which the models are equivalent with respect to resettable zero-knowledge hold also with respect to resettable-soundness; the reason being that this argument merely shows how a resetting-adversary in the interleaving model can be perfectly emulated by a resetting-adversary in the non-interleaving model.

Following Canetti *et al.* [7], we extend the basic model to allow the adversary to interact (many times) with several random independent incarnations of V (rather than with a single one). That is, rather than interacting many times with one $V_{x,r}$, where r is randomly selected and x is predetermined, the adversary may interact many times with different V_{x_i,r_j}'s, where the r_j's are independently and randomly selected and the x_i's are chosen dynamically by the adversary. One may be tempted to say that the ability to interact with several incarnations of V should not add power to the model, but as shown in [7] this intuition is not valid.

One important deviation from the formalism of Canetti *et al.* [7], is in not fixing a sequence of (polynomially-many) x_i's ahead of time, but rather allowing the adversary to select such x_i's on the fly. Furthermore, adversarial selection of inputs is used in both the completeness and soundness conditions. (We comment that the latter strengthening of the definition is applicable and desirable also in the setting of resettable zero-knowledge.)

Definition 2.1 (resettable verifier – main model): *A* resetting attack *of a cheating prover P^* on a* resettable verifier *V is defined by the following two-step random process, indexed by a security parameter n.*

1. *Uniformly select and fix $t = \mathrm{poly}(n)$ random-tapes, denoted $r_1, ..., r_t$, for V, resulting in deterministic strategies $V^{(j)}(x) = V_{x,r_j}$ defined by $V_{x,r_j}(\alpha) = V(x, r_j, \alpha)$, where $x \in \{0,1\}^n$ and $j \in [t]$.*[5] *Each $V^{(j)}(x)$ is called an* incarnation of *V.*
2. *On input 1^n, machine P^* is allowed to initiate $\mathrm{poly}(n)$-many interactions with the $V^{(j)}(x)$'s. The activity of P^* proceeds in rounds. In each round P^* chooses $x \in \{0,1\}^n$ and $j \in [t]$, thus defining $V^{(j)}(x)$, and conducts a complete session with it.*

Let P and V be some pair of interactive machines, and suppose that V is implementable in probabilistic polynomial-time. We say that (P, V) is a resettably-sound proof system for *L* (resp., resettably-sound argument system for *L) if the following two conditions hold:*

- Resettable-completeness: *Consider an arbitrary resetting attack* (resp., polynomial-size resetting attack), *and suppose that in some session, after selecting an incarnation $V^{(j)}(x)$, the attacker follows the strategy P.*[6] *Then, if $x \in L$ then $V^{(j)}(x)$ rejects with negligible probability.*
- Resettable-soundness: *For every resetting attack* (resp., polynomial-size resetting attack)*, the probability that in some session the corresponding $V^{(j)}(x)$ has accepted and $x \notin L$ is negligible.*

We stress that by a resettably-sound *proof* we mean that the resettable-soundness requirement holds also for computationally unbounded cheating provers, whereas only polynomial-size cheating provers are considered in the definition of resettably-sound *arguments*.

We also adapt the definition of a proof of knowledge to the resettable context. We assume that the reader is familiar with the basic definition of a proof of knowledge (cf., [4]). The basic approach is to link the probability that any prover strategy convinces the verifier to the efficiency of extracting the claimed knowledge by using this prover strategy as an oracle. Thus, a definition of a resettably-sound proof (or argument) of knowledge should refer to the probability of convincing the verifier during a resetting attack (rather than to the probability of convincing the verifier in an ordinary interaction). On the other hand, we relax the definition by allowing the extractor to depend on the size of the prover strategy (i.e., we focus on polynomial-size provers and allow a different extractor per each polynomial size-bound).[7]

[5]Recall that $V(x, r, \alpha)$ denotes the message sent by the strategy V on common input x, random-tape r, after seeing the message-sequence α.

[6]In fact, in order to consider honest prover strategies that are implementable in probabilistic polynomial-time, we need to supply P with an adequate NP-witness. That is, let R be an NP-relation that corresponds to the NP-language L. Then we consider a resetting attack that for every selected $x \in L$ also provides P with (an NP-witness) w satisfying $(x, w) \in R$. In this case, we require that when $V^{(j)}(x)$ interacts with $P(w)$ it rejects with negligible probability.

[7]In fact, it suffices to allow a different extractor per each polynomial bound on the number of sessions initiated by the prover. (Such a relaxation coincides with the standard definition for the case of a single session.) However, since we focus on polynomial-size provers, we may as well refer to their size.

The latter relaxation seems important for our positive results and is certainly sufficient for our applications (as well as for any other application we can think of). For simplicity, we consider below only NP-relations.

Definition 2.2 (resettably-sound argument of knowledge, sketch): *Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be an NP-relation for an NP-language $L = \{x : \exists w\, (x,w) \in R\}$. We say that (P,V) is a* resettably-sound argument of knowledge for *R if*

- *(P,V) is a resettably-sound argument for L, and*
- *for every polynomial q there exists a probabilistic expected polynomial-time oracle machine E such that for every resetting attack P^* of size $q(n)$, the probability that $E^{P^*}(1^n)$ outputs a witness for the input selected in the last session is at most negligibly smaller than the probability that P^* convinces V in the last session.*

The focus on the last session of the resetting attack is done for simplicity and is valid without loss of generality (because the prover can always duplicate the interaction of any session of its choice in the last session).

Important Note: We stress that, in the rest of this section, zero-knowledge mean the standard notion (rather than *resettable zero-knowledge*).

2.2 Limitations of resettably-sound proofs

In this subsection we justify our focus on resettably-sound *arguments* (rather than resettably-sound *proofs*): As demonstrated below, resettably-sound proofs exist only in an almost trivial manner, and more annoyingly resettably-sound *zero-knowledge* proofs exist only for languages having (non-uniform) polynomial-size circuits (and thus are unlikely to exist for all $\mathcal{NP}$).

Theorem 2.3 *Suppose that there exists a resettably-sound proof for L. Then, L is contained in non-uniform $\mathcal{NP}$* (i.e., $L \in \mathcal{NP}/\mathrm{poly}$). *Furthermore, if this proof system is zero-knowledge then L is contained in non-uniform polynomial-time* (i.e., $L \in \mathcal{P}/\mathrm{poly}$).

Note that $\mathcal{AM} \subset \mathcal{NP}/\mathrm{poly}$ does have resettably-sound proof systems (e.g., the first message sent in a properly amplified AM-proof system can be used to correctly prove membership of all strings of adequate length).[8] Similarly, $\mathcal{BPP} \subset \mathcal{P}/\mathrm{poly}$ does have resettably-sound zero-knowledge proof systems (in which the verifier just decides by inspecting the input). We believe that Theorem 2.3 holds also with $\mathcal{NP}/\mathrm{poly}$ replaced by $\mathcal{AM}$ and $\mathcal{P}/\mathrm{poly}$ replaced by $\mathcal{BPP}$.

[8]Recall that $\mathcal{AM}$ stands for the class of languages having two-round public-coin interactive proofs.

Proof Idea: Intuitively, the verifier's randomness is a-priori bounded, whereas the number of sessions in which it takes place (in a resettable attack) is not a priori bounded. Thus, there must be a session in which the verifier uses very little truly new randomness (i.e., there is a session in which the verifier's moves are almost determined by the history of previous sessions). Loosely speaking, the limitations of deterministic verifiers (with respect to interactive proofs and zero-knowledge proofs; cf., [16] and [21], respectively) should apply here.

2.3 On the triviality of resettably-sound black-box zero-knowledge

In this section we explain why the resettably-sound zero-knowledge arguments presented in the next subsection are not accompanied (as usual) by a black-box simulator. Specifically, we show that only a language in $\mathcal{BPP}$ can have a resettably-sound argument with a black-box zero-knowledge simulator (and, in fact, $\mathcal{BPP}$ languages have trivial "proof" systems in which the prover does not even take part). Independently, Reyzin in [26] showed that Theorem 2.4 holds even in the public-key model.

Theorem 2.4 *Suppose that there exists a resettably-sound argument for L, and that this protocol is black-box zero-knowledge. Then, $L \in \mathcal{BPP}$.*

Proof Idea: Intuitively, a (probabilistic polynomial-time) cheating prover mounting a resettable attack on the verifier, may emulate the actions of the black-box simulator (with access to the deterministic verifier defined by fixing the random-tape). Thus, in case $x \in L$, this cheating prover causes the verifier to accept x (because the emulated simulator succeeds in producing an accepting conversation). On the other hand, by the resettable-soundness condition, the cheating prover is unlikely to cause the verifier to accept $x \notin L$. Finally, observing that the cheating prover described above is implementable in probabilistic polynomial-time, we obtain a probabilistic polynomial-time decision procedure for L.

2.4 How to construct resettably-sound zero-knowledge arguments

The main result of this section is obtained by combining the following transformation with a recent result of Barak [1].

Proposition 2.5 (a transformation): *Let $L \in \mathcal{NP}$ and R be a corresponding witness relation. Suppose that (P,V) is* *a* constant-round public-coin *argument of knowledge for R, and let $\{f_s : \{0,1\}^* \to \{0,1\}^{|s|}\}$ be a collection of pseudorandom functions. Assume, without loss of generality, that on common input x, in each round, the verifier V sends a uniformly distributed $|x|$-bit string. Let W_s be a deterministic verifier program that, on common input $x \in \{0,1\}^{|s|}$, emulates V except that it determines the current round message by applying f_s to the transcript so far. Let W be defined so that on common input x and uniformly random-tape $s \in \{0,1\}^{|x|}$, it acts as $W_s(x)$. Then:*

1. *(P, W) is a resettably-sound argument for L. Futhermore, (P, W) is a resettably-sound argument of knowledge for R.*
2. *If (P, V) is zero-knowledge then so is (P, W). Furthermore, if the simulator of (P, V) runs in strict polynomial-time, then so does the simulator of (P, W).*
3. *If (P, V) is witness-indistinguishable then so is (P, W).*

Recall that only languages in $\mathcal{BPP}$ have a constant-round public-coin zero-knowledge argument with a black-box simulator. Thus, Proposition 2.5 may yields something interesting only when applied to protocols that do not have a black-box simulator.[9] Here is where the result of Barak [1] plays a role: Using his recent constant-round public-coin zero-knowledge argument of knowledge for $\mathcal{NP}$ (which indeed uses a non-black-box simulator), we obtain resettably-sound zero-knowledge arguments (of knowledge) for $\mathcal{NP}$. This establishes Theorem 1.1.

Proof Sketch: Parts 2 and 3 follows immediately because the zero-knowledge condition (as well as the witness-indistinguishability condition) does not refer to the honest verifier (which is the only thing we have modified) but rather to all possible polynomial-size adversaries (representing cheating verifiers).

As a preliminary step towards proving Part 1, we consider an imaginary verifier (denoted W_F) that, on common input x, uses a truly random function $F : \{0,1\}^* \to \{0,1\}^{|x|}$ (rather than a pseudorandom function f_s, for $|s| = |x|$). Loosely speaking, by the definition of pseudorandomness, all non-uniform polynomial-size provers must behave in essentially the same way under this replacement.

(The actual proof is slightly more subtle than one may realize because in our context this "behavior" is the ability to convince the verifier of the membership in L of $x \notin L$ that is chosen on the fly by the prover. The problem is how will the distinguisher determine that this event took place (notice that the distinguisher itself may not necessarily know whether or not $x \in L$). Thus, we actually proceed as follows. First, we show (below) that any feasible P' may convince the resettable W_F to accept an input not in L only with negligible probability. Next, we use the hypothesis that (P, V) is an argument of knowledge of an NP-witness, to extract such witnesses for every input accepted by W_F. Specifically, for any input accepted by W_F we may employ the knowledge-extractor to a related (non-resetting) P'' (defined below) and obtain an NP-witness. Thus, for the (P', W_F) transcript, we expect to couple each accepted input with a corresponding NP-witness. If this does not occur with respect to the (P', W_s) transcript (where s specifies a pseudorandom function), then we distinguish a random function from a pseudorandom one. Otherwise, we are done (because when a random function is used all accepted inputs will be coupled with NP-witness guaranteeing that they are indeed in L.)[10]

We claim that for any polynomial-size cheating prover P' that convinces the resettable-verifier W_F to accept some common input x with probability ϵ, there exists a polynomial-size cheating prover P'' that convinces the original (non-resettable) verifier V to accept the same x with probability at least ϵ/m^c, where m is a bound on the number of messages sent by the prover P' and c is the number of rounds in the original protocol. Furthermore, we shall show how to transform a cheating prover for the resettable-verifier setting into a cheating prover for the standard setting of interactive proofs. This will establish Part 1 (both its main claim and the furthermore-clause).

For sake of simplicity, we shall assume that the original cheating prover P' tries to convince the resettable-verifier W_F to accept a fixed x, and only invokes a single incarnation of W_F (and does so on common input x). The argument extends easily to the general case in which P' invokes multiple incarnations of W_F and selects the common inputs adaptively.

The new cheating prover, denoted P'', tries to convince the original (non-resettable) verifier V to accept x, while emulating the actions of a cheating prover P' that may reset the imaginary resettable verifier W_F. The new cheating prover P'' proceeds as follows: It uniformly selects $i_1, ..., i_c \in \{1, ..., m\}$, and invokes (the resetting prover) P' while emulating an imaginary verifier W_F as follows. If the

[9] In particular, we may apply Proposition 2.5 to some known constant-round public-coin interactive proofs that are known to be witness-indistinguishable (e.g., parallel repetitions of the basic zero-knowledge proof of Goldreich, Micali and Wigderson [20]). This yields witness-indistinguishable arguments that are resettably-sound. However, for witness-indistinguishable protocols, stronger results are known; see Section 1.4. Thus, we focus below on zero-knowledge protocols.

[10] The above text suffices for establishing the main part of Part 1. To establish the furthermore-part, we employ analogous reasoning with respect to the event that P' convinces W_F (or W_s) to accept an input without "knowing" a corresponding NP-witness, where "knowing" means that our extraction succeeds. As before, this event occurs with negligible probability when P' interacts with W_F and therefore the same must hold with respect to the interaction of P' with W_s.

prefix of the current session transcript is identical to a corresponding prefix of a previous session, then P'' answers by copying the same answer it has given in the previous session (to that very same session transcript prefix). If (in the current session) P' sends along a message that together with the previous messages of the current session forms a new transcript prefix (i.e., the prefix of the current session transcript is different from the prefixes of all prior sessions), then P'' answers according to the following two cases:

1. The index of the current message of P' does not equal any of the c integers $i_1, ..., i_c$ selected above. In this case, P'' provides P' with a uniformly selected $|x|$-bit long string.
2. Otherwise (i.e., the index of the current message of P' equals one of the c integers $i_1, ..., i_c$), P'' forwards the current message (of P') to V and feeds P' with the message it obtains from V. (We stress that these are the only c messages of P' for which the emulation involves interaction with V.)

In both cases, the message passed to P' is recorded for possible future use.

Clearly, for any possible choice of the integers $i_1, ..., i_c$, the distribution of messages seen by P' when P'' emulates an imaginary verifier is identical to the distribution that P' sees when actually interacting with such an imaginary verifier. The reason being that in both cases different prefixes of session transcripts are answered with uniformly and independently distributed strings, while session transcripts with identical prefixes are answered with the same string. (Observe that indeed this emulation is possible because the original verifier is of the public-coin type, and thus it is possible to efficiently emulate the next verifier message.)[11]

Towards the analysis, we call a message sent by P' novel if together with the previous messages of the current session it forms a new transcript prefix (i.e., the prefix of the current session transcript is different from the prefixes of all prior sessions). Recall that the novel messages are exactly those that cause P'' to pass along (to P') a new answer (rather than copying an answer given in some previous session). The UrMessage[12] of a non-novel message is the corresponding message that appears in the first session having a transcript-prefix that is identical to the current session transcript-prefix. That is, the answer to the UrMessage of a (non-novel) message is the one being retrieved from memory in order to answer the current message. The UrMessage of a novel message is just the message itself. Using this terminology, note that the new prover P'' succeeds in cheating V if the chosen integers $i_1, ..., i_c$ equal the indices (within the sequence of all messages sent by P') of the c UrMessages that correspond to the c messages sent in a session in which P' convinced the imaginary verifier. Since with probability ϵ such a convincing session exists, P'' succeeds provided it has guessed its message indices (i.e., c indices out of m). ■

[11] In contrast, if the verifier were not of the public-coin type, then generating a next verifier message might have required to invert an arbitrary polynomial-time computable function (mapping the verifier's coins to its first message.)

[12] We use the German prefix *Ur*, which typically means *the most ancient version of*.

3 Overview of the Applications

We present two applications for resettably-sound zero-knowledge arguments. The first (stated in Theorem 1.2) is obtaining resettable zero-knowledge (rZK) and constant-round resettable witness-indistinguishable (rWI) *arguments of knowledge for $\mathcal{NP}$*. In the second (stated in Theorem 1.3), we obtain a constant-round rZK argument (of knowledge) for $\mathcal{NP}$ in the *public-key model*, under weaker assumptions than previously known.

3.1 Proof of Theorem 1.2: rZK and rWI Arguments of Knowledge

The key step in showing the existence of rWI and rZK *arguments of knowledge*) is in constructing a rWI *argument of knowledge*. Given such a protocol, rZK arguments of knowledge can be obtained by combining the rWI argument of knowledge with a modification of the Richardson-Kilian protocol [27]. (An alternative construction combines a rWI argument of knowledge, a rZK proof of membership (that exists by [7]) and a perfectly-binding commitment scheme.)

The core of our rWI argument of knowledge is the 3-round proof of Hamiltonicity by Blum. As in [18], we augment Blum's protocol[13] by having the verifier initially commit to its query string (rather than choose it after the prover's commitment). Then, after receiving the prover's commitments, the verifier reveals its query string and proves to the prover that this query string is consistent with the initial commitment (sent by the verifier in the first step of the protocol). If convinced, the prover continues as in Blum's protocol.

In the protocol of [18], the verifier "proves" the consistency of the query string with the initial commitment by simply providing the decommitment information. (This prevents the standard knowledge-extraction techniques and is the reason that the protocol of [18] is not known to be a proof of knowledge.) In contrast (in order to allow knowledge-extraction), in our protocol the verifier proves the consistency of the query string by using a resettably-sound zero-knowledge argument. The argument used must

[13] The reason we use Blum's protocol rather than the 3-colorability protocol of [20] is that knowledge-extraction is easier with it.

be resettably-sound in order to protect the prover against a cheating verifier that is able to reset it (notice that the prover plays the role of the verifier in the argument of knowledge subprotocol). On the other hand, the fact that a zero-knowledge proof or argument is used, allows the knowledge-extractor to cheat (by running the simulator instead of providing a real proof) and so succeed in extracting the claimed knowledge. (Note that the knowledge-extractor is given the code of the prover, and thus can do things that even a cheating verifier mounting a resetting attack on the prover cannot do.)

Indeed, the novelty of our protocol is that it is an argument *of knowledge* (for Hamiltonicity), rather than merely an argument of membership (in Hamiltonicity). That is, we construct an extractor K who, given access to the *code* of a prover P^*, extracts a Hamiltonian cycle (with probability negligibly close to the probability that P^* convinces V). In general, the strategy for extraction from the basic proof of Hamiltonicity (of Blum) involves obtaining the answer to two *different* query strings with respect to the *same* set of prover commitments. The real verifier is unable to do this since it is bound to its queries by the initial commitment (otherwise, the protocol would clearly not be witness indistinguishable). However, K has an advantage over the verifier in that it has access to the code of P^*. Therefore, K can run the (non black-box) simulator for the resettably-sound zero-knowledge proof that asserts the validity of the decommitment. This enables K to cheat and "decommit" to different query strings, thus extracting a Hamiltonian cycle.

3.2 Proof of Theorem 1.3: rZK in the Public-Key Model

Canetti et al. [7] introduced a weak notion of a public-key setup, in order to obtain constant-round resettable zero-knowledge arguments (a somewhat stronger assumption was independently suggested by Damgård [9] in order to obtain concurrent zero-knowledge). The only requirement in this public-key model is that all verifiers deposit some public-key in a public file *before* the prover begins any interaction. We stress that there is no requirement whatsoever on the "validity" of the public keys deposited. The use of the public-file is simply to limit the number of different identities that a potential adversary may assume (note that the adversary may try to impersonate any registered user, but it cannot act on behalf of a non-registered user). For a more detailed description of the public-key model, see [7].

Similarly to the public-key protocol of [7], in the first stage of our protocol the verifier proves a zero-knowledge argument of knowledge of the secret-key associated to its public-key (in the public file). The nature of the public-key is such that simulation is made "easy" (without any necessity for rewinding) if the associated secret-key is known. Therefore, simulation proceeds by first extracting the secret-key from the argument of knowledge, and then completing the simulation (without any rewinding). As in [7], the role of the public file is to prevent the adversary from using "too many" public keys. We note, however, that (for reasons to be discussed below) the second stage of our protocol is completely different from the protocol of [7]. Despite this difference, both protocols share the property that the "hard part" of the simulation is extracting the secret-key and the "easy part" is simulating the second stage, given the secret key. Another important difference between our protocol and the protocol of [7] relates to the argument of knowledge used by the verifier in the first stage. We use a resettably-sound zero-knowledge argument of knowledge, whereas they rely on a (resettably-sound) proof system that can be simulated in subexponential time. This difference expresses itself in the proof of soundness, and is one of the reasons why we are able to base our construction on a weaker intractability assumption (while using a simpler proof of soundness). The other reason is that our protocol avoids issues related to the malleability of commitment schemes (which are resolved in [7] by using a subexponential intractability assumption).

We now briefly describe the protocol. The verifier's public-key consists of a commitment to a pseudorandom function. In the first stage of the protocol, the verifier provides a zero-knowledge argument of knowledge that it knows the decommitment. (Since the prover is resettable and plays the verifier in this subprotocol, the argument used must be resettably-sound.) In the second stage of the protocol, the prover proves that the common input G is Hamiltonian by running the basic proof of Hamiltonicity in parallel. However, the verifier must be prevented from obtaining answers to more than one query for a single series of prover commitments. (Otherwise, a cheating verifier can extract a cycle by resetting P.) This is achieved by having the verifier choose its queries by applying the pseudorandom function, committed to in the public-key, to the prover's commitments. In order to prevent the verifier from cheating, it continues by proving that it indeed computed the query correctly, where like before, the proof used is a resettably-sound zero-knowledge argument. Notice that in essence, the verifier's query string is determined by its public-key and the prover's set of commitments. Intuitively, this prevents a cheating verifier from gaining anything in a resetting attack. We note that given the pseudorandom function, the simulator can generate commitments that it can answer (without accessing the verifier V^*). Thus, the key point in the simulation is showing how the pseudorandom function (i.e. the secret seed which enables computing it) can be extracted.

Having discussed the simulation strategy, we now briefly mention an important point regarding soundness. The verifier V's instructions in our protocol ensure that V never applies the pseudorandom function to the same value twice (even if the prover tries to prove the same theorem twice and uses the same commitments in the second stage). Therefore, the verifier's queries as determined by the pseudorandom function are indistinguishable from the case that the verifier chooses its queries at random (and independently of other sessions).

Acknowledgements

We are grateful to Ran Canetti for collaboration in early stages of this research. Specifically, our public-key protocol is related to some of the ideas that were developed with Ran, with the crucial exception of not anticipating at that time the existence of (constant-round) resettably-sound zero-knowledge arguments of knowledge for $\mathcal{NP}$.

References

[1] B. Barak. How to Go Beyond the Black-Box Simulation Barrier. In these proceedings.

[2] B. Barak and O. Goldreich. CS Proofs Under a Standard Assumption. In preparation, 2001.

[3] B. Barak, O. Goldreich, S. Goldwasser and Y. Lindell. Resettably-Sound Zero-Knowledge and its Applications (preliminary full version of this work). Available at ePrint: `http://eprint.iacr.org`.

[4] M. Bellare and O. Goldreich. On Defining Proofs of Knowledge. In *Proc. of Crypto'92*, Springer-Verlag LNCS Vol. 740, pages 390–420.

[5] M. Bellare and M. Yung. Certifying Permutations: Non-Interactive Zero-Knowledge Based on Any Trapdoor Permutation. *Jour. of Crypto.*, Vol. 9 (3), pages 149–166, 1996.

[6] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In these proceedings.

[7] R. Canetti, O. Goldreich, S. Goldwasser, and S. Micali. Resettable Zero-Knowledge. In *32nd STOC*, pages 235–244, 2000.

[8] R. Canetti, J. Kilian, E. Petrank and A. Rosen. In *33rd STOC*, 2001.

[9] I. Damgård. Efficient Concurrent Zero-Knowledge in the Auxiliary String Model. In *Eurocrypt'00*, 2000.

[10] D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. In *23rd STOC*, pp. 542–552, 1991.

[11] C. Dwork and M. Naor. Zaps and their applications. In *41st FOCS*, pages 283–293, 2000.

[12] U. Feige, A. Fiat and A. Shamir. Zero-Knowledge Proofs of Identity. *Jour. of Crypto.*, Vol. 1, 1988, pages 77–94.

[13] U. Feige, D. Lapidot, and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs Under General Assumptions. *SICOMP*, Vol. 29 (1), pages 1–28, 1999.

[14] U. Feige and A. Shamir. Witness Indistinguishability and Witness Hiding Protocols. In *22nd STOC*, pages 416–426, 1990.

[15] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solution to Identification and Signature Problems. In *CRYPTO86*, Springer-Verlag LNCS 263, pages 186–189, 1987.

[16] M. Fürer, O. Goldreich, Y. Mansour, M. Sipser, and S. Zachos. On Completeness and Soundness in Interactive Proof Systems. *Advances in Computing Research: a research annual*, Vol. 5 (Randomness and Computation, S. Micali, ed.), pages 429–442, 1989.

[17] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. *JACM*, Vol. 33, No. 4, pages 792–807, 1986.

[18] O. Goldreich and A. Kahan. How To Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Jour. of Crypto.*, Vol. 9, pages 167–189, 1996.

[19] O. Goldreich and H. Krawczyk. On the Composition of Zero-Knowledge Proof Systems. *SIAM J. Computing*, Vol. 25, No. 1, pages 169–192, 1996.

[20] 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, No. 1, pp. 691–729, 1991.

[21] O. Goldreich and Y. Oren. Definitions and Properties of Zero-Knowledge Proof Systems. *Jour. of Crypto.*, Vol. 7, No. 1, pages 1–32, 1994.

[22] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SICOMP*, Vol. 18, pages 186–208, 1989. Preliminary version in *17th STOC*, 1985.

[23] J. Kilian and E. Petrank. Concurrent and Resettable Zero-Knowledge in Poly-logarithmic Rounds. In *33rd STOC*, 2001.

[24] S. Micali and L. Reyzin. Min-round resettable zero-knowledge in the public-key model. In *Eurocrypt 2001*.

[25] S. Micali and L. Reyzin. Soundness in the Public-Key Model. In *Crypto 2001*.

[26] L. Reyzin. *Zero-Knowledge with Public Keys.* Ph.D. Thesis, EECS Dept., MIT, June 2001.

[27] R. Richardson and J. Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. In *EuroCrypt99*, Springer-Verlag (LNCS 1592), pages 415–431.