Resettable Statistical Zero Knowledge

Sanjam Garg[1], Rafail Ostrovsky[1], Ivan Visconti[⋆2], and Akshay Wadia[1]

[1] Department of Computer Science, UCLA, USA
{sanjamg,rafail,awadia}@cs.ucla.edu

[2] Dipartimento di Informatica, University of Salerno, ITALY
visconti@dia.unisa.it

Abstract Two central notions of Zero Knowledge that provide very strong, yet seemingly incomparable security guarantees against malicious verifiers are those of Statistical Zero Knowledge and Resettable Zero Knowledge. The current state of the art includes several feasibility and impossibility results about the two notions *separately*. However, the challenging question of achieving Resettable Statistical Zero Knowledge (i.e., Resettable Zero Knowledge and Statistical Zero Knowledge *simultaneously*) for non-trivial languages is still open. In this paper, we show:

- **Resettable Statistical Zero Knowledge with efficient provers**: *Efficient-prover* Resettable Statistical Zero-Knowledge proof systems exist for all languages that admit hash proof systems (e.g., QNR, QR, $\mathcal{DDH}$, DCR). Furthermore, for these languages, as an application of our technique, we also construct a two-round resettable statistical witness-indistinguishable argument system.
- **Resettable Statistical Zero Knowledge with unbounded provers**: Under the assumption that sub-exponentially hard one-way functions exist, r$\mathcal{SZK} = \mathcal{SZK}$. In other words, every language that admits a Statistical Zero-Knowledge ($\mathcal{SZK}$) proof system also admits a Resettable Statistical Zero-Knowledge (r$\mathcal{SZK}$) proof system. (Further, the result can be re-stated unconditionally provided there exists a sub-exponentially hard language in $\mathcal{SZK}$). Moreover, under the assumption that (standard) one-way functions exist, all languages L such that the complement of L is random self reducible, admit a r$\mathcal{SZK}$, in other words: co-$\mathcal{RSR} \subseteq$ r$\mathcal{SZK}$.

The round complexity of all our proof systems is $\tilde{O}(\log \kappa)$, where κ is the security parameter, and all our simulators are *black-box*.

1 Introduction

The notion of a Zero-Knowledge (ZK, for short) Proof System introduced by Goldwasser, Micali and Rackoff [GMR89] is central in Cryptography. Since its introduction, the concept of a ZK proof has been extremely influential and useful for many other notions and applications (e.g., multi-party computation [GMW87], CCA encryption [NY90]). Moreover, the original definition has been then reformulated under several variations, trying to capture additional security guarantees. Well known examples are the notions of non-malleable ZK [DDN00] introduced by Dolev, Dwork and Naor, which concerns security against man-in-the-middle attacks, of ZK arguments introduced by Brassard, Chaum and Crepeau in [BCC88] where soundness is guaranteed only with respect to probabilistic polynomial-time adversarial provers, and of concurrent ZK [DNS98] introduced by Dwork, Naor and Sahai, which concerns security against concurrent malicious verifiers. Another important variant is that of Statistical Zero Knowledge [GMR89,BMO90,SV03], where it is guaranteed that a transcript of a proof will remain zero knowledge even against computationally unbounded adversaries.

An important model of security against malicious verifiers, known as *Resettable Zero-Knowledge*, was introduced by Canetti, Goldreich, Goldwasser and Micali in [CGGM00]. In this setting, the malicious verifier is allowed to *reset* the prover, and make it re-use its randomness for proving new theorems. Indeed, one of the main motivations for studying resettable ZK was to understand the consequences of re-using limited randomness on the zero-knowledge property. In [CGGM00], it was shown that *computational* zero-knowledge for all of $\mathcal{NP}$ is possible even in this highly adversarial setting.

[⋆] Work partially done while visiting UCLA.

Although resettable zero knowledge has received considerable attention since its inception (see for example, [BGGL01] [DGS09]) and the references therein), almost all the work has been focused on the computational setting.

In this work, we continue the line of research on resettable ZK by investigating the question of resettability when the zero-knowledge property is required to be statistical, i.e., Resettable Statistical Zero Knowledge. This model constrains the prover strategy severely: not only should the prover somehow re-use its limited randomness, it must do so in a way that makes the transcript of the proof statistically secure. Known solutions in the setting of computational resettable ZK involve converting prover's bounded randomness to unbounded pseudo-randomness by using pseudo-random functions (PRF). However, this approach fails in our case, as an unbounded adversary can break the PRF and gain critical information, breaking zero knowledge. In this paper, we develop a new technique to handle this problem. Using this technique, we study resettable statistical zero knowledge in the form of following two *distinct* questions.

- Do there exist *efficient-prover* resettable statistical ZK proofs? This question is motivated by practical applications of resettable ZK, for example, in smart cards. If a prover is to be implemented in a small device like a smart card, it is essential that the prover strategy is polynomial-time.
- What languages in $\mathcal{SZK}$ have resettable statistical ZK proofs? The class $\mathcal{SZK}$ is the class of problems which admit statistical zero-knowledge proofs. This question is purely theoretical in nature, and tries to ascertain the difficulty of achieving resettability where statistical zero-knowledge already exists. In this setting we consider prover's which are forced into giving multiple proofs using the same *limited* random coins. This work can be thought of a natural extension of the recent work on Concurrent Statistical Zero-Knowledge (c$\mathcal{SZK}$) [MOSV06,PTV08].

1.1 Our Contribution

In this paper we address the above questions and present the following results. We stress that our techniques may be of independent interest.

Resettable Statistical Zero Knowledge with efficient provers. We show that *efficient-prover* resettable statistical ZK proof systems exist for all languages in $\mathcal{SZK}$ that admit hash proof systems [CS02] (e.g., Quadratic Non-Residuosity (QNR), Decisional Diffie-Hellman ($\mathcal{DDH}$), Decisional Composite Residuosity (DCR)). Therefore, our techniques show that *efficient-prover* resettable statistical ZK proof systems also exist for non-trivial languages (like $\mathcal{DDH}$) where each instance is associated to more than one witness, where intuitively reset attacks are harder to deal with.[1] Furthermore, using our techniques, for these languages we also construct a two-round resettable statistical witness-indistinguishable argument system.

Resettable Statistical Zero Knowledge with unbounded provers. We show that if a family of sub-exponentially hard one-way functions exists then r$\mathcal{SZK} = \mathcal{SZK}$, i.e., all languages that admit a statistical ZK proof systems also admit a resettable statistical ZK proof system. If there exists an $\mathcal{SZK}$ language L which is (worst-case) sub-exponentially hard for all input length[2] then r$\mathcal{SZK} = \mathcal{SZK}$ without any additional assumptions, as it already implies the existence of sub-exponentially hard one-way functions. Informally, a sub-exponentially hard one-way function is a one-way function that is

[1] When there are multiple witnesses that can prove membership of an instance in a language, in a reset attack the adversarial verifier can potentially force the prover to reuse the same randomness for proving the same instance but using a different witness.

[2] If there exists a language $L \in \mathcal{SZK}$ such that for infinite sequence of input lengths, the worst-case decision problem for L is sub-exponentially-hard, [Ost91] implies that there exists a non-uniform sub-exponentially hard one-way functions for that sequence of input length.

secure against sub-exponential (2^{κ^ϵ} for some $0 < \epsilon < 1$) size circuits. Moreover, we show that if a family of (standard) one-way functions exists then co-$\mathcal{RSR} \subseteq$ r$\mathcal{SZK}$, i.e., all languages whose complement is a random self-reducible language (e.g. Graph Non-Isomorphism), admit a resettable statistical ZK proof system. Our results are achieved through a novel use of instance-dependent (ID, for short) commitment schemes, a new simulation technique, and a coin-tossing protocol that is secure under reset attacks that we build on top of a new ID commitment for all $\mathcal{SZK}$.

Our simulators are *black-box* and the round complexity of all our constructions is $\tilde{O}(\log \kappa)$ which is almost optimal considering the lower bounds achieved so far for black-box concurrent ZK [CKPR02,MY08].

We stress that since the very introduction in [CGGM00] of the notion of resettable ZK, our results are the first in establishing Resettable *Statistical* Zero Knowledge. We finally leave open an interesting question of proving that $\mathcal{SZK} =$ r$\mathcal{SZK}$ unconditionally or under relaxed complexity-theoretic assumptions and of establishing whether resettable statistical ZK *arguments* are achievable for all $\mathcal{NP}$.

As a final note, we remark upon the complexity of the verifiers in our protocols. Historically, the notion of $\mathcal{SZK}$ was developed with bounded verifiers (and unbounded distinguishers), for example, see [BMO90,Vad99]. Moving in the same direction, we obtain our results in this model, where the verifiers are computationally bounded. In subsequent literature on $\mathcal{SZK}$, the stronger notion of statistical zero-knowledge against *unbounded* verifiers was developed. In this scenario, the notion of resettability seems hard to achieve: unbounded verifiers can compute statistical correlations on the fly by making multiple reset queries to the prover. We leave the question of constructing such protocols or showing impossibility in a setting with unbounded verifiers as an open problem for future work.

1.2 Technical Difficulties and New Techniques

We begin by asking the general question: "Why is the problem of constructing resettable statistical zero-knowledge proof systems hard?" The problem lies in the fact that the prover has limited randomness and can be reset. Therefore, prover's messages are essentially a deterministic function of the verifier's messages, and the verifier can probe this function by resetting the prover and thereby obtaining information that might be useful for an unbounded distinguisher. We highlight the issues by demonstrating why existing techniques fail. The most well studied way of achieving resettable *computational* zero-knowledge proofs [CGGM00], is by using a pseudorandom function. In particular, very informally, using this technique the prover applies a pseudorandom function on the common input and the verifier's first messages (this message is called the *determining message*), which fixes all future messages of verifier, and uses the output as its random tape. Now, when the verifier resets and changes its determining message, prover's random tape changes, and thus, intuitively, the verifier does not gain any advantage by resetting the prover. However, for our goal of obtaining resettable *statistical* zero knowledge, this approach is not sufficient. In fact, intuitively, any protocol (as far as we know) in which there exists a message computed using both the witness and the randomness, where the randomness is fixed but the witnesses could change with theorem statements, can not be statistically "secure" in presence of reset attacks. Indeed, an adversarial verifier could interact multiple times with provers that use a fixed randomness but different statements and witnesses. This information can be used by an unbounded distinguisher to establish certain correlations among the values used in different executions, ultimately breaking the statistical ZK property. Because of these restrictions, previously known techniques, which were sufficient for resettable [CGGM00,BGGL01] and statistical ZK [MOSV06,GMOS07,GOS06a] independently, turn out to be insufficient for achieving both of them simultaneously.

In light of the intuition above resettable statistical ZK for non-trivial languages at first sight might be considered impossible to achieve. But, on the contrary we develop a new technique that overcomes the above problems. We demonstrate this new technique by considering a *toy version* of our protocol. The protocol consists of three phases. In the first phase the verifier sends a "special" *instance-dependent non-interactive* (ID, for short) *commitment* of a random string m to the prover. (In this commitment, if the prover is lying and $x \notin L$, then m will be undefined, while if $x \in L$, then m will be unique.) The second phase consists of a *PRS preamble* [PRS02]. Very roughly, in the PRS preamble the verifier commits to random shares of m, which are opened depending on the provers challenges. Finally, the prover is required to send m to the verifier. The prover can obtain m by extracting it from the commitment either efficiently using a witness in case of efficient-prover proofs, or running in exponential time in case of unbounded-prover proofs. We stress that when the theorem being proved is true the message m that can be extracted is unique. This can be thought of as a new abstraction of the Zero Knowledge protocol for *Graph Non-Isomorphism* (GNI). Consider a prover that wants to prove to a verifier that two graphs G_0 and G_1 are not isomorphic. Informally speaking, this protocol consists of the verifier sending a random permutation of G_b for a random b and the prover is required to guess b. The are other messages in the protocol, like the PRS phase, are required to help the simulator in the simulation. We stress that our new idea is this generalization. Protocols with this specific property have been studied before [MP06].

First, the protocol just described has the following property: every message sent by the prover is *public coin*[3] except its last message, which is *uniquely* determined by the first message of the verifier (we use [CGGM00] terminology and refer to it as the *determining message*). Most importantly, no message depends on the witness of the prover. It is this property that allows a simulator to generate a transcript that is statically close to the transcript generated in the interaction with a real prover. An honest prover uses a pseudorandom function on the common input and the determining message and uses the output as its random tape. A simulator can sample the messages from the *same* distribution as the real prover. Finally, the simulator will be able to obtain m by using rewinding capabilities, through a variation of a PRS rewinding strategy [PRS02]. The need for the variation arises from the fact that a simulator that uses pseudo-random coins does not gain anything by rewinding (i.e., after a rewind it would re-send the same message). We deal with this problem by having the simulator cleverly use pseudorandom coins for some messages while using pure random coins for others. We elaborate on this in § 4. This toy version, described above, illustrates the key ideas that we use in achieving simultaneously both resettable and statistical zero knowledge. To transform our toy version into a full proof system, for even the most basic languages that we consider in this paper, we need an extra instance-dependent primitive. But we defer this discussion to § 3 and § 5.

Second, our protocol also has the property that if the theorem is false then the prover has almost no chance (in the information-theoretic sense) of sending an accepting last message. This follows from the fact that the ID commitment from verifier is statistically hiding. This property guarantees soundness.

Unfortunately, the above ideas are insufficient to prove that r$\mathcal{SZK} = \mathcal{SZK}$. This is because statistically hiding non-interactive ID commitments, introduced by Chailloux, Ciocan, Kerenidis and Vadhan [CCKV08] for $\mathcal{SZK}$ are "honest-sender." To force the sender into using purely random coins we need a coin-flipping protocol secure against resetting senders. For this coin-flipping protocol an *ID commitment* scheme which is *computationally* binding with respect to a resetting sender for instances in the language and statistically hiding for instances not in the language, suffices. We will use some techniques introduced by Barak, Goldreich, Goldwasser and Lindell in [BGGL01] on top of a previous result of Ong and Vadhan [OV08] for obtaining such an ID commitment scheme.

However the more subtle problem arises in the use of pseudorandom functions. To obtain security against reset attacks, the coin-flipping message played by the receiver of the commitment must be computed by using a pseudorandom function. This again turns out to be insufficient for our analysis since the use of the pseudorandom function does not guarantee that the outcome of the coin-flipping protocol is a uniform string to be used in the honest-sender non-interactive ID commitment scheme. In order to solve this additional problem, we use sub-exponentially hard pseudorandom functions (constructed from sub-exponentially hard one-way functions). These stronger primitives

[3] Looking ahead, we will use a pseudorandom function to generate these messages.

have the additional property that they are secure against sub-exponential size circuits. This technique is referred to as *complexity leveraging*, and has been previously used in various applications (e.g., [CGGM00,Lin03,BLV03,DFN06,CEMT09,PW10,Wee10]). However, we stress that in all our constructions, the simulator runs in expected polynomial time, and the above assumptions play a role only inside our security proof.

Comparison with [MOSV06]. Before concluding this section, we point out an important difference between our approach and ideas developed by Micciancio, Ong, Sahai and Vadhan in [MOSV06]. In [MOSV06], the authors give *unconditional* constructions of concurrent statistical zero-knowledge proofs for many non-trivial problems. Like their construction we use similar ID commitments but our general approach and overall protocol is dramatically different from their approach. In [MOSV06], a compiler is constructed that (using ID commitments) provides a generic way to construct statistical zero-knowledge protocols. But, as pointed earlier, such a compiling technique along with standard resettability techniques [CGGM00] is not sufficient for us. Therefore, we develop our zero-knowledge protocol from scratch. This is needed because obtaining resettability along with statistical zero knowledge is different and for various reasons (as pointed earlier) harder than obtaining concurrent statistical zero knowledge. We further note that in fact our techniques imply that $\mathcal{SZK} = \mathrm{c}\mathcal{SZK}$ unconditionally. We discuss this further in Remark 2 in § 7.

Road map: We start by giving some preliminary definitions in § 2. We use three ID primitives in this paper. We elaborate on those in § 3. In § 4 we construct a resettable statistical ZK proof secure against partially honest verifiers. Then in § 5 we remove this limitation for certain classes of languages. In § 6, we construct the proof system that works for all language in $\mathcal{SZK}$. Finally, in § 7 we give some more applications of our technique.

2 Preliminaries

We assume basic familiarity with the notions [GMR89] of *Probabilistic Polynomial Time* (PPT for brevity) *Interactive Turing Machines* (ITM for brevity) and *protocol* (which is essentially a pair of PPT ITM's). We use the word *algorithm* interchangeably with an ITM.

We say that a function is *negligible* in κ if it is asymptotically smaller than the inverse of any fixed polynomial. More precisely, a function $\eta(\kappa)$ from non-negative integers to reals is called negligible in κ if for every constant $c > 0$, $\exists \kappa_c$ such that $\forall \kappa > \kappa_c$, $|\eta(\kappa)| < \kappa^{-c}$. Otherwise, $\eta(\kappa)$ is said to be *non-negligible* in κ. If M is a probabilistic ITM, $M(x)$ denotes the distribution of output generated by M on input x, over the random coins of M.

The *statistical difference* between two random variables X and Y taking values from a discrete set U (U is the set of all string in $\{0,1\}^n$) is defined as

$$\Delta(X,Y) \stackrel{def}{=} \max_{S \subset U} |\Pr[\, X \in S \,] - \Pr[\, Y \in S \,]| \, .$$

We say that the two distributions are *statistically indistinguishable* if $\Delta(X,Y)$ is negligible in n.

2.1 Interactive Proofs

Two probabilistic ITM's P and V define an interactive proof system in which the prover P is trying to convince the verifier V that $x \in L$. Let $\langle P, V \rangle(x)$ denote the output of the the verifier after the execution of the protocol between the prover P and the verifier V on common input x. The verifier outputs `accept` if it accepts in the protocol.

Definition 1 (Interactive Proof System). *An interactive protocol $\langle P, V \rangle(x)$ between a prover P and a probabilistic polynomial-time verifier V is said to be an interactive proof system for a language L with completeness error $c(\kappa)$ and soundness error $s(\kappa)$ if:*

– *Completeness. If $x \in L$, then $Pr[\langle P, V\rangle(x) = \textbf{\textit{accept}}] \geq 1 - c(|x|)$.*
– *Soundness. If $x \notin L$, then for all (possibly unbounded) adversarial prover's P^*, $Pr[\langle P^*, V\rangle(x) = \textbf{\textit{accept}}] \leq s(|x|)$.*

We require that $c(\kappa)$ and $s(\kappa)$ are negligible in κ.

If the honest prover algorithm in Definition 1 is probabilistic polynomial time then the interactive proof system is called an *efficient prover interactive proof system.*

Let $\text{view}_V(\langle P(z^P), V(z^V)\rangle(x))$ denote the random variable representing the contents of the random tape of V together with all the messages sent between P and V during the interaction on common input x, the prover's auxiliary input z^P and verifiers auxiliary input z^V. In general an adversarial verifier may receive an auxiliary input z^V. But we ignore it in our notation for the sake of simplicity. Also an honest prover executing in polynomial time receives as input a witness for every theorem x that it proves. In such cases we use w to represent the witness (the auxiliary input) that the prover gets as input. The witness is needed by efficient provers only.

Concurrent ZK We will use tools developed for the study of concurrent zero-knowledge within our protocols. We give the suitable definitions below.

Definition 2 (Concurrent Statistical ZK). *An interactive proof system $\langle P, V\rangle$ for a language L is said to be concurrent statistical ZK if for every probabilistic polynomial-time adversarial verifier V^* there exists a probabilistic polynomial-time machine $\mathcal{S}^*$ so that the following two distribution ensembles D_1 and D_2 are statistically indistinguishable: let each distribution be indexed by a sequence of common inputs $\overline{x} = x_1, \ldots, x_{\mathsf{poly}(\kappa)} \in L \cap \{0,1\}^{\mathsf{poly}(\kappa)}$ and a corresponding sequence of prover's auxiliary-inputs $\overline{w} = w_1, \ldots, w_{\mathsf{poly}(\kappa)}$,*

– *Distribution D_1 is defined by the following random process which depends on P and V^*.*
  1. *Let $P^i = P_{x_i,w_i}$ denote the prover that is trying to prove to the verifier that $x_i \in L$. w_i is the auxiliary input of the prover. Every time the prover P^i is instantiated it uses fresh randomness.*
  2. *Machine V^* is allowed to run polynomially-many sessions with P^i's. We allow V^* to send arbitrary messages to each of P^i, and obtain the responses of P^i to such messages.*
  3. *Once V^* decides it is done interacting with P^i's, it (i.e., V^*) produces an output based on its view of these interactions. Let us denote this output by $\langle P(\overline{w}), V^*\rangle(\overline{x})$.*
– *Distribution D_2 is the output of $\mathcal{S}^*(\overline{x})$.*

In case there exists a universal probabilistic polynomial time machine, $\mathcal{S}$, so that $\mathcal{S}^*$ can be implemented by letting $\mathcal{S}$ have oracle access to V^*, we say that P is concurrent statistical ZK *via a black-box simulation.*

PRS Preamble from [PRS02]. A PRS preamble is a protocol between a committer $\mathcal{C}$ and a receiver $\mathcal{R}$ that consists of two main phases, namely, (a) the commitment phase, and (b) the challenge-response phase. Let k be a parameter that determines the round-complexity of the protocol. Then, in the commit phase, very informally, the committer commits to a secret string σ and k^2 pairs of its 2-out-of-2 secret shares. The challenge-response phase consists of k iterations, where in each iteration, very informally, the committer "opens" k shares, one each from k different pairs of secret shares as chosen by the receiver.

The goal of this protocol is to enable the simulator to be able to rewind and extract the "preamble secret" σ with overwhelming probability. In the concurrent setting, rewinding can be difficult since one may rewind to a time step that precedes the start of some other protocol [DNS98]. However, as it has been demonstrated in [PRS02], there is a fixed "time-oblivious" rewinding strategy that the simulator can use to extract the preamble secrets from every concurrent cheating committer, except with negligible probability. Moreover this works as long as $k = \tilde{\Omega}(\log \kappa)$ for some positive ϵ. We refer to this as the PRS rewinding strategy or the PRS simulation strategy. We refer the reader to [PRS02] for more details.

Resettable/Statistical Zero Knowledge. In this paper we consider statistical [GMR89,BMO90,SV03] and resettable [CGGM00] notions of zero-knowledge. The notion of resettability requires that a protocol remains zero-knowledge even if the verifier can reset the prover. The notion of statistical zero knowledge provides security guarantees against unbounded distinguishers. This paper constructs resettable statistical zero-knowledge proof systems. In other words we try to achieve both the resettability and the statistical guarantees simultaneously.

Definition 3 (Resettable Statistical ZK). *An interactive proof system $\langle P, V\rangle$ for a language L is said to be resettable statistical ZK if for every probabilistic polynomial-time adversarial verifier V^* there exists a probabilistic polynomial-time machine $\mathcal{S}^*$ so that the following two distribution ensembles D_1 and D_2 are statistically indistinguishable: Let each distribution be indexed by a sequence of common inputs $\overline{x} = x_1, \ldots, x_{\mathsf{poly}(\kappa)} \in L \cap \{0,1\}^{\mathsf{poly}(\kappa)}$ and a corresponding sequence of prover's auxiliary-inputs $\overline{w} = w_1, \ldots, w_{\mathsf{poly}(\kappa)}$,*

- *Distribution D_1 is defined by the following random process which depends on P and V^*.*
  1. *Randomly select and fix $t = \mathsf{poly}(\kappa)$ random tapes, $\omega_1, \ldots, \omega_t$, for P, resulting in deterministic strategies $P^{(i,j)} = P_{x_i,w_i,\omega_j}$ defined by $P_{x_i,w_i,\omega_j}(\alpha) = P(x_i, w_i, \omega_j, \alpha)$, for $i, j \in \{1, \ldots t\}$. Each $P^{(i,j)}$ is called an* incarnation *of P.*
  2. *Machine V^* is allowed to run polynomially-many sessions with $P^{(i,j)}$'s.*
  3. *We allow V^* (*interleaving version *[CGGM00]) to send arbitrary messages to each of $P^{(i,j)}$, and obtain the responses of $P^{(i,j)}$ to such messages.*
  4. *Once V^* decides it is done interacting with $P^{(i,j)}$'s, it (i.e., V^*) produces an output based on its view of these interactions. Let us denote this output by $\langle P(\overline{w}), V^*\rangle(\overline{x})$.*
- *Distribution D_2 is the output of $\mathcal{S}^*(\overline{x})$.*

In case there exists a universal probabilistic polynomial time machine, $\mathcal{S}$, so that $\mathcal{S}^*$ can be implemented by letting $\mathcal{S}$ have oracle access to V^*, we say that P is resettable statistical ZK *via a black-box simulation.*

Sub-exponentially hard one-way functions. A sub-exponentially hard one-way function is a one-way function that is hard to invert even by sub-exponential (2^{κ^ϵ} for some $1 > \epsilon > 0$) size circuits. They imply the existence of *sub-exponentially hard pseudorandom functions*. We stress that we need this assumption only for proving that $\mathcal{SZK} = \mathrm{r}\mathcal{SZK}$.

2.2 Complexity Assumptions

Definition 4 (Pseudorandom function (PRF)). *A family of functions $\{f_s\}_{s\in\{0,1\}^*}$ is called pseudorandom if for all adversarial PPT machines $\mathcal{A}$, for every positive polynomial $p(\cdot)$, and sufficiently large n's it holds that*

$$\left|Pr[\mathcal{A}^{f_s}(1^n) = 1)] - Pr[\mathcal{A}^{R}(1^n) = 1)]\right| < \frac{1}{p(n)}$$

where $|s| = n$ and R denotes a random function.

A sub-exponentially hard one-way function is a one-way function that is hard to invert even by sub-exponential size circuits.

Definition 5 (Sub-Exponentially Hard One-Way Functions (OWF)). *A one-way function $f : \{0,1\}^* \to \{0,1\}^*$ is* sub-exponentially hard *if for every circuit $\mathcal{A}$ of size 2^{n^ϵ} for some constant $1 > \epsilon > 0$, every positive polynomial $p(\cdot)$, and sufficiently large n's, and any auxiliary input z it holds that*

$$Pr[\mathcal{A}(f(U_n), z) \in f^{-1}(f(U_n))] < \frac{1}{p(n)}$$

where U_n denotes a random variable uniformity distributed over $\{0,1\}^n$.

A sub-exponentially hard one-way function implies the existence of *sub-exponentially hard pseudorandom functions* which can be defined in a similar manner.

3 Instance-Dependent Commitments and Proofs

In this section we construct three instance-dependent primitives, that we use in this paper: (1) a non-interactive instance-dependent commitment scheme, (2) an interactive instance-dependent commitment scheme, and finally (3) an instance-dependent argument system.

Non-Interactive Instance-Dependent Commitment Scheme. An important tool that we will re-define, construct and use in our proof systems, is that of "special" *non-interactive instance-dependent* (ID, for short) *commitment schemes.* A commitment scheme allows one party (referred to as the *sender*) to commit to a value while keeping it *hidden*, with the ability to *reveal* the committed value later. Commitments also have the property that once the sender commits to a value, it can not change its mind later. This property is refereed to as the *binding* property. In certain settings commitment schemes, for which these properties are not required to hold simultaneously, suffice. Such schemes are parameterized by a value x and a language L and either the binding or the hiding property holds depending upon the value x. These schemes are referred to as ID commitment schemes [CCKV08]. Typically, the ID commitment schemes that have been considered in the literature require hiding property to hold when $x \in L$ and binding property to hold otherwise. We actually need the reverse properties, i.e., we need hiding property when $x \notin L$ and binding property otherwise.

In particular we consider an ID commitment scheme with further special properties. We require that the commitment scheme be statistically binding for $x \in L$ and statistically hiding otherwise. In other words we want binding and hiding properties to hold against unbounded adversaries. Also we require that our ID commitment scheme be secure against a resetting sender. This always holds when the commitment scheme is *non-interactive.* All the non-interactive ID commitments that we consider are statistically hiding. So to simplify notation we refer to a non-interactive instance-dependent commitment scheme with perfect (binding holds with probability 1) binding and statistical hiding as a *perfect non-interactive ID commitment.* Similarly, we refer to a non-interactive instance-dependent commitment scheme with statistical binding and statistical hiding as a *statistical non-interactive ID commitment.*

Since the commitment is statistically binding, when $x \in L$, the committed value can always (with overwhelming probability) be extracted in exponential time. Extractability instead becomes tricky when the extractor has to run in polynomial time. We will call an ID commitment scheme *efficiently extractable* if when $x \in L$ then there exists an extractor that takes as input a witness for the membership of x in L and the commitment, and outputs in polynomial-time the committed message.

It turns out that perfect non-interactive ID commitment schemes [IOS97,TW87,SCPY94] are actually known to exist for all languages in co-$\mathcal{RSR}$. co-$\mathcal{RSR}$ is the class of languages such that the complement of each of these languages is random self-reducible. Another class of languages that is amenable to our techniques is the class of languages that are in $\mathcal{SZK}$ and that admit a hash proof system. Observing that these languages imply instance-dependent primitives that are analogous to ID commitments described above, we get efficient-prover resettable statistical ZK proof systems for this interesting class. In particular, for $\mathcal{DDH}$ (the language that consists of all Diffie-Hellman quadruples and that admits two different witnesses for proving the membership of a quadruple to the language), we give an ad-hoc separate ID commitment scheme highlighting how our techniques work with multiple witnesses.

We notice that for the whole $\mathcal{SZK}$ we only know a weak form of statistical non-interactive ID commitment scheme where statistical binding holds with respect to honest senders only. We formalize these notions in the next subsection.

3.1 Perfect (resp. Statistical) Non-Interactive ID Commitment Schemes

Perfect (resp. statistical) non-interactive ID commitment schemes are one of the key tools that we use in this work. Instance-dependent (ID, for short) commitment schemes are parameterized by a value x

and a language L. Typically, ID commitment schemes (e.g., [CCKV08]) that have been considered in the literature require hiding property to hold when $x \in L$ and binding property to hold otherwise. As highlighted in Section 1.2, we actually need the reverse properties, i.e. we need hiding property when $x \notin L$ and binding property otherwise.

Definition 6 (Instance-Dependent Function). *An auxiliary-input function (or Instance-Dependent Function) ensemble is a collection of functions $\mathcal{F} = \{f_x : \{0,1\}^{p(|x|)} \to \{0,1\}^{q(|x|)}\}_{x \in \{0,1\}^*}$, where p and q are polynomials. We call $\mathcal{F}$ probabilistic-polynomial-time computable, if there is a probabilistic polynomial-time algorithm F such that for every $x \in \{0,1\}^*$ and $y \in \{0,1\}^{p(|x|)}$, we have $F(x, y; r) = f_x(y; r)$, where r denotes the randomness used by the function F.*

Definition 7 (Non-Interactive Instance-Dependent Commitment). *A Non-Interactive Instance-Dependent Commitment Scheme $\mathcal{COM}$ with respect to (L, x), between sender and a receiver (S, R), consists of two probabilistic polynomial time algorithms: $\mathsf{C}_{L,x}$, used in the commit phase; and $\mathsf{V}_{L,x}$, used in the reveal phase. All parties receive the security parameter κ as input.*

Commit Phase: *In the commit phase, sender S receives a private input $m \in \{0,1\}^{\ell_0}$ where ℓ_0 is polynomial in the security parameter κ. It uses its private random bits $r \in \{0,1\}^{\ell_1}$, where ℓ_1 is polynomial in κ, and evaluates $C \leftarrow \mathsf{C}_{L,x}(m; r)$. It then sends C to the receiver. Also, S outputs private information $\mathsf{pvt} = (m, r)$.*

Reveal Phase: *In the reveal phase the sender S sends the private information $\mathsf{pvt} = (m, r)$ to the receiver R. At the end of this phase, R executes $\mathsf{V}_{L,x}(C, \mathsf{pvt})$, where C is the commitment value previously received and pvt is the private information revealed by the sender, that returns accept or reject. R accepts m as opening if V outputs accept and rejects otherwise.*

We next define the correctness and security requirements of a perfect non-interactive instance-dependent commitment scheme.

Definition 8 (Perfect (resp. Statistical) Non-Interactive Instance-Dependent Commitment Scheme). *Let $R_L, L, \mathcal{COM}$ be as defined above. Then, $\mathcal{COM}$ is called a perfect (resp. statistical) non-interactive instance-dependent commitment scheme, if the following properties hold.*

Correctness. *For all n, all $x \in \{0,1\}^n$, all $m \in \{0,1\}^{\ell_0}$ and all random $r \in \{0,1\}^{\ell_1}$, all $(C, (m, r))$ pairs such that $C = \mathsf{C}_{L,x}(m; r)$, it is the case that $\mathsf{V}_{L,x}$ always outputs accept on input $(C, (m, r))$.*

Binding. *The perfect (resp. statistical) non-interactive instance-dependent commitment scheme has the perfect (resp. statistical) binding property.*

Perfect Binding. *For all n, all $x \in L \cap \{0,1\}^n$, all $m, m' \in \{0,1\}^{\ell_0}$ and $r, r' \in \{0,1\}^{\ell_1}$, such that $C = \mathsf{C}_{L,x}(m; r), \mathsf{pvt} = (m, r)$ and $\mathsf{pvt}' = (m', r')$ and $m \neq m'$ we have,*

$$\Pr\left[\, \mathsf{V}_{L,x}(C, \mathsf{pvt}) = \mathsf{accept} \wedge \mathsf{V}_{L,x}(C, \mathsf{pvt}') = \mathsf{accept} \,\right] = 0.$$

For perfect non-interactive instance-dependent commitment scheme, given a transcript of the commitment phase, C, there is a unique message that the Receiver will accept in the reveal phase.

Statistical Binding. *For all n, when r is chosen randomly in $\{0,1\}^{\ell_1}$, then for all $x \in L \cap \{0,1\}^n$, all $m, m' \in \{0,1\}^{\ell_0}$, $r' \in \{0,1\}^{\ell_1}$, such that $C = \mathsf{C}_{L,x}(m; r), \mathsf{pvt} = (m, r)$ and $\mathsf{pvt}' = (m', r')$ and $m \neq m'$ we have, $\Pr[\, \mathsf{V}_{L,x}(C, \mathsf{pvt}) = \mathsf{accept} \wedge \mathsf{V}_{L,x}(C, \mathsf{pvt}') = \mathsf{accept} \,]$ is negligible in κ.*

Statistical Hiding. *For all n, all $x \notin L \cap \{0,1\}^n$, all (possibly unbounded) adversaries $\hat{R}$, $\forall m, m' \in \{0,1\}^{\ell_0}$ such that $m \neq m'$, the following distributions are statistically indistinguishable: $\{r \xleftarrow{\$} \{0,1\}^*; \mathsf{C}_{L,x}(m; r)\}$ and $\{r' \xleftarrow{\$} \{0,1\}^*; \mathsf{C}_{L,x}(m'; r')\}$.*

When this scheme is used as subprotocol and the sender is allowed to cheat then the statistical binding property does not always hold. If this is case then we will explicitly specify "honest senders" to stress that the malicious player behaves honestly when computing the non-interactive commitment.

Note that for a perfect non-interactive instance-dependent commitment scheme when $x \in L$, the commitment scheme is perfectly binding, and thus, given the commitment C, the *committed message* is fixed. We denote this message by $\sigma(C)$. The commitment scheme defined above, will be *extractable* in the following sense: if $x \in L$, then there exists an algorithm called the *extractor*, that on input the transcript of the first phase C outputs the message $\sigma(C)$. This extractor could be efficient or it could might be computationally unbounded. For a statistical non-interactive instance-dependent commitment scheme, when $x \in L$ the commitment scheme is statistically binding, and thus, given transcript C, the *committed message* is fixed except with a negligible probability. Hence, there exists an *extractor*, that on input the transcript of the first phase C outputs the message $\sigma(C)$ except with negligible probability. Again this extractor could be computationally unbounded.

Since the definition of extraction with an unbounded extractor is trivial, we now focus on defining efficient extractability, and later we will show efficient extractors matching our definition.

Let $R_L(\cdot,\cdot)$ be a polynomially-bounded, polynomial-time relation, with L as the corresponding language. Let $x \in \{0,1\}^n$, where n is polynomial in the security parameter κ. A Non-Interactive Instance-Dependent Commitment with respect to (L, x), denoted by $\mathcal{COM}$, is defined as follows.

Definition 9 (Efficiently-Extractable Non-Interactive Instance-Dependent Commitment Scheme). *Let $R_L, L, \mathcal{COM}$ be as defined previously. Then, $\mathcal{COM}$ is called an* ***efficiently extractable*** *instance-dependent commitment scheme if, for all $(x, w) \in R_L$, and commitment string C that can be generated by the commitment algorithm $\mathsf{C}_{L,x}(\cdot)$, there exists a polynomial time extractor $\mathsf{E}_{L,x}$, such that for all commitment strings C that can be generated by the commitment algorithm $\mathsf{C}_{L,x}(\cdot)$, the extractor, on input C and a witness w, outputs $\sigma(C)$.*

ID Commitments for co-$\mathcal{RSR}$. The class of random self-reducible languages was introduced in [TW87,AFK89] and has been since then referred to as $\mathcal{RSR}$.

Definition 10 ($\mathcal{RSR}$ [SCPY94]). *Let $\mathcal{N}$ be a countably infinite set and, for each $x \in \mathcal{N}$, let R_x be a relation verifiable in polynomial-time. Define the domain of R_x as the set $\{z | \exists w \text{ such that } (z, w) \in R_x\}$ then we say when $(z, w) \in R_x$ that w is an x-witness for z. The language $L_R = \{(x, z) | \exists w \text{ such that } (z, w) \in R_x\}$ is random self reducible (in $\mathcal{RSR}$) if there exists a polynomial time algorithm $Sample_{L_R}$ that, on input x, z, r outputs y such that $(x, y) \in L_R$ and*

1. *If r's bits are random, uniform, and independent then y is uniformly distributed over the domain of R_x.*
2. *There exists a polynomial time algorithm A_1 (A_2) that, on input x, y, r and an x-witness for y (for z) outputs an x-witness for z (for y).*
3. *There exists an algorithm $Generate_{L_r}$ that, on input x, outputs in expected polynomial time a pair $(z, w) \in R_x$, with z uniformly distributed over the domain of R_x, and w uniformly distributed over all x-witnesses for z.*

$\mathcal{RSR}$ includes the language of Graph Isomorphism, the language of Quadratic Residuosity, and the language of the multiplicative subgroup of $\mathbb{Z}_p^*$ generated by g (p a prime).

Definition 11 (co-$\mathcal{RSR}$). *The class of languages such that the complement of each of these languages is in $\mathcal{RSR}$ is said the class co-$\mathcal{RSR}$.*

Definition 12 (r$\mathcal{SZK}$). *A language L is said to be in the complexity class r$\mathcal{SZK}$ if L admits a resettable statistical ZK proof system.*

It is known that every language $L \in$ co-$\mathcal{RSR}$ admits a perfect non-interactive ID commitment scheme [IOS97,TW87,SCPY94]. A folklore theorem and an explicit construction[4] can be found in Lemma 5.2 of [KMV07].

Lemma 1. *Every language $L \in$ co-$\mathcal{RSR}$ has a perfect non-interactive ID commitment scheme.*

Hash Proof Systems. We recall the notion of hash proof systems as introduced by Cramer and Shoup [CS02]. We use the formalization as in [CHK10] that does not use the concept of subset membership problem, and where only the proof system itself is relevant.

Definition 13 (Hash Proof System).

Let L be a language and let ϵ be a real number with $0 \leq \epsilon < 1$. A hash proof system $\mathsf{HPS}_L = (\mathcal{V}_L, \mathcal{P}_L, \alpha, \Pi_L, \mathsf{Pub}_L, \mathsf{Priv}_L)$ with error probability ϵ consists of the following:

– *A finite non-empty set $\mathcal{V}_L$, where the verifier samples a secret verification-key from, to enable him to check proofs.*
– *A finite non-empty set $\mathcal{P}_L$ and a function $\alpha : \mathcal{V}_L \to \mathcal{P}_L$: this maps a verification key to its projection, which is an auxiliary input for the prover to construct a proof.*
– *A non-empty finite set Π_L: this is where proof strings will be sampled from.*

Furthermore, efficient algorithms for the following tasks exist:

– *Sampling with the uniform distribution from $\mathcal{V}_L$.*
– *Computing $\alpha(v) \in \mathcal{P}_L$ when given $v \in \mathcal{V}_L$.*
– *The proof $\pi \in \Pi_L$ can be computed:*
  - *either, by an efficient* public evaluation algorithm $\pi \leftarrow \mathsf{Pub}_L(x, \alpha(v), w)$ *given the statement $x \in L$, along with the projection $\alpha(v)$ and a witness w for $x \in L$.*
  - *or, by an efficient* private evaluation algorithm $\pi \leftarrow \mathsf{Priv}_L(x, v)$ *given the statement $x \in L$ and the verification key $v \in \mathcal{V}_L$ itself.*

The following security properties hold (Soundness and Uniqueness hold even for an unbounded adversary):

– *Completeness. If indeed $x \in L$, a proof $\pi \in \Pi_L$ thus computed is accepted when verified using the secret verification key v. This verification is performed efficiently by the verifier.*
– *Soundness. For every $x \notin L$, every projection $u \in \mathcal{P}_L$, and every purported proof $\tilde{\pi} \in \Pi_L$: the probability (over uniform $v \in \mathcal{V}_L$ with $\alpha(v) = u$) that $\tilde{\pi}$ will be accepted is at most ϵ.*
– *Uniqueness. The proof $\pi \in \Pi_L$ is unique. In the verification procedure referred to above, the verifier actually first computes π' from $x \in L$ and the verification key v. The decision is then made by checking whether $\pi' = \pi$. In other words, the verifier can compute the proof itself from seeing the statement, using his secret verification key.*

As in [CHK10], we note that the soundness error ϵ can be reduced to a negligible function by running copies based on independently selected keys in parallel.

Analogues of ID Commitment for Hash Proof Systems. We note that a hash proof system is a simple instance-dependent primitive that can be thought of as an analogue of ID commitment schemes described in previous sections. Indeed, it turns out that hash proof systems enjoy the crucial properties that we required from non-interactive ID commitments in the previous sections. Therefore we can use an HPS in our resettable protocols as we would use ID commitments for other languages. For convenience, we describe a HPS as a two party protocol between a sender S and a receiver R. Consider the protocol in Fig. 1.

We reiterate two key properties we will use in our constructions.

[4] Their construction is for $\mathcal{RSR}$ languages with the binding (for false theorems) and hiding (for true theorems) properties reversed. This directly implies Lemma 1.

Common Input: Instance $x \in \{0,1\}^n$.
Receiver's Input: Witness w for x.

1. S chooses uniformly at random a verification-key $v \in \mathcal{V}_L$, and sends $\alpha(v)$ to R.
2. R computes $\hat{\pi} = \mathsf{Pub}_L(x, \alpha(v), w)$, and sends $\hat{\pi}$ to S, who accepts if $\hat{\pi} = \mathsf{Priv}_L(x, v)$.

Figure 1. A Hash Proof System $\mathsf{HPS}_L = (\mathcal{V}_L, \mathcal{P}_L, \alpha, \Pi_L, \mathsf{Pub}_L, \mathsf{Priv}_L)$ for a language L reformulated as a two-party protocol (S, R).

1. When $x \in L$, the proof $\pi = \mathsf{Priv}_L(x, \alpha(v))$ is unique. Moreover it is extractable since the receiver can compute π using the witness w, making the sender accept.
2. When $x \notin L$, it follows from soundness of the hash proof system, that the probability that S accepts is negligible (even against unbounded receivers).

Efficiently-Extractable Perfect Non-Interactive ID Commitment Scheme for $\mathcal{DDH}$. Now we show how to construct an efficiently extractable perfect non-interactive instance-dependent commitment scheme for the $\mathcal{NP}$-language $\mathcal{DDH}$. We show this additional construction as an evidence that our techniques are general and indeed can be applied also to languages where multiple witnesses are associated to an instance. This is indeed one of the properties that a resetting verifier can try to exploit during a reset attack. Indeed, it can ask the prover to run multiple times a proof of the same statement with same randomness by changing the witness.

Let G be a cyclic group of order t where the DDH assumption is believed to hold. We define the following language, $\mathcal{DDH}$,

$$\mathcal{DDH} = \{ (g_1, g_2, g_3, g_4) \in G^4 \,:\, \mathsf{dlog}_{g_1}(g_2) \cdot \mathsf{dlog}_{g_1}(g_3) = \mathsf{dlog}_{g_1}(g_4) \}.$$

Here, $\mathsf{dlog}_g(h)$ refers to the discrete logarithm of h to the base g in group G; that is, $\mathsf{dlog}_g(h) = k$ iff $h^k = g$. Note that each instance $(g_1, g_2, g_3, g_4) \in \mathcal{DDH}$ has two witnesses of membership: $\mathsf{dlog}_{g_1}(g_2)$ and $\mathsf{dlog}_{g_1}(g_3)$. Indeed, $g_3^{\mathsf{dlog}_{g_1}(g_2)} = g_4$ iff $(g_1, g_2, g_3, g_4) \in \mathcal{DDH}$, and $g_2^{\mathsf{dlog}_{g_1}(g_3)} = g_4$ iff $(g_1, g_2, g_3, g_4) \in \mathcal{DDH}$.

As above, let G be a group of order t, where the DDH assumption is believed to hold. We now instantiate our instance dependent commitment scheme from the language $\mathcal{DDH}$. Consider Fig. 2.

Common Input: Instance (g_1, g_2, g_3, g_4).
Sender's Input: $m \in G$.
Commit Phase: The sender S chooses random elements $r_1, r_2, r_1', r_2' \in Z_t$. Now, S computes:

– $\alpha_1 = g_4^{r_1} g_2^{r_2}/m$ and $\alpha_2 = g_3^{r_1} g_1^{r_2}$.
– $\beta_1 = g_4^{r_1'} g_3^{r_2'}/m$ and $\beta_2 = g_2^{r_1'} g_1^{r_2'}$.

Finally, it sends $(\alpha_1, \alpha_2, \beta_1, \beta_2)$ to the receiver R.
Decommit Phase: The sender S sends r_1, r_2, r_1', r_2', m to the receiver R, who checks if $\alpha_1, \alpha_2, \beta_1, \beta_2$ were constructed correctly. Receiver R accepts if this is the case, and rejects otherwise.

Figure 2. Efficiently Extractable Non-Interactive Instance-Dependent Commitment Scheme (S, R) for $\mathcal{DDH}$.

Lemma 2. *Let G be a group of order t where the DDH assumption holds. Then, $\mathcal{DDH}$ admits a non-interactive instance dependent commitment scheme, with efficient extraction.*

Proof. We now prove that the commitment scheme presented in Fig. 2 is perfectly binding and efficiently extractable when $(g_1, g_2, g_3, g_4) \in \mathcal{DDH}$, and statistically hiding when $(g_1, g_2, g_3, g_4) \notin \mathcal{DDH}$.

Perfect Binding and Efficient Extraction. Let $a := \mathsf{dlog}_{g_1}(g_2)$, $b := \mathsf{dlog}_{g_1}(g_3)$ and $c := \mathsf{dlog}_{g_1}(g_4)$. Then, as $c = ab$, we have,

$$\frac{\alpha_2^a}{\alpha_1} = \left(\frac{(g^{br_1+r_2})^a}{g^{abr_1+ar_2}}\right) m \;=\; m,$$
$$\frac{\beta_2^b}{\beta_1} = \left(\frac{(g^{ar'_1+r'_2})^b}{g^{abr'_1+br'_2}}\right) m \;=\; m.$$

For efficient extraction, the receiver computes either α_2^a/α_1 or β_2^b/β_1, depending upon whether it has witness a or b.

Statistically Hiding. We show that if $(g_1, g_2, g_3, g_4) \notin \mathcal{DDH}$, then even an unbounded adversary does not obtain any information about message m from the commitment. Let $\mathsf{dlog}_{g_1}(m) = \tau$, $\mathsf{dlog}_{g_1}(g_2) = a$, $\mathsf{dlog}_{g_1}(g_3) = b$ and $\mathsf{dlog}_{g_1}(g_4) = c$. The information that the unbounded adversary has after the commitment phase can be captured by the following linear system (all discrete logarithms are base g_1):

$$\begin{aligned}\mathsf{dlog}(\alpha_1) &= cr_1 + ar_2 - \tau\\ \mathsf{dlog}(\alpha_2) &= br_1 + r_2\\ \mathsf{dlog}(\beta_1) &= cr'_1 + br'_2 - \tau\\ \mathsf{dlog}(\beta_2) &= ar'_1 + r'_2.\end{aligned}$$

This is a system with four unknowns $(r_1, r_{2,}, r'_1, r'_2)$ and four equations. If $c \neq ab$, then the corresponding matrix has full rank (that is, rank 4). Thus, for every setting of τ, there exist (r_1, r_2, r'_1, r'_2) that will generate the same commitment. Thus, given the transcript of the commitment phase, all messages are equally likely.

Honest-Sender Non-Interactive ID Commitment Scheme for $\mathcal{SZK}$.

Lemma 3 ([CCKV08]). *Every language $L \in \mathcal{SZK}$ has a honest-sender statistical non-interactive ID commitment scheme.*

3.2 Interactive ID Commitment Scheme.

We use an *interactive ID commitment scheme* $\mathsf{Com}_{L,x} = (S_x, R_x)$, where S_x and R_x are the sender and the receiver respectively, with common input x. This ID commitment scheme is *computationally binding* against a *resetting* sender when the instance x is in the language, and is *statistically hiding* otherwise. Very roughly, we construct such a scheme by using the constant-round public-coin ID commitment scheme of [OV08]. This scheme has statistical binding and statistical hiding properties. We make it secure under resetting senders by having the receiver determine its messages by applying a pseudo-random function (similarly to Proposition 3.1 in [BGGL01]) to the transcript so far. Because of this, the statistical binding property is degraded to computational[5] binding. We stress that unlike the non-interactive ID commitment described earlier, we will not need any extractability from these commitments. We obtain this new ID commitment scheme for all of $\mathcal{SZK}$ under the assumption that one-way functions exist.We provide the details below.

[5] However, looking ahead we note that, computational binding will be sufficient for our applications since the role of the sender will be played by a polynomially bounded party.

Definition 14 (Instance-Dependent Commitment, Definition 4 from [OV08]). *An* instance-dependent commitment scheme *is a family of protocols* $\{\mathsf{Com}_x\}_{x\in\{0,1\}^*}$ *with the following properties:*

1. *Scheme* Com_x *proceeds in two stages: a* commit stage *and a* reveal stage*. In both stages, the* sender *and* receiver *receive instance x as common input, and hence we denote the sender and receiver as S_x and R_x, respectively, and write* $\mathsf{Com}_x = (S_x, R_x)$*.*
2. *At the beginning of the commit stage, sender S_x receives a private input $b \in \{0,1\}$, which denotes the bit that S is supposed to commit to. At the end of the commit stage, both sender S_x and receiver R_x output a* commitment c*.*
3. *In the reveal stage, sender S_x sends a pair (b, d), where d is the* decommitment *string for bit b. Receiver R_x accepts or rejects based on x, b, d, and c.*
4. *The sender S_x and receiver R_x algorithms are computable in polynomial time (in $|x|$), given x as auxiliary input.*
5. *For every $x \in \{0,1\}^*$, R_x will always accept (with probability 1) if both the sender S_x and receiver R_x follow their prescribed strategy.*

An instance-dependent commitment scheme $\{\mathsf{Com}_x = (S_x, R_x)\}_{x\in\{0,1\}^*}$ *is* public coin *if for every $x \in \{0,1\}^*$, all messages sent by R_x are independent random coins.*

To simplify the notation we use Com_x or (S_x, R_x) to denote an instance-dependent commitment scheme $\{\mathsf{Com}_x\}_{x\in\{0,1\}^*}$. The hiding and binding properties of standard commitments extend in a natural way to their instance-dependent analogues.

Definition 15 (Definition 5 from [OV08]). *An instance-dependent commitment scheme* $\mathsf{Com}_x = (S_x, R_x)$ *is* statistically *hiding on $I \subseteq \{0,1\}^*$ if for every R^*, the ensembles $\{view_{R^*}(S_x(0), R^*)\}_{x\in I}$ and $\{view_{R^*}(S_x(1), R^*)\}_{x\in I}$ are statistically indistinguishable, where random variable $view_{R^*}(S_x(b), R^*)$ denotes the view of R^* in the commit stage interacting with $S_x(b)$.*

Definition 16 (Statistical [resp, computational] Binding, Definition 6 from [OV08]). *An instance-dependent commitment scheme* $\mathsf{Com}_x = (S_x, R_x)$ *is* statistically *[resp., computationally] binding on $I \subseteq \{0,1\}^*$ if for every [resp., nonuniform PPT] S^*, there exists a negligible function ε such that for all $x \in I$ the malicious sender S^* succeeds in the following game with probability at most $\varepsilon(|x|)$.*

S^* interacts with R_x in the commit stage obtaining commitment c. Then S^* outputs pairs $(0, d_0)$ and $(1, d_1)$, and succeeds *if in the reveal stage,* $R_x(0, d_0, c) = R_x(1, d_1, c) = \mathsf{accept}$*.*

Definition 17 (ID Resettably Computationally Binding Commitment Scheme). *Consider a sequence of common inputs $\overline{x} = x_1, \dots x_{\mathsf{poly}(\kappa)} \in I$. Randomly select and fix $t = \mathsf{poly}(\kappa)$ random tapes $\omega_1, \dots \omega_t$, for P, resulting in deterministic strategies for a receiver. A receiver using random tape ω and common input x is represented by R_x^ω.*

An instance-dependent commitment scheme $\mathsf{Com}_x = (S_x, R_x)$ is resettably computationally binding on $I \subseteq \{0,1\}^*$ if for every nonuniform PPT S^*, there exists a negligible function ε such that for all $\overline{x} \in I$ the malicious sender S^* succeeds in the following game with probability at most $\varepsilon(|x|)$.

S^* interacts with polynomially many instantiations of receivers R_x^ω where $x \in \{x_1, \dots x_{\mathsf{poly}(\kappa)}\}$ and $\omega \in \{\omega_1, \dots \omega_t\}$ for x and ω of its choice. It finally obtains a commitments for each of these sessions*. Then S^* outputs pairs $(0, d_0)$ and $(1, d_1)$, and a common input x and a commitment c generated in one of these sessions with R_x, and* succeeds *if in the reveal stage,* $R_x(0, d_0, c) = R_x(1, d_1, c) = \mathsf{accept}$*.*

Lemma 4 (Follows from Theorem 1 of [OV08]). *For every language $L \in \mathcal{SZK}$, there exists an instance-dependent commitment scheme*[6] *that is statistically hiding on instances $x \notin L$ and statistically*

[6] It can be observed that we have reversed the binding and hiding properties. Since $\mathcal{SZK}$ is closed under complement [Oka96], this follows immediately.

binding[7] *on instances $x \in L$. Moreover, this instance-dependent commitment scheme is public coin and is constant round.*

Next we use this instance-dependent commitment scheme and construct an instance-dependent commitment scheme which is resettable computationally binding on instances $x \in L$ and statistically hiding on instances $x \notin L$. We do this in a way very similar to the BGGL Transformation (Proposition 3.5 of [BGGL01]). The key idea is that the receiver chooses its messages based on the output of a pseudorandom function applied on the transcript of the protocol, so far.

Lemma 5 (Similar to Proposition 3.1 of [BGGL01]). *Let $L \in \mathcal{SZK}$ and (S_x, R_x) be a public-coin and constant-round instance-dependent commitment scheme that is statistically hiding on instances $x \notin L$ and computationally binding on instances $x \in L$. Also, let $\{f_s : \{0,1\}^* \to \{0,1\}^{|s|}\}$ be a family of pseudorandom functions. Assume, without loss of generality, that on common input x, in each round, the receiver R_x sends a uniformly distributed $|x|$-bit string. Let $R'_x(s)$ be a deterministic receiver program that, on common input $x \in L$, emulates R_x except that it determines the current round message by applying f_s to the transcript so far. Let R'_x be defined so that on common input x and uniformly random tape $s \in \{0,1\}^{|x|}$, it acts as $R'_x(s)$. Then the commitment scheme (S_x, R'_x) is an instance-dependent commitment scheme that is statistically hiding on instances $x \notin L$ and resettable computationally binding on instances $x \in L$.*

Proof. First note that the statistically hiding property is defined for a honest sender, and of course is not affected by the fact that an adversarial sender can reset the receiver, since the hiding property cares about dishonest receivers only.

Now we need to prove that if a commitment (S_x, R_x) is computationally binding when $x \in L$ then (S_x, R'_x) is resettable computationally binding when $x \in L$. The proof follows essentially the principles of [BGGL01] but we give a sketch here for the sake of completeness. Parts of the proof presented here are verbatim from [BGGL01]. We claim that for any polynomial-size adversarial sender S^* that interacts with a resettable receiver R'_x and breaks the binding property in one of the sessions with a probability ϵ, there exists a polynomial-size adversarial sender S^{**} that interacts with a (non-resettable) receiver R_x and breaks the binding property with a probability at least ϵ/k^c, where k is the bound on the number of messages sent by S^* and c is the number of rounds in the original protocol. Note that c is a constant.

Our cheating sender S^{**} proceeds as follows: It uniformly selects $i_1, \dots, i_c \in \{1 \dots k\}$, and invokes (the resetting sender) S^* while emulating an imaginary resettable receiver R_x^F (This verifier R_x^F is same as R'_x, except that it uses a truly random function $F : \{0,1\}^* \to \{0,1\}^{|x|}$, rather that a pseudorandom function f_s, for $|s| = |x|$. Loosely speaking, by the definition of pseudorandom functions, all non-uniform polynomial-size senders must behave in essentially the same way under this replacement.) as follows. If the prefix of the current session transcript is identical to a corresponding prefix of a previous session, then S^{**} 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) S^* sends 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 S^{**} answers according to the following two cases:

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

[7] This implies that the scheme is also computationally binding. We rely on this weaker property only.

Clearly, for any possible choice of integers $i_1, \ldots, i_c$, the distribution of messages seen by S^* when actually interacting with such an imaginary receiver. 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. This is possible because the original receiver is public coin.

Toward the analysis, we call a message sent by S^* *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 novel messages are exactly those that cause S^{**} to pass along (to S^*) a new answer (rather than copying an answer given in a previous session). The *UrMessage* 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 the memory in order to answer the current message. The UrMessage of a novel message is the message itself. Using this terminology, note that the new sender S^{**} succeeds in cheating R_x if the chosen integers $i_1, \ldots, i_c$ equal the indices (within the sequence of all messages sent by S^*) of the c UrMessages that correspond to the c messages sent in a session in which S^* is able to cheat, and therefore provide two commitments, with the imaginary receiver. Since, with probability ϵ such a cheating session exists, S^{**} succeeds provided it has guessed its message indices (i.e., c indices out of k).

Note. Although the commitment schemes described in this section are for committing to bits, we can use standard arguments to extend them to the case of strings of length greater than 1. Further, as we are only concerned with string commitments in this paper, we will abuse notation and use $\mathsf{Com}_{L,x} = (S_x, R_x)$ to mean the corresponding string commitment scheme.

3.3 Instance-Dependent Argument System $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$.

We will need an instance-dependent argument system $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ where PrsSWI_x and VrsSWI_x are the prover and the verifier respectively, with common inputs x and a theorem statement ξ. When[8] x is in the language, we want that $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ be a resettably sound argument of knowledge for $\mathcal{NP}$. In this case, very roughly, $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ has the additional property that the soundness holds even when the prover can reset the verifier. If instead x is not in the language then $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ must be statistical witness indistinguishable. We construct this argument system by instantiating Blum's Hamiltonicity protocol with the constant-round public-coin ID commitment scheme of [OV08]. We make it resettably sound by using a pseudorandom function [BGGL01]. Details follow.

We define an instance-dependent argument system $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$, where PrsSWI_x and VrsSWI_x are the prover and the verifier respectively. In our constructions, $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ will be used as a sub-protocol within an *outer* protocol. We stress that there are two languages involved: the language L, for which the outer protocol is constructed, and the language $\mathcal{L}$ for which the argument system $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is used. It is important to note that the instance on which $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ depends is an instance x of the language L, and not instance ξ of language $\mathcal{L}$. Looking ahead, $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is simply going to be the resettably sound Blum witness indistinguishable argument of knowledge[9] with the following modification: all commitments are replaced by ID commitments corresponding to the language L.

[8] In general, in proof systems when an ID commitment is used, it is parameterized by the theorem statement ξ being proven. In our case the ID commitment is actually parameterized by a different value x. Looking ahead, x would be the theorem statement of an interactive proof system that uses the sub-protocol $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ to prove the $\mathcal{NP}$ statement ξ.

[9] Blum's witness indistinguishable proof system for Hamiltonicity can be made resettably sound by using the BGGL transformation. See [BGGL01].

Definition 18 (Instance-Dependent Argument System). *Given a language L, an* instance-dependent argument system *for a language $\mathcal{L}$ is a family of protocols $\{\langle P_x, V_x\rangle\}_{x\in\{0,1\}^*}$, such that for every $x \in \{0,1\}^*$, we have:*

1. *At the beginning of the protocol, both the prover and the verifier get an instance of language $\mathcal{L}$, ξ, as common input. The prover also gets a witness w for the theorem ξ.*
2. *The prover P_x and verifier V_x run in polynomial time in the size of the input.*
3. *For every $x \in \{0,1\}^*$, if both prover P_x and verifier V_x follow their prescribed strategy, V_x accepts with overwhelming probability if $\xi \in \mathcal{L}$.*

An Instance-Dependent Interactive Argument System $\{\langle P_x, V_x\rangle\}_{x\in\{0,1\}^*}$ is public coin *if for every $x \in \{0,1\}^*$, all messages sent by V_x are independent random coins.*

We want the following instance dependent behaviour from $\{\langle P_x, V_x\rangle\}_{x\in\{0,1\}^*}$:

1. When $x \in L$, we want that $\langle P_x, V_x\rangle$ be a resettably sound argument of knowledge for $\mathcal{NP}$.
2. When $x \notin L$, $\langle P_x, V_x\rangle$ must be statistical witness indistinguishable.

To simplify the notation we use $\langle P_x, V_x\rangle$ to denote an instance-dependent argument system $\{\langle P_x, V_x\rangle\}_{x\in\{0,1\}^*}$. The resettable-completeness, resettable-soundness and statistical witness indistinguishability properties of standard proof systems extend in a natural way to their instance-dependent analogues. Here, as in [BGGL01], we stress that resettable-completeness concerns the fact that a verifier will accept with overwhelming probability instances adaptively selected by the resetting adversary as long as the instances are in the language.

Definition 19 (Instance-Dependent Resettably-Sound Argument, modified from Definition 3.1 from [BGGL01]). *A* resetting attack *of an adversarial prover P_x^* on a* resettable verifier *V_x, is defined by the following two-step random process, indexed by a security parameter κ.*

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

We say that an instance-dependent argument system $\langle P_x, V_x\rangle$ is a resettably sound argument system *for $\mathcal{L}$ on $I \subseteq \{0,1\}^*$ if for all $x \in I$ the following two conditions hold:*

1. Resettable-Completeness: *Consider a resetting attack of polynomial-time adversary, and suppose that in some session, after selecting an incarnation $V_x^{(j)}(\xi)$, the attacker follows the strategy P_x. Then, if $\xi \in \mathcal{L}$ then $V_x^{(j)}(\xi)$ rejects with negligible probability.*
2. Resettable-Soundness: *For every resetting attack of a polynomial-time adversary, the probability that in some session the corresponding $V_x^{(j)}(\xi)$ has accepted and $\xi \notin \mathcal{L}$ is negligible.*

Definition 20 (Instance-Dependent Resettably-Sound Argument of Knowledge, sketch, modified from Definition 3.2 from [BGGL01]). *Let $R_\mathcal{L} \subseteq \{0,1\}^* \times \{0,1\}^*$ be the $\mathcal{NP}$-relation for a $\mathcal{NP}$-language $\mathcal{L} = \{\xi : \exists w \text{ such that } (\xi, w) \in R_\mathcal{L}\}$. We say $\langle P_x, V_x\rangle$ is a resettably sound argument of knowledge for $R_\mathcal{L}$ on $I \subseteq \{0,1\}^*$ if,*

1. *$\langle P_x, V_x\rangle$ is resettably sound argument for $\mathcal{L}$ on I, and,*

2. *for all $x \in I$ and for every polynomial q, there exists a probabilistic expected polynomial-time oracle machine E such that for every resetting attack P_x^* of size $q(\kappa)$, the probability that $E^{P_x^*}(1^\kappa)$ outputs a witness for the input selected in the last session is at most negligibly smaller than the probability that P_x^* convinces V_x in the last session.*

In [BGGL01], the authors present a transformation that achieves the following result.

Lemma 6 (Simplified from Proposition 3.5 of [BGGL01]). *Let $\mathcal{L} \in \mathcal{NP}$ and $R_{\mathcal{L}}$ be a corresponding witness relation. Let $\langle P_x, V_x \rangle$ be a constant-round, public-coin argument of knowledge for $R_{\mathcal{L}}$ on I. Then $\langle P_x, V_x \rangle$ can be transformed into $\langle P'_x, V'_x \rangle$, such that, if $\langle P_x, V_x \rangle$ is witness-indistinguishable then, $\langle P'_x, V'_x \rangle$ is a resettably sound witness-indistinguishable argument of knowledge for $R_{\mathcal{L}}$ on I.*

We note (as observed in [BGGL01] for a similar claim) that if $\langle P_x, V_x \rangle$ is *statistically* witness indistinguishable on I, then so is $\langle P'_x, V'_x \rangle$, as the witness-indistinguishability property of $\langle P_x, V_x \rangle$ refers to all possible adversarial verifiers, including V'_x. We will use V_x^ω to denote the deterministic verifier strategy obtained by fixing V_x's random tape as ω.

Let $\langle P_x, V_x \rangle$ be a constant-round, public-coin argument of knowledge on $I \subseteq \{0,1\}^*$ for $\mathcal{NP}$ relation $R_{\mathcal{L}}$, and let $\langle P'_x, V'_x \rangle$ be the argument system on I obtained as in Lemma 6. Then, we have the following lemma.

Corollary 1 (Follows directly from Proof of Proposition 3.5 of [BGGL01]). *For, any polynomial-size adversarial prover P'_x that convinces V'_x to accept some common input ξ with probability ϵ, there exists a polynomial-size adversarial prover P''_x that convinces a (non-resettable) verifier V_x to accept the same ξ with probability at least ϵ/m^c, where m is a bound on the number of messages sent by the prover P'_x and c is the number of rounds in the original protocol.*

Definition 21 (Statistical Witness Indistinguishability). *Let $\langle P_x, V_x \rangle$ be an instance-dependent argument system for language $\mathcal{L}$ and witness relation $R_{\mathcal{L}}$. Then $\langle P_x, V_x \rangle$ is statistically witness-indistinguishable on $I \subseteq \{0,1\}^*$ if for every non-uniform PPT adversarial verifier V^*, with auxiliary input z, for all $\xi \in \mathcal{L}$ (where $|\xi|$ is polynomial in the security parameter) and witnesses $w_1, w_2 \in R_{\mathcal{L}}(\xi)$, the following ensembles:*

$$\{\mathrm{view}_{V_x^*}(\langle P_x(w_1), V_x^*(z)\rangle(\xi))\}_{x \in I, \xi \in \mathcal{L}, z \in \{0,1\}^*}, \{\mathrm{view}_{V_x^*}(\langle P_x(w_2), V_x^*(z)\rangle(\xi))\}_{x \in I, \xi \in \mathcal{L}, z \in \{0,1\}^*}$$

are statistically indistinguishable.

Construction. We will use Blum's constant-round public-coin argument system for Hamiltonicity where commitments in the first message of Blum's protocol are played using the instance-dependent commitment scheme described in earlier. Since for $x \in L$, $\mathcal{COM}$ is computationally binding this protocol will be an argument of knowledge (which implies soundness). On the other hand, for $x \notin L$ the commitment is statistically hiding and thus our instantiation of Blum's protocol will be statistically witness-indistinguishable. Applying the BGGL [BGGL01] transformation, using Lemma 6 we have that when $x \notin L$, then the transformed protocol will enjoy resettable soundness (i.e., soundness will hold even in case the adversarial prover can reset the verifier) remaining an argument of knowledge. Obviously the transformation has no impact on the witness indistinguishability since it only affect the behavior of the honest verifier. We will denote this transformed instance-dependent protocol by $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$. We stress that this protocol exists for all languages in $\mathcal{SZK}$.

4 Resettable Partially Honest Verifier Statistical Zero Knowledge

We aim at constructing a resettable statistical zero-knowledge proof system. We start by building a simpler protocol which is resettable statistical zero knowledge only against a restricted class of adversarial verifiers. In subsequent sections, we build upon this simpler protocol to achieve our general results. The adversarial verifiers that we consider here are restricted to "act honestly" but only in a limited manner. We call such verifiers *partially honest*. As pointed out in § 3, we use a non-interactive ID commitment scheme. Looking ahead, in our protocol this commitment is used by the verifier to commit to certain messages. A partially honest verifier is required to behave honestly when computing the commitment function, using pure randomness to commit to messages. Besides this it can cheat in any other way.

Definition 22 (Partially Honest Verifier). *An adversarial verifier V^* for* cpHSZK *is said to be a partially honest verifier if the first message of the verifier* $(\alpha, \alpha^0_{1,1}, \alpha^1_{1,1}, \ldots, \alpha^1_{k,k})$ *is such that: there exist unique* $m, \sigma^b_{i,j}$, *and there exist* ρ_0, $\rho^b_{i,j}$ *for* $1 \leq i, j, \leq k$, $b \in \{0,1\}$, *such that,*

1. $\alpha = \mathsf{C}_{L,x}(m, \rho_0)$, *and,*
2. $\alpha^b_{i,j} = \mathsf{C}_{L,x}(\sigma^b_{i,j}; \rho^b_{i,j})$ *for each* $1 \leq i, j \leq k$ *and* $b \in \{0,1\}$, *and,*
3. $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = m$ *for* $1 \leq i, j \leq k$.

except with negligible (in κ) probability.

Consider the protocol cpHSZK in Fig. 3. The first message generated by the honest verifier uses the commitment scheme $\mathcal{COM}$. We call this message the *determining message*. An adversarial verifier V^* is called a partially honest verifier if it generates these commitments "honestly." This requires that: 1) commitments are "well-formed", and 2) commitments generated in the first step have unique openings except with negligible probability. It follows from the description in Section 3.1, that a perfectly binding $\mathcal{COM}$ always has a unique opening. On the other hand a statistically binding $\mathcal{COM}$ has a unique opening (except with negligible probability) given that the coins used in generation of the commitments are chosen uniformly at random. Because of this difference in the primitive used our final constructions based on these primitives will be different.

We proceed as follows:

1. We first show a proof system cpHSZK (see Fig. 3) that is concurrent *partially honest* verifier statistical zero-knowledge (defined below). Informally, a verifier is said to be partially honest if its first message is "honestly" generated. The formal definition appears in Definition 22.
2. Then, we transform cpHSZK into a resettable partially honest verifier statistical zero-knowledge rpHSZK (see Fig. 4).

Concurrent Partially Honest Verifier SZK. We start by informally describing the protocol cpHSZK of Fig. 3. It consists of three phases. The first phase, called the *Determining Message Phase*, consists of the verifier sending a commitment to a string m to the prover. We use the extractable non-interactive ID commitment scheme described informally earlier and formally defined in Section 3.1. The second phase is roughly a PRS preamble [PRS02] and we refer to it as the *PRS Phase*. Note that some commitments are made in the PRS preamble, but we lump these with the commitment to m, in the Determining Message Phase itself. Finally the prover sends to the verifier the value m. This is referred to as the *Final Message*. An adversarial verifier, denoted by V^*, is called a partially honest verifier (defined formally in Definition 22) if it generates the non-interactive ID commitments of the Determining Message Phase "honestly." This requires that these ID commitments are: (1) "well-formed" and (2) have unique[10] openings (except with negligible probability).

We begin by briefly sketching why cpHSZK is a concurrent statistical zero-knowledge proof system for L. Completeness follows from binding property of $\mathcal{COM}$: when $x \in L$, the commitments in the Determining Message Phase are statistically binding with unique openings with overwhelming probability.

[10] It follows from the description in Section 3.1, that a perfectly non-interactive ID commitment $\mathcal{COM}$ always has a unique opening. On the other hand an honest sender statistically non-interactive ID commitment $\mathcal{COM}$, has a unique opening with overwhelming probability, for honest senders only.

Common Input: $x \in L \cap \{0,1\}^n$ and security parameter κ where parameter $k = \omega(\log \kappa)$ and $n = \mathsf{poly}(\kappa)$.
Secret Input to P: Witness w such that $(x, w) \in R_L$ (not needed in case of unbounded prover).

1. **Determining Message** ($V \to P$) V chooses message m randomly from $\{0,1\}^{\ell_0}$, and computes $\alpha = \mathsf{C}_{L,x}(m; \rho_0)$ for some random $\rho_0 \in \{0,1\}^{\ell_1}$. For $1 \leq i \leq k$ and $1 \leq j \leq k$, V randomly chooses $\sigma^0_{i,j}$ and $\sigma^1_{i,j}$ such that $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = m$. For each (i, j, b), where $1 \leq i \leq k$, $1 \leq j \leq k$ and $b \in \{0,1\}$, V randomly chooses $\rho^b_{i,j} \in \{0,1\}^{\ell_1}$ computes the commitment $\alpha^b_{i,j} := \mathsf{C}_{L,x}(\sigma^b_{i,j}; \rho^b_{i,j})$. Finally, V sends all the commitments $\alpha, \alpha^0_{1,1}, \alpha^1_{1,1}, \ldots, \alpha^1_{k,k}$ to the prover.
2. **PRS Phase** ($V \Leftrightarrow P$) For $1 \leq l \leq k$:
   (a) P sends b_l chosen randomly in $\{0,1\}^k$ to V.
   (b) Let b^i_l be the i^{th} bit of b_l. V sends the openings of $\alpha^{b^1_l}_{l,1}, \ldots, \alpha^{b^k_l}_{l,k}$.
   (c) If the opening sent by the verifier is invalid, then P sends ABORT to verifier, and aborts the protocol.
3. **Final Message** ($P \to V$) P runs the extractor associated to the ID commitment of the Determining Message Phase. If the extractor aborts then P aborts, else P sends the output of the extractor m' to V, who accepts if $m' = m$.

Figure 3. Concurrent Partially Honest Verifier Statistical Zero-Knowledge Proof System: cpHSZK.

Thus, the prover can extract the unique message m and make the verifier accept in the Final Message Phase. For soundness, note that when $x \notin L$, the commitments in the first phase are statistically hiding. Thus, m committed to in the Determining Message Phase is information theoretically hidden from a cheating prover (also shares received during the preamble do not give any information), and therefore, it can convince the verifier only with negligible probability.

To argue zero knowledge, we use the rewinding strategy of [PRS02]. Using the PRS rewinding strategy we can construct a simulator that obliviously rewinds the verifier and is guaranteed (except with negligible probability) to obtain the opening m committed to in the Determining Message Phase, before the end of the PRS Phase for every session (except with negligible probability) initiated by the cheating verifier. Once the cpHSZK simulator knows the message m committed in the Determining Message Phase, it can play it back to the verifier in the Final Phase.

Note that to prove zero knowledge, we crucially use the fact that the verifier is partially honest. First, we need that the commitment sent by the verifier are correctly formed. This is to make sure that the commitments are done in accordance with the specifications for the first message of PRS preamble. Secondly, we need that these commitments have unique openings with overwhelming probability. If, for example, verifier's commitment to m in the Determining Message Phase has two openings, then the simulation would fail. Indeed, an unbounded prover unable to decide which is the right opening, would always abort while the simulator would still extract some message from the PRS Phase and send that to the verifier in the Final Phase. The case of an efficient prover instead would result in extracting a message that could depend on the witness used, while the one obtained by the simulator would not depend on the witness, therefore potentially generating a distinguishable deviation in the transcript.

Resettable Partially Honest Verifier SZK. We now exploit a key property of cpHSZK and transform it into a resettable statistical zero-knowledge proof system secure against partially honest adversaries. We note that the final message of cpHSZK depends only on the first message of the verifier. In particular, it depends neither on the random tape of the prover, nor on its witness. Also messages of the prover in the PRS phase are just random strings. Thus, very informally, an adversarial verifier can not obtain any advantage by resetting the prover, as after every reset, the verifier will get the same message back in the final round. This is a crucial fact that allows us to achieve resettability.

The transformed protocol, called rpHSZK (Fig. 4), is the same as cpHSZK, except for one difference: in the PRS Phase, instead of sending random challenges in Step 2(a), the prover uses pseudorandom challenges. The prover chooses a random seed s for selecting a function from a PRF family $\{f_s\}_{s \in \{0,1\}^*}$, and sets ω as the output of $f_s()$ evaluated on the message received during the Determining Message

Common Input: $x \in L \cap \{0,1\}^n, k = \omega(\log \kappa), n = \mathsf{poly}(\kappa)$ for a security parameter κ.
Secret Input to P**:** Witness w such that $(x, w) \in R_L$ (not needed in case of unbounded prover).

1. **Determining Message** Same as in Fig. 3.
2. **PRS Phase** $(V \Leftrightarrow P)$ P chooses a random seed s, and sets $\omega = f_s(x, \alpha, \alpha^0_{1,1}, \ldots, \alpha^1_{k,k})$. Now P divides ω into k blocks of k-bits each, i.e., $\omega = \omega^1 \circ \ldots \circ \omega^k$. For $1 \le l \le k$,
   (a) P sends ω^l to V.
   (b) Same as Fig. 3 Step 2b.
   (c) Same as in Fig. 3 Step 2c.
3. **Final Message** Same as in Fig. 3.

Figure 4. Resettable Statistical Partially Honest Verifier Zero-Knowledge Proof System rpHSZK.

Phase. The prover uses this ω as its random tape for the PRS phase. A modification of the PRS simulation where the simulator uses both pseudorandom and random messages during the preamble, along with other known tricks [CGGM00] allows us to prove that this protocol is a resettable statistical zero-knowledge proof system with respect to partially honest verifiers.

We now provide the details of the proofs discussed above.

Concurrent Statistical Partially Honest Verifier Zero-Knowledge. Consider the protocol cpHSZK (in Fig. 3). To show that the protocol cpHSZK is concurrent partially honest verifier statistical zero-knowledge, we directly use the rewinding strategy of PRS [PRS02]. The PRS simulator $\textsc{sim}$ (Fig. 5) executes an oblivious rewinding strategy in the hope of obtaining two different responses from V^* for the same *slot*. A slot is defined as prover's k-bit challenge, and verifier's response to that challenge, in the same session, and the 'challenge-response' phase of that session consists of the k-slots in Step 2 of protocol cpHSZK. Note that if the simulator is able to obtain two different responses from V^* for the same slot, then there exists (i, j) such that the simulator has obtained both $\sigma^0_{i,j}$ and $\sigma^1_{i,j}$. This allows the simulator to obtain message m committed by V^* in the first round, and can play the final round of the session (when P has to return m) correctly, making V^* accept.

Let V^* be an adversarial verifier, and consider the simulation of V^* via $\textsc{sim}$. Over the probability space defined by the random tapes of simulator $\textsc{sim}$, we define the event $\mathsf{ExtractFail}_{V^*}$ as follows: there exists a session s, such that the simulator $\textsc{sim}$ has finished the challenge-response phase for session s, but has not extracted the message m for that session yet. Note that in this case, $\textsc{sim}$ can not proceed further in the simulation. It is proved in [PRS02] that the probability of this event occurring is negligible.

Lemma 7 ([PRS02]). *Let V^* be an adversarial verifier for* cpHSZK*. Then, the probability of occurrence of* $\mathsf{ExtractFail}_{V^*}$ *is negligible in κ.*

We now show that protocol cpHSZK is statistical concurrent partially honest verifier zero-knowledge.

Lemma 8. *The protocol* cpHSZK *is concurrent partially honest verifier statistical zero-knowledge proof any for language L that admits a statistical non-interactive instance-dependent commitment scheme $\mathcal{COM}$.*

Proof. We first argue completeness and soundness. Let $x \in L$ and first consider the case that $\mathcal{COM}$ is perfectly binding on yes-instances. In this case, P can (possibly inefficiently) extract the message committed by V in the determining message, and send the correct message back to V, which accepts with probability 1. Now consider the case that $\mathcal{COM}$ is statistically binding on the yes-instances. Then, as V constructs all instance-dependent commitments honestly (that is by uniformly choosing the randomness), the determining message would have two openings only with negligible probability. Thus, in this case, V accepts with probability negligibly close to 1.

Now let us consider the soundness property. Let $x \notin L$. In this case, the commitment scheme $\mathcal{COM}$ is statistically hiding. Thus, the instance-dependent commitments sent as the determining message do not reveal any information about the message m committed by V. Now, consider the PRS Phase. In this phase, the prover receives openings of shares of m. However, for each (i, j), it learns either $\sigma^0_{i,j}$ or $\sigma^1_{i,j}$, and never both. As these are random shares, they don't reveal any information about m. Thus, when $x \notin L$, prover's view is statistically independent of the message m, and therefore the probability that it can send the correct message in the final round is negligible.

Now we prove that cpHSZK enjoys concurrent partially honest verifier statistical zero knowledge. We use the simulator SIM mentioned before. First, note that messages sent by SIM in the challenge-response phase are identically distributed to messages sent by the real prover in that phase. Also, as the probability of occurrence of $\mathsf{ExtractFail}_{V^*}$ is negligible, with all but negligible probability, SIM extracts some message m', which is then sent to the verifier. It follows from the definition of partially honest verifier, and the statistical binding property of the instance-dependent commitment scheme $\mathcal{COM}$, that (for every session) m' is identical (except with negligible probability) to the final message played by the real prover.

```
**Input:** $(l, \mathsf{hist}, T)$
**Bottom level** $(l = 1)$:

– Uniformly choose a first stage prover message $\mathsf{p}$, and feed $V^*$ with $(\mathsf{hist}, \mathsf{p})$.
– Store $V^*$'s answer $\mathsf{v}$, in T.
– Output $(\mathsf{p}, \mathsf{v}), T$.

**Recursive Step** $(l > 1)$:

– Set $(\tilde{\mathsf{p}}_1, \tilde{\mathsf{v}}_1, \ldots, \tilde{\mathsf{p}}_{l/2}, \tilde{\mathsf{v}}_{l/2}), T_1 \leftarrow \textsc{sim}(l/2, \mathsf{hist}, T)$.
– Set $(\mathsf{p}_1, \mathsf{v}_1, \ldots, \mathsf{p}_{l/2}, \mathsf{v}_{l/2}), T_2 \leftarrow \textsc{sim}(l/2, \mathsf{hist}, T_1)$.
– Set $(\tilde{\mathsf{p}}_{l/2+1}, \tilde{\mathsf{v}}_{l/2+1}, \ldots, \tilde{\mathsf{p}}_l, \tilde{\mathsf{v}}_l), T_3 \leftarrow \textsc{sim}(l/2, \mathsf{hist}, T_2)$.
– Set $(\mathsf{p}_{l/2+1}, \mathsf{v}_{l/2+1}, \ldots, \mathsf{p}_l, \mathsf{v}_l), T_4 \leftarrow \textsc{sim}(l/2, \mathsf{hist}, T_3)$.
– Output $(\mathsf{p}_1, \mathsf{v}_1, \ldots, \mathsf{p}_l, \mathsf{v}_l), T_4$.
```

Figure 5. The PRS simulator, SIM.

Resettable Partially Honest Verifier Statistical ZK. In this subsection, we convert the concurrent partially honest verifier statistical zero-knowledge protocol cpHSZK to a resettable partially honest verifier statistical zero-knowledge protocol rpHSZK (Fig. 4). The transformation is very similar to the CGGM-transformation ([CGGM00]), where the prover applies a PRF to the determining message of the session, and uses its output as the randomness for the rest of the protocol. If the verifier resets the prover and changes the first (determining) message of the session, prover's subsequent responses also change, as the PRF is now applied to the new determining message. On the other hand, if the verifier resets and plays the same determining message, then all of prover's responses are the same as before. Thus, intuitively, the verifier does not gain any advantage by resetting the prover.

As pointed out in the introduction, the standard techniques proposed for constructing resettable zero-knowledge proof systems are not suited to construct resettable statistical zero-knowledge proof systems. We stress that novelty of our argument here is in cpHSZK, which has certain special properties that allow it to be converted to a resettable statistical zero-knowledge proof systems. In [CGGM00], the CGGM-transformation is applied to obtain resettable *computational* zero-knowledge. On the other hand, because of the special property of cpHSZK, we are able to obtain *statistical* zero-knowledge (using the same CGGM transformation). This is because the messages of the prover depend either only on its randomness, or only on the verifier's messages. Soundness comes from the fact that the prover can successfully generate these messages only if the theorem is true. We stress that our proof differs from the one of [CGGM00] also in the construction of the simulator. Our simulator must use pseudorandom coins unlike [CGGM00] where as they use pure random coins. However a simulator using pseudorandom coins does not gain anything by rewinding (since the same identical messages would be played again). We deal with this problem by having the simulator use pseudorandom coins for messages that will appear in the final transcript while using pure random coins for messages played during a look-ahead procedure, that therefore will not be visible to the unbounded distinguisher.

We begin by informally describing the steps we take in order to prove that rpHSZK is a resettable partially honest verifier statistical zero-knowledge proof system.

1. We first prove that rpHSZK is in fact a concurrent statistical partially honest verifier zero-knowledge protocol. We do this by constructing a modified PRS simulator MSIM (note that because the real transcript of rpHSZK contains pseudorandom challenges from the prover, we can not use the original PRS simulator SIM, as it uses random challenges).
2. Then we convert the resetting partially honest verifier V^* to an intermediate concurrent partially honest verifier W^*. This is similar to the construction of the 'hybrid adversarial verifier' in [CGGM00]. This transformation ensures that the output of V^* (when interacting with rpHSZK prover) is distributed identically to the output of W^* (when interacting with the cpHSZK prover). Thus, the zero-knowledge property will follow from the simulator constructed in Step 1 above.

The modified simulator MSIM (Fig. 6) uses the same rewinding strategy as SIM, except that it uses pseudorandom challenges in the "main thread." Because V^* (and thus W^*) is polynomially bounded, the probability that MSIM gets stuck is only negligibly different from the probability that SIM gets stuck. Further, the challenges generated by MSIM and the real prover (in rpHSZK) are identically distributed in the main thread, and thus the views given in output by V^* corresponding to the main thread execution are statistically close.

We begin by proving that rpHSZK is also concurrently secure. The completeness and soundness follow from the completeness and soundness of cpHSZK. To prove zero-knowledge, we modify the PRS simulator obtaining MSIM (Fig. 6). The simulator MSIM is similar to the PRS simulator, except it uses pseudorandom challenges in the main thread, unlike SIM which uses completely random challenges (both simulators use random challenges in all the look-ahead threads). Let V^* be an adversarial concurrent verifier for rpHSZK, and consider the simulation of V^* via MSIM. Over the probability space defined by the random tapes of simulator MSIM, we define the event $\mathsf{ExtractFail}_{V^*}$ [11] as follows: there exists a session s, such that the simulator MSIM has finished the challenge-response phase for session s, but has not extracted the message m for that session yet. We note that conditioned on $\mathsf{ExtractFail}_{V^*}$ not occurring (and because V^* is partially honest), the distribution of the main thread of MSIM is statistically close to the distribution of the real transcript. This is because the messages in the PRS preamble are chosen from the same distribution by both the real prover and MSIM(in its main thread). Further, as the determining message has a unique opening (with all but negligible probability), and V^* does not cheat in constructing the shares (as it is partially honest), the final message is identical in both the simulation and the real executions. Thus, to complete the proof of zero-knowledge, we need to prove that the probability of occurrance of $\mathsf{ExtractFail}_{V^*}$ is negligible in κ.

Lemma 9. *Let V^* be a partially-honest adversarial verifier for* rpHSZK*. Then, the probability of occurrance of* $\mathsf{ExtractFail}_{V^*}$ *is negligible in κ.*

[11] Note that this event is different from the $\mathsf{ExtractFail}_{V^*}$ event used in Lemma 7, as the underlying probability spaces are different. We use the same name for the two events for notational convenience.

Proof. The proof follows from a standard hybrid argument. We construct hybrids starting with SIM and leading to MSIM in the final hybrid. Let the first hybrid correspond to the interaction between SIM and V^*. We call this $\mathcal{H}_0$. We recall that SIM uses random coins to generate queries in the PRS Phase. Therefore all queries in $\mathcal{H}_0$ are generated using random coins. Consider the hybrid $\mathcal{H}_1$ such that pseudorandom coins are used in the main thread of the first session and random coins for the rest of the sessions. We define a series of hybrids $\mathcal{H}_i$ such that pseudorandom coins are used in the main thread for the first i sessions and random coins are used for the rest of the sessions. If there are N sessions then we will have $N + 1$ hybrids. The only difference between $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ is that pseudorandom coins are used in main thread for the $i + 1^{th}$ session while random coins are used in the $\mathcal{H}_i$. Note that $\mathcal{H}_N$ corresponds to the interaction between MSIM and V^*. If the probability of MSIM getting stuck is non-negligibly different from the probability that SIM gets stuck, then the hybrids $\mathcal{H}_0$ and $\mathcal{H}_N$ will be non-negligibly far apart. Since $N = \mathsf{poly}(\kappa)$, there will exist hybrids $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ which are non-negligibly far apart. In that case we can construct an adversary that can distinguish random bits from pseudorandom bits. This is not possible since V^* is polynomially bounded. Therefore, since the probability that SIM gets stuck is negligible (as in Lemma 7), we have that the probability that MSIM gets stuck is negligible.

INTERMEDIATE VERIFIER W^*. Here we explain the construction of W^* in brief.

Informally speaking W^* acts as a "mediator" that intercepts messages from V^* on one side, and conveys them to the prover on the other side. Verifier W^* must handle messages from V^* to the prover in such a way that it looks like a concurrent adversary. To do this, W^* keeps a record of the sessions opened by the resetting adversary V^*. Every time V^* opens a new session, W^* proceeds as follows: if the session is opened with a new incarnation of the prover, then W^* opens a new session with the prover. However, if the session is a repeat session (i.e., a reset of a session previously opened), then W^* simply plays the messages from the previous invocation of this session. In this case, W^* does not need to communicate with the prover at all.

We describe W^* more formally now. In the following, we will use the notation P^{i,j,Δ_t} to refer to the interaction of V^* with the $(i, j)^{th}$ incarnation of the prover, where V^*'s determining message is Δ_t.

1. *V^* initiates a new session $P^{(i,j,\Delta_t)}$.* In this case W^* initiates a new session $P^{(i,j,\Delta_t)}$ and forwards the message Δ_t as the determining message for the session to the prover.
2. *V^* initiates a repeat session $P^{(i,j,\Delta_t)}$.* In this case W^* must send the message that W^* had sent last time. In order to deal with this W^* retrieves the message that was generated previously from the store and sends it to V^*.
3. *V^* responds to a challenge corresponding to session $P^{(i,j,\Delta_t)}$.* If the response is not well-formed then W^* sends abort to V^*. If the response challenge message (or the final message of prover) has already been sent to W^*, then retrieve that message and send it to V^*. Otherwise, it forwards the message of V^* to the prover and forwards the response of the prover to W^*. It also stores the prover's response (challenge value or the final message of prover), in case it needs to be sent again to V^*.
4. *V^* terminates.* Whenever V^* sends a termination message, W^* also terminates. It outputs the transcript generated by V^* as its output.

Lemma 10. *Views $\langle P(\overline{y}), V^*\rangle(\overline{x})$ and $\langle P(\overline{y}), W^*\rangle(\overline{x})$ are identical.*

Proof. The proof of the the claim follows directly by the construction of W^*.

Lemma 11. *If a family of one-way functions exist, and L admits a statistical non-interactive ID commitment scheme $\mathcal{COM}$ then there exists a resettable partially honest verifier statistical zero-knowledge proof system for L.*

```
Input: $(l, \mathsf{hist}, T, b)$ (Invoked with $b = 1$)
Bottom level ($l = 1$):
  – If $b = 0$ then uniformly choose a first stage prover challenge message $\mathsf{p}$, else generate an honest prover challenge message $\mathsf{p}$ using a pseudo-random function and feed $V^*$ with $(\mathsf{hist}, \mathsf{p})$.
  – Store $V^*$'s answer $\mathsf{v}$, in T.
  – Output $(\mathsf{p}, \mathsf{v}), T$.
Recursive Step ($l > 1$):
  – Set $(\tilde{\mathsf{p}}_1, \tilde{\mathsf{v}}_1, \ldots, \tilde{\mathsf{p}}_{l/2}, \tilde{\mathsf{v}}_{l/2}), T_1 \leftarrow \textsc{msim}(l/2, \mathsf{hist}, T, 0)$.
  – Set $(\mathsf{p}_1, \mathsf{v}_1, \ldots, \mathsf{p}_{l/2}, \mathsf{v}_{l/2}), T_2 \leftarrow \textsc{msim}(l/2, \mathsf{hist}, T_1, b)$.
  – Set $(\tilde{\mathsf{p}}_{l/2+1}, \tilde{\mathsf{v}}_{l/2+1}, \ldots, \tilde{\mathsf{p}}_l, \tilde{\mathsf{v}}_l), T_3 \leftarrow \textsc{msim}(l/2, \mathsf{hist}, T_2, 0)$.
  – Set $(\mathsf{p}_{l/2+1}, \mathsf{v}_{l/2+1}, \ldots, \mathsf{p}_l, \mathsf{v}_l), T_4 \leftarrow \textsc{msim}(l/2, \mathsf{hist}, T_3, b)$.
  – Output $(\mathsf{p}_1, \mathsf{v}_1, \ldots, \mathsf{p}_l, \mathsf{v}_l), T_4$.
```

Figure 6. Modified PRS simulator, MSIM.

Proof. Completeness of rpHSZK follows directly from the completeness of cpHSZK. Soundness of rpHSZK follows directly from the soundness of the protocol cpHSZK. This is because the soundness for rpHSZK is guaranteed against all cheating senders.

We first construct the simulator $\mathcal{S}_{\mathsf{rpHSZK}}$.

1. $\mathcal{S}_{\mathsf{rpHSZK}}$ invokes both MSIM and V^*, and uses W^*. Simulator $\mathcal{S}_{\mathsf{rpHSZK}}$ relays the messages as follows. All messages generated by V^* for the prover are sent to W^*. All messages generated by W^* for the verifier are sent to V^*. All messages generated by W^* for the prover are sent to MSIM. All messages generated by MSIM are sent to W^*.
2. Finally, $\mathcal{S}_{\mathsf{rpHSZK}}$ outputs the transcript generated by W^*, corresponding to the main thread execution of MSIM.

We show that the output of the verifier in the real interaction and the output of the simulator $\mathcal{S}_{\mathsf{rpHSZK}}$ are statistically close via the following hybrids.

Hybrid $\mathcal{H}_0$. In this experiment, an honest prover strategy is executed with V^*. In particular, a random seed s is chosen, and the PRF $f_s(\cdot)$ is applied to the determining message of each session. Thereafter, this output is used for the challenge-response phase. In the end, if the final message of the session is reached without aborting, the committed message m is extracted from the determining message, and sent to V^*. The output of the experiment is the output of V^*. This is clearly the real interaction.

Hybrid $\mathcal{H}_1$. Same as Hybrid $\mathcal{H}_0$, except the intermediate verifier W^* is constructed, and used to mediate the interaction between V^* and the honest prover strategy. For each session, the final message is still extracted from the determining message, and sent to W^*, which sends it to V^*. If for any session, the determining message does not have a unique opening (for the case of statistically-binding $\mathcal{COM}$), that session is aborted. By Lemma 10, hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$ are statistically close.

Hybrid $\mathcal{H}_2$. In this hybrid, the honest prover strategy is replaced with MSIM. For each session, it is still checked if the determining message has a unique opening or not. If not, then that session is aborted. Else, as the final message, the message extracted by MSIM is sent for that session.

The challenges sent by MSIM to V^* are identically distributed to the challenges sent by the honest prover strategy. By Lemma 9 and the fact that V^* is a partially honest adversarial verifier, it follows that hybrids $\mathcal{H}_2$ and $\mathcal{H}_1$ are statistically close.

Hybrid $\mathcal{H}_3$. This is the same as previous hybrid, except the determining message is not checked for a unique opening. By the fact that V^* is partially honest verifier, and that $\mathcal{COM}$ is statistically binding, it follows that Hybrids $\mathcal{H}_2$ and $\mathcal{H}_3$ are statistically close. This is the output of the simulator $\mathcal{S}_{\mathsf{rpHSZK}}$.

5 Resettable Statistical ZK from Perfect Non-Interactive ID Commitments

Sub-protocol: $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is a resettably sound argument of knowledge when $x \in L$ and a statistical witness indistinguishable argument when $x \notin L$.
Common Input: $x \in L \cap \{0,1\}^n, k = \omega(\log \kappa), n = \mathsf{poly}(\kappa)$ for a security parameter κ.
Secret Input to P^a: Witness w such that $(x,w) \in R_L$ (not needed in case of unbounded prover).

1. **Determining Message:** Same as in Fig. 4.
2. **Proof of Consistency:** $(V \Leftrightarrow P)$ V and P run $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$, where V plays the role of PrsSWI_x, and P plays the role of VrsSWI_x. V proves to P knowledge of $m, \sigma^b_{i,j}, \rho_0, \rho^b_{i,j}$ for $1 \le i,j, \le k$, $b \in \{0,1\}$, such that:
   (a) $\alpha = \mathsf{C}_{L,x}(m, \rho_0)$, and,
   (b) $\alpha^b_{i,j} = \mathsf{C}_{L,x}(\sigma^b_{i,j}; \rho^b_{i,j})$ for each $1 \le i,j \le k$ and $b \in \{0,1\}$, and,
   (c) $\sigma^0_{i,j} \oplus \sigma^1_{i,j} = m$ for $1 \le i,j \le k$.
3. **PRS Phase:** Same as in Fig. 4.
4. **Final Message:** Same as in Fig. 4.

[a] P aborts the protocol in case any proof from the verifiers does not `accept` or some message is not well formed. Notice that P uses two different seeds for the PRF f (one in Step 2 and the other one in Step 3).

Figure 7. Resettable Statistical Zero-Knowledge from Perfect Non-Interactive ID Commitments: rSZK.

In this section we consider languages that admit perfect non-interactive ID commitments and we construct a resettable statistical ZK proof system which is secure against *all* malicious verifiers.

Let L be a language that admits a perfect non-interactive ID commitment scheme, and let $\mathcal{COM}$ be the corresponding commitment function. We extend the proof system rpHSZK for L to handle arbitrary malicious verifiers. The main idea is to enforce "partially honest behavior" on the malicious verifier. We recall that the partially honest restriction on a verifier required that the verifier uses $\mathcal{COM}$ to generate commitments honestly. More specifically, we required that these commitments have unique openings and are correctly constructed. A fully malicious verifier however can deviate and compute commitments that do not have the prescribed form. Therefore, the only concern we have is to make sure that commitments are correctly generated. We enforce this by modifying rpHSZK and adding an extra step to it. This step requires that the verifier proves to the prover that shares constructed in Step 1 (as part of the Determining Message) are correct. If this proof is accepted then the prover can conclude that the first message of the verifier is indeed honestly generated and the malicious verifier is forced into following the desired partially honest behavior. In our protocol the verifier uses an instance-dependent argument system $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ such that: when $x \in L$, $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is a resettably sound argument[12] of knowledge, while when $x \notin L$, $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is statistically witness indistinguishable. Since the protocol is resettably sound the malicious verifier can not go ahead with incorrect commitments even when it can reset the prover. For the protocol see Fig. 7.

We need to prove that rSZK (in Fig. 7) is a resettable statistical zero-knowledge proof system. More formally,

Theorem 1. *If a family of one-way functions exists, and L admits a perfect non-interactive ID commitment scheme then there exists a resettable statistical ZK proof system for L.*

[12] Note that in $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$, we used the interactive ID commitment scheme Com_x (defined in Section 3.3). We could have also used the perfect non-interactive ID commitment $\mathcal{COM}$ in the case of co-$\mathcal{RSR}$ and $\mathcal{DDH}$.

We begin by first showing completeness and soundness of rSZK (Fig. 7). We then argue that the protocol is resettable statistical ZK.

Completeness. If $x \in L$, then the commitment scheme $\mathcal{COM}$ is perfectly-binding. Thus, the prover can always correctly extract the message m committed by the verifier in the determining message, and send it in the final message. If the commitment scheme permits, this extraction can be done in polynomial time.

Soundness. Now let us consider the soundness property. Let $x \notin L$. In this case, the commitment scheme $\mathcal{COM}$ is statistically hiding. It follows from the statistical hiding of the commitment scheme, the statistical WI property of the proof of consistency in Step 2, and the fact that only one of $\sigma^0_{i,j}$ and $\sigma^1_{i,j}$ is ever revealed in the PRS phase, that the view of prover is independent of the message m committed in the determining message. Thus, the probability that a malicious prover sends the correct m in the final step is negligible.

Resettable Statistical Zero Knowledge. In order to show that our protocol is resettable statistical zero-knowledge, by first converting the malicious verifier V^* (adversarial verifier for protocol rSZK) and converts it to a verifier X^* that is resetting partially honest verifier (except with negligible probability) for protocol rSZK. We first describe this step. Then we obtain two results about X^* in Lemma 12 and Lemma 13. After this has been accomplished we argue in Lemma 14 that we can indeed construct a simulator that can simulate the view of any malicious V^*.

Construction of a resetting partially honest verifier X^*. Informally, X^* acts as a "mediator," that intercepts the messages generated by the V^* (adversarial verifier for protocol rSZK) and converts it to a verifier X^* that is resetting partially honest verifier (except with negligible probability) for protocol rpHSZK. X^* acts as the prover P_{rSZK} of rSZK for V^* and interacts with the prover P_{rpHSZK} of protocol rpHSZK. Informally to achieve this conversion, X^* internally deals with the messages corresponding to the proof of consistency phase following the honest prover algorithm P_{rSZK} for these stages. After this part of the interaction with V^* have been successfully completed X^* initiates a session with the P_{rpHSZK}. Note that V^* can interact with prover $P^{(i,j)}_{\mathsf{rSZK}}$ of its choice and can reset it as it desires. We describe how X^* handles the messages with respect to one particular $P^{(i,j)}_{\mathsf{rSZK}}$. The messages corresponding to every $P^{(i,j)}_{\mathsf{rSZK}}$ are handled in the same way.

1. Corresponding to each $P^{(i,j)}_{\mathsf{rSZK}}$, X^* chooses a fixed random tape.
2. It follows the honest prover algorithm (for protocol rSZK) and responds to messages in the proof of consistency phase.
3. If in a session the proof of consistency phase is successfully completed then X^* initiates a new session with $P^{(i,j)}_{\mathsf{rpHSZK}}$ and sends the determining message received earlier from V^* as the determining message for the session. All messages between $P^{(i,j)}_{\mathsf{rpHSZK}}$ and V^* after the determining message is sent to $P^{(i,j)}_{\mathsf{rpHSZK}}$ are relayed as such.
4. In the end, output the output generated by V^*.

Lemma 12. *The outputs $\langle P_{\mathsf{rSZK}}(\overline{y}), V^*\rangle(\overline{x})$ and $\langle P_{\mathsf{rpHSZK}}(\overline{y}), X^*\rangle(\overline{x})$ are identically distributed.*

Proof. The proof of the claim follows from the construction of X^*.

Lemma 13. *For every resetting verifier V^* of protocol* rSZK*(Fig. 7) let ϵ be the probability that X^* is not a resetting* partially honest *verifier for Protocol* rpHSZK*(Fig. 4). Then ϵ is negligible in the security parameter κ.*

Proof. We note when $x \in L$, $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is resettably sound. In every session V^* proves that its first message is well formed. X^* only forwards this message outside when this proof is accepting. It follows from the resettable soundness of the proof that these proofs are correct.

We first note that as $x \in L$, all instance-dependent commitments sent by the verifier have unique openings. Thus, if X^* is not a resetting partially honest verifier for Protocol rpHSZK, then it must be the case that shares $\sigma_{i,j}^b$s are not consistent with the committed message m. We will show that this violates the resettable-soundness of $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$.

We construct a malicious prover PrsSWI_x^* for $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$. The prover PrsSWI_x^* uses V^* to interact with an external verifier VrsSWI_x, while also internally running $\mathcal{S}_{\mathsf{rpHSZK}}$. Prover PrsSWI_x handles V^*'s messages as follows:

1. For all sessions, all messages from Step 2 of the protocol consisting of the Proof of Consistency are forwarded to the external verifier VrsSWI_x. Messages sent back by VrsSWI_x are sent to V^*.
2. For all sessions, messages in the PRS Phase (Step 3) are sent to $\mathcal{S}_{\mathsf{rpHSZK}}$, and its responses conveyed back to V^*.
3. For all sessions, for the final message, PrsSWI_x^* sends the message extracted by $\mathcal{S}_{\mathsf{rpHSZK}}$ for that session. If no such message exists, PrsSWI_x^* aborts.

We first note that by Lemma 9, the probability that PrsSWI_x^* can not send a final message for some session is negligible in κ. Thus, V^*'s view in all sessions where it committed to the correct shares, is statistically close in the above interaction and the real interaction (with real prover of Protocol rpHSZK). Let s be the first session in which V^* cheats; that is, uses incorrect shares of the determining message, and yet, the proof of consistency is accepted. Then, in session s, V^*'s view is again statistically close to the real interaction up to the point it receives the final message. Note that as the shares are incorrect, the message extracted by $\mathcal{S}_{\mathsf{rpHSZK}}$ might be different from message committed to the determining message phase. Thus, V^* would detect this deviation, and its execution from that point forward might be very different from the real interaction. However, the proof of consistency for session s has already been forwarded to the external verifier, which accepted the proof. Thus, this truncated prover was successful in proving a false theorem, which contradicts the resettable-soundness of $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$, and thus ϵ must be negligible.

Construction of $\mathcal{S}_{\mathsf{rSZK}}$. Given an adversarial verifier V^*, we show a simulator $\mathcal{S}_{\mathsf{rSZK}}$ such that views $\langle P_{\mathsf{rSZK}}(\overline{y}), V^* \rangle(\overline{x})$ and $\mathcal{S}_{\mathsf{rSZK}}^{V^*}(\overline{x})$ are statistically close. We now construct the simulator $\mathcal{S}_{\mathsf{rSZK}}$ as follows.

1. Given oracle access to V^* that is an adversarial resetting verifier for protocol rSZK we construct an adversarial partially honest verifier X^* for protocol rpHSZK.
2. The output of $\mathcal{S}_{\mathsf{rSZK}}$ is the output generated by X^* when interacting with $\mathcal{S}_{\mathsf{rpHSZK}}$.

Next we give the proof that the distribution of the transcripts generated by the simulator $\mathcal{S}_{\mathsf{rSZK}}$ is statistically close to the distribution on transcripts generated in interaction with an honest resettable prover.

Lemma 14. *Random variables $\langle P_{\mathsf{rSZK}}(\overline{y}), V^* \rangle(\overline{x})$ and $\mathcal{S}_{\mathsf{rSZK}}^{V^*}(\overline{x})$ are statistically close.*

Proof. Consider the following hybrid experiments.

Hybrid $\mathcal{H}_0$. This experiment corresponds to the interaction between an honest prover P of rSZK which has access to the witness $\overline{y}$ and V^*. This clearly corresponds to the real prover interaction $\langle P_{\mathsf{rSZK}}(\overline{y}), V^* \rangle(\overline{x})$.

Hybrid $\mathcal{H}_1$. This is the same as the previous hybrid, except that we use the given resetting verifier V^* to construct a resetting partially honest verifier X^*. In this hybrid X^* now interacts with the prover of rpHSZK that has access to to the witness $\overline{y}$. More formally, this hybrid represents $\langle P_{\mathsf{rpHSZK}}(\overline{y}), X^*\rangle(\overline{x})$. By lemma 12, this hybrid is identical to hybrid $\mathcal{H}_0$.

Hybrid $\mathcal{H}_2$. This is the same as the previous hybrid, except that now instead of letting X^* interact with a prover with a witness, we let X^* of $\mathcal{H}_1$ interact with $\mathcal{S}_{\mathsf{rpHSZK}}$. By lemma 13, X^* is a partially honest resetting verifier except with negligible probability. Given that X^* is a partially honest resetting verifier by Theorem 11, the output of $\mathcal{S}_{\mathsf{rpHSZK}}^{X^*}$ is statistical close to the output generated by an honest prover while interacting with X^*. So, hybrids $\mathcal{H}_1$ and $\mathcal{H}_2$ are statistically close. Note that this hybrid corresponds exactly to $\mathcal{S}_{\mathsf{rSZK}}^{V^*}$.
Hence, Hybrids $\mathcal{H}_0$ and $\mathcal{H}_2$ generate transcripts that are statistically close.

Implications. It turns out that perfect non-interactive ID commitment schemes [IOS97,TW87,SCPY94] are actually known to exist for all languages in co-$\mathcal{RSR}$. co-$\mathcal{RSR}$ is the class of languages such that the complement of each of these languages is random self-reducible. We refer the reader to Section 3.1 for formal definitions of these classes and how such ID commitment schemes can be constructed. Finally we note that our protocol rSZK, works for all languages in co-$\mathcal{RSR}$.

Corollary 2. *If a family of one-way functions exists and $L \in$ co-$\mathcal{RSR}$ then there exists a resettable statistical ZK proof system for L.*

Proof. It follows directly from Lemma 1 and Theorem 1.

Similarly, we have the following corollary for languages that admit hash proof systems.

Corollary 3. *If $L \in \mathcal{SZK}$ admits a hash proof system then there exists an efficient-prover resettable statistical ZK proof system for L.*

Proof. The corollary follows directly from the definition of hash proof systems, and Theorem 1.

6 Resettable Statistical ZK for all Languages in $\mathcal{SZK}$

In this section we construct the general proof system which is actually resettable statistical zero knowledge for all languages that have a statistical zero knowledge proof. Just like in previous section, we start with a resettable partially honest verifier statistical ZK proof system. But we look at all languages in $\mathcal{SZK}$ and construct a resettable statistical ZK proof system which is secure against *all* malicious verifiers.

Let L be a language that admits an honest sender statistical non-interactive ID commitment scheme $\mathcal{COM}$. We extend the proof system rpHSZK for L to handle arbitrary malicious verifiers in our protocol in Fig. 8. The main idea is to enforce "partially honest behavior" on the malicious verifier. We recall that the partially honest restriction on a verifier required that the verifier uses $\mathcal{COM}$ to generate commitments honestly. More specifically, we required that these commitments are correctly constructed and have unique openings. The first requirement can be handled in a way just like in previous section, i.e. by having the verifier prove to the prover that shares constructed in Step 1 (as part of the Determining Message) are correct. We use the ID argument system $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x\rangle$ to achieve this. The problem of uniqueness is more tricky, and we discuss that next.

The difficulty lies in the fact that the statistical non-interactive ID commitment scheme for all languages in $\mathcal{SZK}$ [CCKV08], only works with respect to honest senders. Indeed, if the sender chooses the randomness for the commitment uniformly, then, with overwhelming probability, the computed commitment has a unique valid opening. However a malicious sender could focus on a set of negligible size, $\mathcal{B}$, of *bad* random strings r, such that $\mathsf{C}_{L,x}(m;r)$ does not have a unique opening. If a malicious verifier (that plays as sender of this commitment scheme) is able to pick random strings from $\mathcal{B}$, then the real interaction and the simulation can be easily distinguished. In the real protocol, the prover tries to invert the commitment α, finds it does not have a unique opening, and aborts. In the simulation, the simulator extracts some message m from the PRS phase, and sends m as the final message. As the simulator is polynomially bounded, it *can not* detect if the commitment has a unique opening or not. To use this commitment scheme, we must somehow ensure that the verifier does not use bad randomness for its commitments. We do this by adding a special coin-flipping subprotocol at the beginning of the protocol. However, because of reset attacks, the coin-flipping subprotocol introduces several technical problems.

We begin by describing our coin-flipping protocol.The coin-flipping protocol requires a commitment scheme such that computational binding holds against resetting senders when $x \in L$ and statistical hiding holds when $x \notin L$. We use the interactive ID commitment scheme $\mathsf{Com}_{L,x} = (S_x, R_x)$. The coin flipping proceeds as follows: first the verifier commits to a random string r_1. Let the transcript of the interactive commitment be c. Then prover applies the sub-exponentially hard PRF $f_s(c)$ and obtains r_2 that is sent to the verifier. The randomness that the verifier will use for the non-interactive ID commitment is $r_1 \oplus r_2$. For technical reasons, the verifier also needs to prove knowledge of r_1 after it has committed to r_1. We use the interactive ID argument system $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ for this. In the security proof, we will use some of the techniques introduced in [BGGL01] to show an adversary that extracts the randomness used by the adversarial verifier in order to break the binding of the commitment scheme.

Next we highlight the reasons behind the use of sub-exponentially hard pseudorandom (PRF) functions for our construction. Let α be the statistical non-interactive ID commitment of some message m sent by the verifier. There are two ways in which α might not have a unique opening. In the first case, a malicious V^*, after looking at prover's response r_2, might use an opening of c such that $r_1 \oplus r_2 \in \mathcal{B}$. This however would violate the computational binding of the interactive ID commitment scheme secure against resetting senders used in the coin flipping, thus this event occurs with negligible probability. The second case is more subtle. It might be possible that performing reset attacks, the verifier can study the behavior of the PRF, and then can be able to succeed in obtaining that $r_1 \oplus r_2 \in \mathcal{B}$ with non-negligible probability (even though the polynomial-time V^* does not know the two openings). In this case, we can not construct a polynomial-time adversary that breaks f_s, as we can not efficiently decide if $r \in \mathcal{B}$. This is where we need the sub-exponential hardness of the one-way function and in turn of the PRF. As $|\mathcal{B}|$ is only 2^ℓ while the size of the set of all random strings is 2^L, where $l = \mathrm{o}(L)$, we can give the entire set $\mathcal{B}$ as input to the sub-exponential size circuit that aims at breaking the PRF. The circuit can now check if the string r is a bad string or not, by searching through its input. Notice that one can give as input to the circuit the whole $\mathcal{B}$ for each of the polynomial number of statements (since for each x there can be a different $\mathcal{B}$) on which the reset attack is applied (see Definition 3). This sub-exponential size circuit has still size $\mathrm{o}(L)$ and breaks the PRF which contradicts the sub-exponential hardness of the PRF.

Completeness follows from the fact that when $x \in L$, with overwhelming probability, the commitment α in the determining message will have a unique opening. Thus, the prover will be able to extract the committed message and send it as the final message to the verifier, that will accept.

Statistical resettable zero knowledge property of our protocol also follows the same argument. Indeed, when $x \in L$ even a resetting verifier can not cheat during the proofs in Steps 1(c) and 3. Moreover, the above discussion about the security of the coin-flipping protocol implies that a resetting adversarial verifier is forced into following partially honest behavior when computing the non-interactive ID commitments.

Finally, we look at soundness. Note that when $x \notin L$ non-interactive ID commitments are statistically hiding and the protocol $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is statistical WI in Steps 1(c) and 3. Also note that the fact that only a single share is revealed in the PRS phase. From this it follows that the prover's view when verifier commits message m is statistically close to its view when verifier commits to m', where $m \neq m'$. Thus, the probability that it replies with the correct final message is negligible. Now we are ready to give the detailed protocol and security proof.

Sub-protocols: $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$ is a resettably sound argument of knowledge when $x \in L$ and a statistical witness indistinguishable argument when $x \notin L$; (S_x, R_x) is an interactive ID commitment scheme that is resettably computational binding when $x \in L$ and statistically hiding when $x \notin L$.
Common Input: $x \in L \cap \{0,1\}^n, k = \omega(\log \kappa), n = \mathsf{poly}(\kappa)$ for a security parameter κ.
Secret Input to P^a: Witness w such that $(x, w) \in R_L$ (not needed in case of unbounded prover).

1. **Coin Flipping**
   (a) $(V \Leftrightarrow P)$: V picks a random value $r_1 \in \{0,1\}^\ell$, where $\ell = (2k^2+1)\ell_1$. Now P and V run the commitment scheme (S_x, R_x) where V plays the sender S_x and commits the string r_1. Let r_1' be the random tape used by V for S_x, and let c be the transcript of this sub-protocol.
   (b) **PoK of a Committed String** $(V \Leftrightarrow P)$: Now, P and V run $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$, where P plays the verifier VrsSWI_x, and V plays the prover PrsSWI_x. V proves to P that there exist r_1, r_1' such that when S_x is used the commit r_1 using randomness r_1', then the transcript is consistent with c.
   (c) $(P \to V)$: Let $\tau|x$ be the concatenation of the transcript so far and x. P picks a random $s_1 \in \{0,1\}^{\mathsf{poly}(\kappa)}$, sets $r_2 = f_{s_1}(\tau|x)$ (note that $|r_2| = (2k^2+1)\ell_1$), and sends it to V.
2. **Determining Message**
   (a) $(V \to P)$: V sets $\rho = r_1 \oplus r_2$. Divide ρ into blocks of length ℓ_1, i.e., $\rho = \rho_0 \circ \rho_{1,1}^0 \circ \rho_{1,1}^1 \circ \ldots \circ \rho_{k,k}^0 \circ \rho_{k,k}^1$.
   (b) V chooses message m randomly from $\{0,1\}^{\ell_0}$, and computes $\alpha = \mathsf{C}_{L,x}(m; \rho_0)$.
   (c) For $1 \le i \le k$ and $1 \le j \le k$, V randomly chooses $\sigma_{i,j}^0$ and $\sigma_{i,j}^1$ such that $\sigma_{i,j}^0 \oplus \sigma_{i,j}^1 = m$. For each (i,j,b), where $1 \le i \le k$, $1 \le j \le k$ and $b \in \{0,1\}$, V computes the commitment $\alpha_{i,j}^b := \mathsf{C}_{L,x}(\sigma_{i,j}^b; \rho_{i,j}^b)$.
   Finally, V sends all the commitments $\alpha, \alpha_{1,1}^0, \alpha_{1,1}^1, \ldots, \alpha_{k,k}^1$ to P.
3. **Proof of Consistency** $(V \Leftrightarrow P)$: Now, P and V run $\langle \mathsf{PrsSWI}_x, \mathsf{VrsSWI}_x \rangle$, where P plays the verifier VrsSWI_x, and V plays the prover PrsSWI_x. V proves to P the knowledge of $r_1, r_1', m, \sigma_{i,j}^b$ for $1 \le i, j, \le k$, $b \in \{0,1\}$, where $r_1 \oplus r_2 = \rho_0 \circ \rho_{1,1}^0 \circ \ldots \circ \rho_{k,k}^1$, such that,
   (a) When S_x is used to commit r_1 with randomness r_1', the transcript is consistent with c, and,
   (b) $\alpha = \mathsf{C}_{L,x}(m, \rho_0)$, and,
   (c) $\alpha_{i,j}^b = \mathsf{C}_{L,x}(\sigma_{i,j}^b; \rho_{i,j}^b)$ for each $1 \le i, j \le k$ and $b \in \{0,1\}$, and,
   (d) $\sigma_{i,j}^0 \oplus \sigma_{i,j}^1 = m$ for $1 \le i, j \le k$.
4. **PRS Phase**: Same as in Fig. 4.
5. **Final Message**: Same as in Fig. 4.

[a] P aborts the protocol in case any proof from the verifier does not `accept` or some message is not well formed. Notice that P uses four different seeds for the PRF f (one in Step 1.b, one in Step 1.c, one in Step 3 and one in Step 4).

Figure 8. Resettable Statistical Zero-Knowledge Proof System rSZK' for $\mathcal{SZK}$.

Theorem 2. *If a family of sub-exponentially hard one-way functions exists, and L admits a honest-sender statistical non-interactive ID commitment scheme then there exists a resettable statistical ZK proof system for L.*

As every language in $\mathcal{SZK}$ has an honest-sender statistical non-interactive ID commitment scheme (see Lemma 3, Section 3.1), the previous theorem gives us the following result.

Corollary 4. *If a family of sub-exponentially hard one-way functions exists, then $\mathcal{SZK} = r\mathcal{SZK}$.*

We prove this by actually using a cheating resetting verifier that breaks the statistical ZK property of rSZK' to construct another resetting partially honest verifier that breaks the statistical ZK property of rpHSZK. We begin by constructing a resetting partially honest verifier X^* for rpHSZK.

Construction of a resetting partially honest verifier X^*. The construction of the intermediate verifier X^* is very similar to the construction of the intermediate verifier of Section 5. Here, X^* must handle the additional messages for the coin-flipping stage. These messages are handled using the honest prover strategy. All other messages are handled as described in Section 5.

Lemma 15. *$\langle P_{\mathsf{rSZK}'}(\overline{y}), V^*\rangle(\overline{x})$ and $\langle P_{\mathsf{rpHSZK}}(\overline{y}), X^*\rangle(\overline{x})$ are identically distributed.*

Proof. The proof of the claim follows from the construction of X^*.

Now we prove that the above X^* is a resetting partially honest verifier for rpHSZK. Then we can use our machinery developed in the previous section for the simulation.

Lemma 16. *For every resetting verifier V^* of rSZK'(Fig. 8) X^* is a resetting* partially honest *verifier for rpHSZK(Fig. 4) except with negligible probability.*

Proof. Before starting the proof, we make the following observation: it is sufficient to show that in the *last* session, verifier V^* is not partially honest with only negligible probability. This is because given a V^* that cheats with non-negligible probability in any session, we can construct V^{**} that cheats in the last session with non-negligible probability: V^{**} runs V^*, and after V^* terminates, it randomly selects a completed session s, and replays it with the external prover. Thus, if V^* cheats with non-negligible probability in some session, then V^{**} cheats in the last session with non-negligible probability. As a further simplification, we only prove that the commitment α sent in Step 2(b) has a unique opening with overwhelming probability. The same analysis applies to all other instance-dependent commitments in the session.

Lemma 17. *Let α be the commitment played in Step 2(b) of the last session of rSZK'. Let ϵ be the probability that α has two distinct openings and P does not abort the protocol before Step 4. Then ϵ is negligible in κ.*

Proof. (Sketch) As in [BGGL01], we use the extractor for the Blum argument of knowledge twice to extract the witnesses used by V^* in the two proofs in its last session. We have the added complication of having to extract twice: once from the proof of commitment in Step 1(c), and once from the proof of consistency in Step 3. We have to be careful that we extract both the proofs from the same session of the protocol, otherwise we might extract witnesses for different theorems, and we won't be able to reach a contradiction. However, these issues can be handled using techniques similar to [BGGL01]. We first extract from the proof of commitment in Step 1(c). We run V^*, and follow honest prover's strategy for the messages before Step 4. We also run MSIM and let it handle all PRS preamble messages. For the final message of a session, we use the message extracted by MSIM for that session. By Lemma 9, and resettable-soundness of proof of consistency in Step 3., it follows that with overwhelming probability, the correct message for the session is extracted.

Let c_{last} be the commitment in Step 1(b) of the last session. Note that we can not use Blum's extractor on this last session only. This is because the verifier might have played the same session before. In this case, as the verifier is sending the same determining message, it knows the messages from the prover. Thus, if the extractor tries to extract by rewinding and changing its challenge, the verifier will detect the deviation, and might abort. Thus, we must rewind V^* to the first time it plays the determining message in the last session. That is, we rewind V^* to session s, where s is the first session such that the commitment in Step 1(b) of session is c_{last}. Now we use Blum's extractor to find the witness (r_1, r_1'). Next, we rewind and extract from the proof of consistency (Step 3) for the same session

s using Blum's extraction. Let the opening of c_{last} extracted from the proof of consistency be (t_1, t_1'). The probability of extracting both these values, ϵ', is polynomial in ϵ. At least one of the following two cases occurs with probability at least $\epsilon'/2$.

Case 1. The two openings are different, that is, $r_1 \neq t_1$. In this case we break the binding of the commitment scheme (S_x, R_x) with probability polynomial in ϵ'. Thus, ϵ must be negligible in the security parameter.

Case 2. The openings are same, i.e., $r_1 = t_1$. In this case, we construct a circuit $\mathcal{A}$ that distinguishes the PRF family $\{ f_s \}$ from a random function. The non-uniform circuit[13] $\mathcal{A}$ gets as input the sets $\mathcal{B}_x$ (for all polynomial instances x on which is based the experiment) of bad strings of $\mathsf{C}_{L,x}$. Adversary $\mathcal{A}$ interacts with V^* playing the role of the honest prover, with the following modifications:

1. For all sessions, all messages of Step 1(c) are forwarded to the BGGL extractor for the resettably-sound argument of knowledge.
2. For all sessions, all messages of Step 4 (PRS Preamble) are forwarded to the simulator MSIM.
3. For the final session, when V^* sends commitment c in Step 1(b), adversary $\mathcal{A}$ forwards this to the external challenger, and gets r_2 in return, which is either uniformly distributed, or the output of the PRF. Adversary $\mathcal{A}$ sends r_2 to V^* in Step 1(d).

Let (r_1, r_1') be the witness extracted from the argument of knowledge above and let x be the instance of this final session. Adversary $\mathcal{A}$ checks if $r_1 \oplus r_2 \in \mathcal{B}_x$. If so, it guesses that the challenger is a PRF, else it guesses the challenger is a random function. Thus, by the security of the PRF family, it follows that ϵ is negligible.

Finally, it follows from the proof of Lemma 14 using Lemma 15 and Lemma 16 instead of Lemma 12 and Lemma 13, respectively, rSZK' is a resettable statistical zero-knowledge proof system.

7 Applications

In this section, we highlight the applicability of our techniques, and construct a simple two-round resettable statistical witness-indistinguishable argument for languages that have efficiently extractable perfectly binding instance-dependent commitment schemes. As discussed before, this class contains, in particular, all languages that admit hash proof systems. We note that all results in this section hold in the stronger model of statistical zero-knowledge where the verifier is computationally unbounded.

Informally, the two-round WI argument consists of the verifier committing to a randomly chosen message m using the instance-dependent commitment scheme for that language. The prover, using the witness and the efficient extractor, extracts a message m' from the commitment and sends it to the verifier. The verifier accepts if $m = m'$. Intuitively, as long as verifier's commitment is well-formed, this protocol is a perfect WI, as irrespective of the witness and randomness, the prover always extracts the same message (in fact, prover's strategy is deterministic). Thus, the only complication is to ensure that verifier's commitment is well-formed in a round efficient manner. We enforce this by making the verifier provide a non-interactive WI proof of "well-formedness" in the first round. Details follow.

As an ingredient, we will need non-interactive WI (NIWI) proofs. The following discussion on non-interactive proofs is taken from [GOS06b].

[13] For simplicity of the exposition and without loss of generality we now describe $\mathcal{A}$ as an ITM.

Preliminaries. Let R be an efficiently computable binary relation. For pairs $(x, w) \in R$ we call x the statement and w the witness. Let L be the language consisting of statements in R.

A non-interactive proof system [BFM88] for a relation R consists of a common reference string generation algorithm K, a prover P and a verifier V. We require that they all be probabilistic polynomial time algorithms, *i.e.*, we are looking at *efficient prover* proofs. The common reference string generation algorithm produces a common reference string σ of length $\Omega(k)$. The prover takes as input (σ, x, w) and produces a proof π. The verifier takes as input (σ, x, π) and outputs 1 if the proof is acceptable and 0 if rejecting the proof. We call (K, P, V) a non-interactive proof system for R if it has the completeness and soundness properties described below.

Completeness. A proof system is complete if an honest prover with a valid witness can convince an honest verifier. For all adversaries $\mathcal{A}$ we have

$$\Pr\left[\sigma \leftarrow K(1^k); (x, w) \leftarrow \mathcal{A}(\sigma); \pi \leftarrow P(\sigma, x, w) : V(\sigma, x, \pi) = 1 \text{ if } (x, w) \in R\right] = 1.$$

Computational soundness. A proof system is sound if it is infeasible to convince an honest verifier when the statement is false. For all polynomial size families $\{x_k\}$ of statements $x_k \notin L$ and all non-uniform polynomial-time adversaries $\mathcal{A}$ running on auxiliary input z, we have

$$\Pr\left[\sigma \leftarrow K(1^k); \pi \leftarrow \mathcal{A}(\sigma, x_k, z) : V(\sigma, x_k, \pi) = 1\right] < \nu(\lambda)$$

where $\nu(\cdot)$ is a negligible function.

Witness-indistinguishability and non-interactive zap. We call (P, V) a non-interactive zap for R if (P, V) is a non-interactive proof (with trivial key generation $K(1^k) = 1^k$) with witness-indistinguishability.

Witness-indistinguishability means that proof does not reveal which witness the prover used. For all non-uniform polynomial-time interactive adversaries $\mathcal{A}$ running on auxiliary input z, we have that

$$\Pr\left[(x, w_0, w_1) \leftarrow \mathcal{A}(1^k, z); \pi \leftarrow P(1^k, x, w_0) : \mathcal{A}(\pi, z) = 1 \text{ and } (x, w_0), (x, w_1) \in R\right]$$
$$\approx \Pr\left[(x, w_0, w_1) \leftarrow \mathcal{A}(1^k, z); \pi \leftarrow P(1^k, x, w_1) : \mathcal{A}(\pi, z) = 1 \text{ and } (x, w_0), (x, w_1) \in R\right].$$

A construction of a NIWI proof for $\mathcal{NP}$ based on number-theoretic assumptions is shown in [GOS06b].

Our Protocol. Our protocol is presented in Figure 9.

Common Input: Instance x of language L.
Prover's Input: Witness w.

1. $V \to P$ Verifier chooses a random message m, and sends a commitment string $c = C_{L,x}(m; r)$ to Prover, along with NIWI proof π of the following statement: there exists a random string r and message m, such that $c = C_{L,x}(m; r)$.
2. $P \to V$ Prover checks if the NIWI verifier will accept the proof π. If so, using witness w, Prover extracts a message m' from c and sends it to Verifier.
3. Finally, Verifier outputs `accept` iff $m = m'$.

Figure 9. Resettable Statistical 2-round WI for Language L that admits a Perfect Instance-Dependent Commitment Scheme $\mathcal{COM}$.

The completeness of the protocol is straightforward. Soundness and witness-indistinguishability is shown in the following claims.

Lemma 18. *Let κ be the security parameter. For every adversarial probabilistic polynomial time prover strategy P^*, and for every $x \notin L$, the probability that P^* makes the Verifier in Figure 9 `accept` is negligible in κ.*

Proof. Let $x \notin L$ and fix an adversarial strategy P^*. Over the random coins of the verifier, let the probability that P^* convinces honest Verifier be ϵ. Denote this event by $\mathcal{E}$. For the sake of contradiction, lets assume that ϵ is non-negligible, or in other words ϵ grows as at least an inverse polynomial, on an infinite number of input lengths. Now, there exists a first round commitment string c such that the probability that P^* convinces the verifier conditioned on the first round commitment is c is at least ϵ. Fix this commitment string c. We want to consider the probability that P^* convinces the verifier given the commitment string was c, and given that the message committed by the (honest) verifier in the first round was m. Denote this probability by $\Pr[\,\mathcal{E} \mid c, m\,]$. Consider the following subset of messages,

$$S = \{\, m : \Pr[\,\mathcal{E} \mid c, m\,] \geq \epsilon/2 \,\}.$$

It follows from an averaging argument to see that $\Pr[\,m \in S\,] \geq \epsilon/2$. Fix some $m_0 \in S$. Consider the following subset of S,

$$S_{m_0} = \{\, m \in S : \Pr[\; P^* \text{ sends } m \text{ to Verifier } \mid c, m_0\,] \geq \epsilon/4 \,\}$$

Thus, set S_{m_0} is the set of messages in S that P^* sends to Verifier with high probability (greater than $\epsilon/4$), when the Verifier commits to message m_0(all conditioned on the commitment string being c). We argue S_{m_0} has small size. Indeed, as P^* returns m_0 with probability at least $\epsilon/2$, we have,

$$|S_{m_0}| \cdot \epsilon/4 \leq 1 - \epsilon/2.$$

Thus, $|S_{m_0}| \leq 4/\epsilon$. By the assumption that ϵ is inverse-polynomial for infinitely many input lengths, the size of S_{m_0} is bounded by a polynomial (for those lengths). However, by the fact that $\Pr[\,m \in S\,] \geq \epsilon/2$ and that m is chosen uniformly at random, the size of S is exponential in κ. Thus, the set difference of these two sets is non-empty. Let m_1 be a message in S but not in S_{m_0}. We now construct a non-uniform adversary $\mathcal{A}$ that breaks the WI property of the WI system.

The non-uniform adversary $\mathcal{A}$ gets (m_0, r_0, m_1, r_1) as advice, where $C_{L,x}(m_0; r_0) = C_{L,x}(m_1; r_1) = c$. Now $\mathcal{A}$ outputs $(c, (m_0, r_0), (m_1, r_1))$, and gets a proof NIWI π, which it sends to P^*. If P^*outputs m_1, adversary $\mathcal{A}$ outputs 1, otherwise it outputs 0.

Note that when π is computed using witness (m_0, r_0), P^* outputs m_1 with probability at most $\epsilon/4$, while if π is computed using witness (m_1, r_1), P^* outputs m_1 with probability at least $\epsilon/2$. Thus, the difference in the probability of $\mathcal{A}$ in outputting 1 in the two cases has a polynomial gap, which contradicts the WI property. □

Next, we prove witness-indistinguishability. Let V^* be a malicious verifier. Let x be an instance of L, and let w_0 and w_1 be two witnesses of membership of x. Let the random variable $\mathsf{msg}_{w_b}^{V^*}(x)$ denote the final message sent by honest prover to V^*, when the honest prover uses witness w_b, and the common input is x.

Lemma 19. *Let x be an instance of L, and let w_0, w_1 be two witnesses of membership for $x \in L$. Then for every PPT verifier V^*, the random variables $\mathsf{msg}_{w_0}^{V^*}(x)$ and $\mathsf{msg}_{w_1}^{V^*}(x)$ are statistically close.*

Proof. Note that as $x \in L$, the commitment scheme $\mathcal{COM}$ is perfectly binding. Thus, if the commitment scheme sent by V^* in the first round is well formed, then both witnesses w_0 and w_1 extract the same unique message, and $\mathsf{msg}_{w_0}^{V^*}(x)$ and $\mathsf{msg}_{w_1}^{V^*}(x)$ are identically distributed. It follows from the soundness of the NIWI proof system that with probability negligibly close to 1, V^*'s commitment is well formed. Thus, $\mathsf{msg}_{w_0}^{V^*}(x)$ and $\mathsf{msg}_{w_1}^{V^*}(x)$ are statistically close. □

Combining the above lemmas, we have the following theorem for this section.

Theorem 3. *Assume Non-Interactive Witness Indistinguishable proofs exist. Let L be a language that admits non-interactive perfectly-binding instance-dependent commitment schemes. Then, there exists a two-round statistical witness-indistinguishable protocol for membership in L.*

Finally, we make the following concluding remarks.

Remark 1. FOUR-ROUND STATISTICAL ZK. We note that our protocol in Figure 7 provides a four-round statistical zero-knowledge proof system for all languages that admit perfect non-interactive instance-dependent commitment schemes, when instead of using the k-round PRS preamble we use a 1-round PRS stage. Furthermore we need to give the proof of consistency in parallel with the rest of the messages of the protocol. We stress that this result was known before, see for example [GMW91,GMR89][IS91][CP94][CDM00]. Nevertheless, our techniques serves as another way to achieve the same result.

Remark 2. We note that our techniques imply that $\mathcal{SZK} = \mathrm{c}\mathcal{SZK}$, unconditionally. The protocol in Fig. 8, without the prover having to use the pseudo-random function, is a concurrent statistical zero-knowledge proof system for all languages in $\mathcal{SZK}$. We stress that one-way functions in our protocol are required just to immunize the protocol from reset attacks and are not needed for $\mathrm{c}\mathcal{SZK}$. The claim about $\mathcal{SZK} = \mathrm{c}\mathcal{SZK}$ unconditionally was already conjectured by Ong and Vadhan in [OV08] and explicitly stated by Pass, Tseng and Venkitasubramaniam in [PTV08].

8 Acknowledgments

We thank Giuseppe Persiano for suggesting the open problem of achieving resettable statistical zero knowledge. Moreover we thank Omkant Pandey for several discussions on our techniques.

This research is supported in part by NSF grants 0830803, 09165174, 106527 and 1118126, US-Israel BSF grant 2008411, grants from OKAWA Foundation, IBM, Lockheed-Martin Corporation and the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. The work of the third author has also been supported in part by the European Commission through the FP7 programme under contract 216676 ECRYPT II.

References

[AFK89] Martin Abadi, Joan Feigenbaum, and Joe Kilian. On hiding information form an oracle. *J. Comput. Syst. Sci.*, 39(1):21–50, 1989.

[AH87] William Aiello and Johan Håstad. Perfect zero-knowledge languages can be recognized in two rounds. In *FOCS*, pages 439–448, 1987.

[APV05] Joël Alwen, Giuseppe Persiano, and Ivan Visconti. Impossibility and feasibility results for zero knowledge wi th public keys. In *Advances in Cryptology – Crypto '05*, volume 3621 of *Lecture Notes in Computer Science*, pages 135–151. Springer Verlag, 2005.

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *42nd Symposium on Foundations of Computer Science, (FOCS '01)*, pages 106–115, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 2001. IEEE Computer Society Press.

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, 1988.

[BFGM01] Mihir Bellare, Marc Fischlin, Shafi Goldwasser, and Silvio Micali. Identification protocols secure against reset attacks. In *EUROCRYPT*, pages 495–511, 2001.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112, 1988.

[BGGL01] Boaz Barak, Oded Goldreich, Shafi Goldwasser, and Yehuda Lindell. Resettably-sound zero-knowledge and its applications. In *FOCS*, pages 116–125, 2001. Full version available at: http://eprint.iacr.org/2001/063.

[BHZ87] Ravi Boppana, Johan Håstad, and Stathis Zachos. Does co-NP have short interactive proofs? *Inf. Process. Lett.*, 25(2):127–132, 1987.

[BJY97] Mihir Bellare, Markus Jakobsson, and Moti Yung. Round-optimal zero-knowledge arguments based on any one-way function. In W. Fumy, editor, *Advances in Cryptology – Eurocrypt '97*, volume 1223 of *Lecture Notes in Computer Science*, pages 280–305. Springer-Verlag, 1997.

[BLV03] Boaz Barak, Yehuda Lindell, and Salil Vadhan. Lower bounds for non-black-box zero knowledge. In *FOCS 2003*, pages 384–393, 2003.

[BMO90] Mihir Bellare, Silvio Micali, and Rafail Ostrovsky. The (true) complexity of statistical zero knowledge. In *STOC*, pages 494–502, 1990.

[CCKV08] André Chailloux, Dragos Florin Ciocan, Iordanis Kerenidis, and Salil P. Vadhan. Interactive and noninteractive zero knowledge are equivalent in the help model. In *TCC*, pages 501–534, 2008.

[CDM00] Ronald Cramer, Ivan Damgård, and Philip D. MacKenzie. Efficient zero-knowledge proofs of knowledge without intractability assumptions. In *Public Key Cryptography*, pages 354–373, 2000.

[CEMT09] James Cook, Omid Etesami, Rachel Miller, and Luca Trevisan. Goldreich's one-way function candidate and myopic backtracking algorithms. In *TCC*, pages 521–538, 2009.

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *STOC*, pages 235–244, 2000.

[CHK10] Ronald Cramer, Dennis Hofheinz, and Eike Kiltz. A twist on the naor-yung paradigm and its application to efficient cca-secure encryption from hard search problems. In *TCC*, pages 146–164, 2010.

[CKPR02] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires (almost) logarithmically many rounds. *SIAM J. Comput.*, 32(1):1–47, 2002.

[CP94] Giovanni Di Crescenzo and Giuseppe Persiano. Round-optimal perfect zero-knowledge proofs. *Inf. Process. Lett.*, 50(2):93–99, 1994.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT*, pages 45–64, 2002.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-Malleable Cryptography. *SIAM J. on Computing*, 30(2):391–437, 2000.

[DFN06] Ivan Damgård, Nelly Fazio, and Antonio Nicolosi. Non-interactive zero-knowledge from homomorphic encryption. In *3rd Theory of Cryptography Conference (TCC '06)*, volume 3876 of *Lecture Notes in Computer Science*, pages 41–59. Springer-Verlag, 2006.

[DGS09] Yi Deng, Vipul Goyal, and Amit Sahai. Resolving the simultaneous resettability conjecture and a new non-black-box simulation strategy. In *FOCS*, 2009.

[DL07] Yi Deng and Dongdai Lin. Instance-dependent verifiable random functions and their application to simultaneous resettability. In *EUROCRYPT*, pages 148–168, 2007.

[DN07] Cynthia Dwork and Moni Naor. Zaps and their applications. *SIAM J. Comput.*, 36(6):1513–1543, 2007.

[DNS98] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. In *STOC*, pages 409–418, 1998.

[DPV04] Giovanni Di Crescenzo, Giuseppe Persiano, and Ivan Visconti. Constant-round resettable zero knowledge with concurrent soundness in the bare public-key model. In *Advances in Cryptology – Crypto '04*, volume 3152 of *Lecture Notes in Computer Science*, pages 237–253. Springer-Verlag, 2004.

[For87] Lance Fortnow. The complexity of perfect zero-knowledge. In *19th ACM Symposium on Theory of Computing (STOC '87)*, pages 204–209, 1987.

[GK96] Oded Goldreich and Ariel Kahan. How to construct constant-round zero-knowledge proof systems for np. *J. Cryptology*, 9(3):167–190, 1996.

[GMOS07] Vipul Goyal, Ryan Moriarty, Rafail Ostrovsky, and Amit Sahai. Concurrent statistical zero-knowledge arguments for np from one way functions. In *ASIACRYPT*, pages 444–459, 2007.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems. *SIAM J. on Computing*, 18(6):186–208, 1989.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game - a completeness theorem for protocols with honest majority. In *Proceedings of the 19th ACM Symposium on Theory of Computing (STOC '87)*, pages 218–229, 1987.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in np have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[GOS06a] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for nizk. In *CRYPTO*, pages 97–111, 2006.

[GOS06b] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Non-interactive zaps and new techniques for nizk. In *CRYPTO*, pages 97–111, 2006.

[GS09] Vipul Goyal and Amit Sahai. Resettably secure computation. In *EUROCRYPT*, pages 54–71, 2009.

[GSV98] Oded Goldreich, Amit Sahai, and Salil P. Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In *STOC*, pages 399–408, 1998.

[HM96] Shai Halevi and Silvio Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *Advances in Cryptology - Crypto '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 201–215. Springer-Verlag, 1996.

[IOS97] Toshiya Itoh, Yuji Ohta, and Hiroki Shizuya. A language-dependent cryptographic primitive. *J. Cryptology*, 10(1):37–50, 1997.

[IS91] Toshiya Itoh and Kouichi Sakurai. On the complexity of constant round zkip of possession of knowledge. In *ASIACRYPT*, pages 331–345, 1991.

[KMV07] Bruce M. Kapron, Lior Malka, and Srinivasan Venkatesh. A characterization of non-interactive instance-dependent commitment-schemes (nic). In *ICALP*, pages 328–339, 2007.

[Lin03] Yehuda Lindell. Bounded-concurrent secure two-party computation without setup assumptions. In *35th ACM Symposium on Theory of Computing (STOC '03)*, pages 683–692. ACM, 2003.

[MOSV06] Daniele Micciancio, Shien Jin Ong, Amit Sahai, and Salil P. Vadhan. Concurrent zero knowledge without complexity assumptions. In *TCC*, pages 1–20, 2006.

[MP06] Silvio Micali and Rafael Pass. Local zero knowledge. In *STOC*, pages 306–315, 2006.

[MR01a] Silvio Micali and Leonid Reyzin. Min-round resettable zero-knowledge in the public-key model. In *Advances in Cryptology – Eurocrypt '01*, volume 2045 of *Lecture Notes in Computer Science*, pages 373–393. Springer-Verlag, 2001.

[MR01b] Silvio Micali and Leonid Reyzin. Soundness in the public-key model. In *Advances in Cryptology – Crypto '01*, volume 2139 of *Lecture Notes in Computer Science*, pages 542–565. Springer-Verlag, 2001.

[MY08] Daniele Micciancio and Scott Yilek. The round-complexity of black-box zero-knowledge: A combinatorial characterization. In *TCC*, pages 535–552, 2008.

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM Symposium on Theory of Computing (STOC '90)*, pages 427–437, 1990.

[Oka96] Tatsuaki Okamoto. On relationships between statistical zero-knowledge proofs. In *28th ACM Symposium on Theory of Computing (STOC '96)*, pages 649–658. ACM, 1996.

[Ost91] Rafail Ostrovsky. One-way functions, hard on average problems, and statistical zero-knowledge proofs. In *Structure in Complexity Theory Conference*, pages 133–138, 1991.

[OV08] Shien Jin Ong and Salil P. Vadhan. An equivalence between zero knowledge and commitments. In *TCC*, pages 482–500, 2008.

[PRS02] Manoj Prabhakaran, Alon Rosen, and Amit Sahai. Concurrent zero knowledge with logarithmic round-complexity. In *FOCS*, pages 366–375, 2002.

[PTV08] Rafael Pass, Wei-Lung Dustin Tseng, and Muthuramakrishnan Venkitasubramaniam. Concurrent zero knowledge: Simplifications and generalizations. 2008. `http://hdl.handle.net/1813/10772`.

[PW10] Rafael Pass and Hoeteck Wee. Constant-round non-malleable commitments from sub-exponential one-way functions. In *EUROCRYPT*, pages 638–655, 2010.

[SCPY94] Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, and Moti Yung. On monotone formula closure of szk. In *FOCS*, pages 454–465, 1994.

[SV97] Amit Sahai and Salil P. Vadhan. A complete promise problem for statistical zero-knowledge. In *FOCS*, pages 448–457, 1997.

[SV03] Amit Sahai and Salil P. Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003.

[TW87] Martin Tompa and Heather Woll. Random self-reducibility and zero knowledge interactive proofs of possession of information. In *FOCS*, pages 472–482, 1987.

[Vad99] Salil Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, MIT, 1999.

[Wee10] Hoeteck Wee. Black-box, round-efficient secure computation via non-malleability amplification. In *FOCS*, 2010.

[YZ07] Moti Yung and Yunlei Zhao. Generic and practical resettable zero-knowledge in the bare public-key model. In *EUROCRYPT*, pages 129–147, 2007.

[ZDLZ03] Yunlei Zhao, Xiaotie Deng, Chan H. Lee, and Hong Zhu. Resettable zero-knowledge in the weak public-key model. In *Advances in Cryptology – Eurocrypt '03*, volume 2045 of *Lecture Notes in Computer Science*, pages 123–139. Springer-Verlag, 2003.

A Further Discussion on Statistical ZK, Resettable ZK and $\mathcal{SZK}$.

Here we give an additional discussion about statistical zero knowledge, resettable zero knowledge and the complexity class $\mathcal{SZK}$.

Perfect and statistical ZK. One of the most important properties inferred by the ZK property is that of forward security, i.e., what has been proved in a ZK proof will be secure for ever. However, while today we can safely assume a bound on the maximal computing power of an adversary, we can not realistically predict its future capabilities. Therefore, protocols with *computational* forward security might be broken when the used security parameters for the used complexity assumption will turn out to be insufficient. It is therefore important to strengthen the ZK property so that the prover does not say anything other than the mere truthfulness of the proven theorem, regardless of the computing power of the distinguisher that in the future will analyze the transcript of the conversation. This notion has been formalized under the two variations of statistical and perfect ZK, where the information encoded in the transcript, beyond the mere truthfulness of the theorem, is essentially null. In this paper we will concentrate on statistical ZK, and we will also consider the complexity class $\mathcal{SZK}$.

Resettable ZK. Motivated by considerations regarding smart cards, the notion of *resettable ZK* (rZK, for short) was introduced in [CGGM00]. A rZK proof system remains "secure" even in case the verifier is able to tamper with the prover and to reset it in the middle of a proof to any previous state, then asking different questions. More specifically, the adversarial verifier can ask the prover to run multiple times the protocol, for several statements, forcing him to reuse randomness, and forcing him also to use different witnesses. Resettable ZK is a strictly stronger primitive than concurrent zero knowledge. The original result by Canetti, Goldreich, Goldwasser and Silvio Micali [CGGM00] showed how to achieve resettable zero-knowledge for any $\mathcal{NP}$ language. Dwork and Naor in [DN07] showed that their ZAP can be turned into a resettably sound resettable witness indistinguishable proof system. Very recently, Deng, Goyal and Sahai in [DGS09] solved the very challenging simultaneous resettability conjecture (i.e., achieving a resettably sound resettable zero-knowledge argument for $\mathcal{NP}$) extending the non-black-box technique of Barak [Bar01], thus making a very important step for understanding the real power of reset attacks. However, all these results only achieve "computational" resettable zero knowledge, i.e., the output of the simulator is distinguishable by an unbounded distinguisher.

Previous work on resettable zero knowledge and $\mathcal{SZK}$. In [CGGM00] Canetti et al. showed how to achieve resettable zero-knowledge proofs for all $\mathcal{NP}$. Then, in [BGGL01], resettably-sound zero-knowledge has been achieved using the non-black-box techniques of Barak [Bar01]. More variations have been considered in [BFGM01,BLV03,GS09]. Further results addressing round complexity issues with some set-up assumptions have been achieved in [MR01a] [MR01b,ZDLZ03,DPV04,APV05,YZ07,DL07]. Very recently, Deng et al. in [DGS09] solved the very challenging simultaneous resettability conjecture (i.e., achieving a resettably sound resettable zero-knowledge argument for $\mathcal{NP}$) using new non-black-box techniques, thus making a very important step for understanding the real power of reset attacks. However, all the previous results that achieve resettable zero-knowledge only obtain "computational" zero knowledge.

The study of the complexity of statistical zero-knowledge started with the works of Fortnow and the one of Aiello and Håstad [For87,AH87] that showed respectively that $\mathcal{SZK} \subset \text{co}\mathcal{AM}$ and that $\mathcal{SZK} \subset \mathcal{AM}$. Boolean closure properties were showed in [SCPY94]. Then Okamoto in [Oka96], proved that $\mathcal{SZK}$ has a public-coin proof system, and proved the closure of the class under complementation. In [SV97], Sahai and Vadhan presented a complete promise problem for $\mathcal{SZK}$, the class of languages possessing statistical zero-knowledge proofs against an honest verifier. This was achieved by showing that all languages in $\mathcal{SZK}$ reduce to the language Statistical Difference, and showing a statistical zero-knowledge proof system for it. In [GSV98] Goldreich et al. showed how to transform any proof system that is statistical zero knowledge against an honest verifier into one that is statistical zero knowledge against all verifiers. Combining [GSV98] with [SV97] one obtains a complete promise problem for $\mathcal{SZK}$, including also adversarial verifiers.

Concurrent statistical zero knowledge has been achieved for several interesting languages in [MOSV06], where Micciancio et al. gave also some efficient-prover constructions for some non-trivial $\mathcal{NP}$ languages. The analysis given in [MOSV06] considers a non-interactive instance-dependent commitment. However [MOSV06], left open the problem of establishing whether c$\mathcal{SZK} = \mathcal{SZK}$.

Ong and Vadhan, conjectured in [OV08] that their interactive instance-dependent commitment scheme for any language in $\mathcal{SZK}$, combined with other techniques introduced in [MOSV06] could show that c$\mathcal{SZK} = \mathcal{SZK}$ unconditionally. The claim about $\mathcal{SZK} =$ c$\mathcal{SZK}$ unconditionally was then explicitly stated by Pass, Tseng and Venkitasubramaniam in [PTV08].

As we have already discussed, reset attacks are much harder to defeat when statistical zero knowledge is desired. All previous techniques turn out to be insufficient and the feasibility of resettable statistical zero-knowledge for a non-trivial language was an interesting open problem. Our work therefore clearly gives a solid contribution to the understanding of the complexity of these notions and opens interesting directions for further research.