T-79.511 Special Course on Cryptology / Zero Knowledge: Resettable Zero Knowledge

Johan Wallén

November 14, 2001

1 Introduction

In this survey, we discuss the concept of *resettable zero knowledge* (RZK) introduced by Canetti, Goldreich, Goldwasser and Micali in [1]. A RZK protocol is essentially a zero knowledge protocol that remains zero knowledge even if the adversary can interact with the prover many times, each time resetting the prover to its initial state and force it to use the same internal random string. Observe that this scenario is very realistic. Consider for example a prover implemented as a simple device such as a smart card without a power supply or writable non-volatile memory.

The resettable model is the strongest notation of zero knowledge put forward to date: a RZK protocol remains secure, even if an attacker is allowed to interact with polynomially many independent clones of the prover in an interleaved manner, and is able to reset each of the clones to its initial state (including its randomness) a polynomial number of times. Thus, RZK in particular implies *concurrent zero knowledge* [7].

Not surprisingly, none of the protocols proposed before [1] remains zero knowledge in the resettable model. As an example, consider the standard zero knowledge proof for GRAPH 3-COLORING (see e.g. [3]). In this protocol, the prover first commits to a random relabeling of the coloring of the graph. Then the verifier picks a random edge (u, v). Finally, the verifier reveals the (relabeled) colors of the nodes u, v. If the colors are different, the verifier accepts. Otherwise, it rejects. In the resettable model, this protocol is obviously insecure. When the prover is reset to its initial state, it is forced to use the same randomization and thus the same relabeling. Thus, the verifier can easily recover the complete coloring by asking the prover to reveal the color of the nodes one by one (and resetting the prover between the iterations).

It turns out that the protocol above easily can be transformed to a *resettable witness indistinguishable* (RWI) protocol for graph coloring. The basic idea is to have the verifier commit to the edges it is going to ask the prover to reveal before the actual protocol starts. The prover then uses this commitment and a pseudorandom function to derive random relabelings for the coloring. The rest of the protocol proceeds in the natural way.

This survey is organized as follows. First, we give a precise definition of the basic resettable model, and show that it remains secure even if the adversary is allowed to interleave polynomially many independent interactions with the prover in addition to being able to reset the prover. Then we present a general transformation for turning a certain class of protocols into resettable ones. This transformation is used to construct constant round resettable witness indistinguishable (RWI) proofs for **NP** and (nonconstant round) resettable zero knowledge (RZK) proofs for **NP**. Finally, we discuss a certain *public key model*, where the verifiers are assume to have public keys. This model allows much more efficient protocols including constant round RZK protocols for **NP**.

Throughout this survey, we assume some familiarity with basic zero knowledge. We refer to the other surveys in this seminar for this background. To improve readability, definitions for some more special concepts are included in an appendix.

2 Definitional issues

We will define two resettable zero knowledge models, one interleaved and one sequential, and show that they are equivalent. This allows us to analyze protocols in the much simpler sequential model.

2.1 Conventions

Throughout this survey, we will use the following conventions. Given a prover P, a common input x, the auxiliary input y for the prover (typically, y will be an **NP** witness for some **NP** language), and random string ω, we let $P_{x,y,\omega}$ denote the deterministic prover resulting when we fix these values.

By convention, each message send between interactive machines contains the complete transcript of the interaction so far.

2.2 The resettable model

Definition 1 (RZK [1]). Let n be a security parameter. A prover strategy P is said to be *resettable zero knowledge (RZK)* for the language L, if there for every probabilistic polynomial time adversary V^* as below, exists a probabilistic polynomial time simulator S^* so that the following two probability ensembles indexed by a sequence of common inputs $x = x_1, \ldots, x_{\text{poly}(n)}$ and a sequence of prover's auxiliary inputs $y = y_1, \ldots, y_{\text{poly}(n)}$, are computationally indistinguishable.

Distribution 1 is defined by the following random process depending on P and V^*.

1. Select and fix $t = \text{poly}(n)$ uniformly distributed random strings w_j for P giving the deterministic strategies $P^{i,j} = P_{x_i,y_i,\omega_j}$ for all $i, j \in \{1, \ldots, t\}$. Each $P^{i,j}$ is called an *incarnation* of P.

2. Machine V^* is allowed to initiate polynomially many interactions with the $P^{i,j}$s (Note that it can instantiate each $P^{i,j}$ more than once).

   - In the general (interleaved) model, V^* is allowed to send messages to all $P^{i,j}$s and obtain the responses.

   - In the sequential (non-interleaved) model, V^* must proceed in rounds. In each round, V^* selects one of the $P^{i,j}$s and completes a full interaction with it.

3. Once V^* decides it is done, it produces an output based on its view of these interactions. This output is denoted by $\langle P(y), V^* \rangle(x)$.

Distribution 2 is simply the output of the simulator: $S^*(x)$.

If there exists a universal probabilistic oracle machine S, such that S^* can be implemented by giving S oracle access to V^*, we say that P is *black-box RZK*.

Recall that a witness indistinguishable proof system is roughly speaking a proof system, where the the verifier cannot tell the difference between two valid witnesses.

Definition 2 (RWI [1]). A prover strategy P is said to be *resettable witness indistinguishable (RWI)*, if every two distribution ensembles of type 1 in Definition 1, where each distribution is indexed by a sequence of common inputs x, depending on two different auxiliary prover inputs $\text{aux}^1(x)$ and $\text{aux}^2(x)$, are computationally indistinguishable. That is, $\{\langle P(\text{aux}^1(x), V^*)\rangle\}_x$ and $\{\langle P(\text{aux}^2(x), V^*)\rangle\}_x$ are computationally indistinguishable.

Observe that the (interleaved) resettable model is inherently concurrent. Thus, resettable zero knowledge provides a solution to the important *concurrent zero knowledge* problem (see e.g. [7]).

2.3 The sequential model is equivalent to the interleaved model

Intuitively, the sequential model is equivalent to the interleaved model, since an adversary in the interleaved model can be simulated by an adversary in the sequential model simply by rewinding the prover in a suitable manner.

Theorem 3 ([1]). Let P be a prover strategy. For every probabilistic polynomial-time verifier V^* for the interleaved model, there exist a probabilistic polynomial-time verifier W^* for the sequential model, such that $\langle P(y), W^* \rangle(x)$ and $\langle P(y), V^* \rangle(x)$ are identically distributed. Furthermore, W^* uses V^* as a black-box.

Proof sketch. Using V^* as a black-box and interacting with instances of P in a sequential manner, W^* will simulate interleaved interactions of V^* with instances of P. When V^* is about to send a message β to some $P^{i,j}$, W^* takes the transcript of all messages sent by V^* to $P^{i,j}$ so far. Then W^* starts a new round, resets $P^{i,j}$, and sequentially simulates the interaction between V^* and $P^{i,j}$ so far (by simply sending the messages from the transcript to $P^{i,j}$). As soon as W^* has send β to $P^{i,j}$ and obtained the response α, W^* completes the interaction with $P^{i,j}$ in an arbitrary way, terminates the round, and passes α to V^*. $\square$

An immediate consequence of this theorem is

Corollary 4. Every RZK (RWI) prover in the sequential model is a RZK (RWI) prover in the interleaved model.

3 A general transformation

3.1 A class of admissible protocols

We will consider an interactive proof (P, V) admissible, if the first verifier message essentially determines its subsequent moves. The first verifier message is interpreted as a commitment to its subsequent moves. During the other moves, it can either open the next commitment, or cause the protocol to abort (by aborting the protocol, or sending an invalid decommitment causing the prover to abort). The verifier's messages are syntactically split into two parts, a main part holding the actual data, and an authenticator consisting of extra decommitment information. The first verifier message is, however, preceded by a prover message. We require that the prover uses two random string. The first is used to produce the first prover message, whereas the second is used for all subsequent prover messages. The precise definition is as follows.

Definition 5. A protocol (P, V) is *admissible*, if the following requirements hold.

- P utilize two random strings, r_1 and r_2. The first string is used to determine the very first prover message m. The second is used for all subsequent prover messages. Both r_i are polynomial in the length of the common input.

- Each verifier message consists of a *main part* and an *authenticator*. The prover P is required to respond to a verifier message as follows. First, the prover decides, based on its own messages, the main part of the very first verifier message, and both parts of the current verifier message, whether to abort or not. If it does not abort, the prover determines its response as a function of r_2 and the main parts of all verifier messages.

- Let V^* be an arbitrary deterministic polynomial size circuit representing a possible verifier strategy in the interactive proof (P, V). Let m be the very first prover message computed as above. Consider a probabilistic polynomial time oracle machine with oracle access to $V_m^*(\cdot) = V^*(m, \cdot)$ (That is, the oracle machine may query V^* on every history of prover messages starting with m). Then it is required that the probability, taken over the choices of m and the oracle machine's internal coin tosses, that the machine obtains for some verifier message two different main parts that are both properly authenticated, is negligible. This requirement ensures that it is infeasible for the adversary to deviate from the predetermined (by the first verifier message) verifier moves, modulo abortion of the protocol.

3.2 The transformation

The following transformation will be our main tool for constructing resettable witness indistinguishable and zero knowledge proof systems in the following sections.

Construction 6. Given an admissible protocol (P, V) and a collection of pseudorandom functions, construct a new protocol (P, V) as follows. The common input x and auxiliary input y are as in (P, V). The new verifier is identical to V. The new prover's randomness is viewed as a pair (r_1, f_2), where r_1 is a string of adequate length for the first part of the random string of P, and f_2 is a description of a pseudo random function from the collection. The new prover simulates P as follows.

- The first message sent by P equals the first message send by P, when the first part of P's random string is r_1. That is, it sends $m = P(x, r_1)$ to the verifier.

- Upon receiving the first verifier message m', the prover sets $r_2 = f_2(x, r_1, m')$. From this step on, P simulates P with random string (r_1, r_2) (Recall that P does not use r_1 for its subsequent messages).

We will apply Construction 6 to admissible protocols secure in the following *hybrid model*. The hybrid model is closer in spirit to a normal concurrent model than to a resettable one.

Definition 7 (Hybrid model). Consider an adversary as in the resettable model, interacting with incarnations of an admissible P, where $P^{i,j,k}$ is determined by the common input x_i, the auxiliary input y_i, and two-part randomization $\omega_{j,k} = (r_1^j, f_2^k)$, where r_1^j and f_2^k are uniformly distributed in the corresponding spaces. The adversary is allowed to initiate at most one session with each incarnation $P^{i,j,k}$. Furthermore, for every k, the adversary is allowed to interact with at most one of the incarnations in $\{P^{i,j,k} \mid i,j\}$.

We are now ready to state and prove the main result of this section.

Theorem 8 ([1]). Let (P,V) be admissible, and let P be the prover strategy obtained by applying Construction 6 to (P,V), assuming that pseudorandom functions exist. Then for every probabilistic polynomial time V^* for the resettable model, there exists a probabilistic polynomial time W^* for the hybrid model, such that $\{\langle P(y), W^*\rangle(x)\}_{x,y}$ and $\{\langle \mathsf{P}(y), V^*\rangle(x)\}_{x,y}$ are computationally indistinguishable.

Proof sketch. By Theorem 3, we may assume that the verifier V^* interacts with incarnations $\mathsf{P}^{i,j} = \mathsf{P}_{x_i,y_i,\omega_j}$ sequentially. We will construct a hybrid model verifier interacting with incarnations of P, $P^{i,j,k}$, as in Definition 7. To satisfy the definition, W^* will initiate each $P^{i,j,k}$ at most once. Furthermore, if it initiates $P^{i,j,k}$, it will not initiate any other $P^{i',j',k}$. The verifier W^* will keep a counter k. When simulating V^*, it handles messages from V^* as follows.

- V^* initiates a new session with some $\mathsf{P}^{i,j}$: In this case, W^* initiates a new session $P^{i,j,k}$ with P and increments the counter k. The initial message m of $P^{i,j,k}$ is forwarded to V^*.
- V^* sends a new first message m' to $\mathsf{P}^{i,j}$: Then W^* sends m' to some of the $P^{i,j,\cdot}$ that still awaits for the first verifier message, obtains the answer, and forwards it to V^*. W^* designates this session with $\mathsf{P}^{i,j}$ as the *active session of* (i,j,m') and stores the prover's response.
- V^* repeats a first message m' to $\mathsf{P}^{i,j}$: In that case, W^* looks up the response for (i,j,m') and sends it to V^*.
- V^* sends a valid non-first message to $\mathsf{P}^{i,j}$: Note that V^* can check the validity of the message by verifying the commitment. There are two cases. Let this be the ξth message for the session. If the prover response for the ξth message in the active session (i,j,m') has not yet been obtained, W^* sends the message to P in the active session, obtains the response, forwards it to V^*, and stores the response. If a response to the ξth message in the corresponding active session already has been obtained, W^* simply looks up the response in its database, and sends it to V^*.
- V^* sends an invalid message to $\mathsf{P}^{i,j}$: In this case, W^* simply sends a standard abort message to V^* indicating that the prover aborted the session.
- V^* terminates: In this case, W^* halts. Its output equals the output of V^*.

W^* clearly operate in the hybrid model. It remains to argue that the outputs of W^* and V^* are computationally indistinguishable. The key observations are:

- In sessions having different first verifier messages, the second parts of the random strings (that is, r_2) are independent. The same holds if a different incarnation $\mathsf{P}^{i,j}$ is used. This follows, since r_2 is computed by applying a pseudorandom function to x_i, r_1^j and m'.
- In sessions having the same common input, the same r_1, and the same first verifier message, the actions of P are essentially determined. The only freedom of V^* on each step is basically to choose to send a predetermined message, or cause the protocol to abort. Thus, all transcripts are prefixes of one predetermined transcript (augmented by an abort message).

The corresponding transcripts are generated by W^* by copying from a real conversation with P. Each set of $\mathsf{P}^{i,j}$ sessions sharing the same first verifier message is simulated using a single active session with P. The way W^* handles invalid messages guarantees that an active session can always be extended to the full predetermined transcript. Thus, W^* and V^* computes their output from essentially the same transcripts in the same way (as W^* outputs the output of V^*). The result follows. □

Corollary 9. Every language L that has an admissible proof system that is zero knowledge (witness indistinguishable) in the hybrid model, has a black-box RZK (RWI) proof system.

4 A constant round RWI protocol for NP

Using the transformation in the previous section, it is straightforward to construct a constant round RWI protocol for **NP**, provided that we have admissible constant round RWI protocols in the hybrid model. Fortunately, we do.

Proposition 10 ([1]). Suppose that there exist a two-round perfectly hiding commitment scheme (see Appendix). Then every language in **NP** has a 5-round admissible proof system that is witness indistinguishable in the hybrid model.

Proof sketch. The 5 round zero knowledge proof system for the **NP** complete GRAPH 3-COLORING in [4] turns out to be both admissible and witness indistinguishable in the hybrid model. In this proof system, the common input is a graph $G = (V, E)$, where $V = \{1, \ldots, n\}$, claimed to be 3-colorable. The auxiliary input of the prover is a 3-coloring $\phi\colon V \to \{1, 2, 3\}$ of G.

1. The verifier commits to a sequence of $t = n|E|$ uniformly and independently chosen edges using a perfectly hiding commitment scheme and sends the commitments to the prover.
2. The prover commits (using a perfectly binding commitment scheme, see Appendix) to tn values corresponding to the colors of all vertices under t random relabelings of ϕ. The commitments are sent to the verifier.
3. The verifier opens the commitment from the first step.
4. The prover verifies the correctness of the decommitments and opens the commitments for the corresponding node colorings.
5. The verifier checks the correctness of the decommitments in the previous step, and checks that the colorings indeed looks as part of a correct 3-coloring. The verifier accepts if and only if all checks pass.

There is a slight problem is this protocol description. Namely, any perfectly hiding commitment scheme (used in the first step) requires at least two rounds (The receiver first sends a random string to the sender. Then the sender sends the commitment to the receiver.). The trick is to consider the prover's initial random string as a pair (r_1, r_2), where r_1 is used in the perfectly hiding commitment scheme, and r_2 is used for the second step.

The resulting protocol is clearly admissible, and as shown in [4], it is indeed a 5 round zero knowledge proof system. By the result in [2, Theorem 5.3.39], it is also witness indistinguishable in the concurrent model. The only difference between the concurrent and the hybrid model is that a verifier in the hybrid model can invoke sessions with the same value of r_1. Since this value is only used to establish the computational binding feature of the verifier's commitment, and this feature continue to hold when r_1 is used more than once, the protocol is also witness indistinguishable in the hybrid model. □

Note that we have used two different types of commitment schemes in the proof. Since we typically do not assume that the prover is computationally bounded, we must ensure that the verifier's commitments are perfectly hiding, and that the prover's commitments are perfectly binding to get a sound proof system. Since a commitment scheme cannot be both perfectly hiding and perfectly binding, we have to use two different schemes. Since perfectly binding (or normal) commitment schemes can be based on any one-way function, we do not list their existence as an additional assumption in the proposition.

Corollary 9 and Proposition 10 immediately give constant round RWI protocols for **NP**.

Corollary 11. Assuming the existence of 2-round perfectly hiding commitment schemes, every language in **NP** has a constant round RWI proof system.

5 A RZK protocol for NP

To show that all languages in **NP** has RZK proofs, we need an admissible proof system for an **NP** complete language that is zero knowledge in the hybrid model.

Proposition 12 ([1]). Suppose that there exist a two round perfectly hiding commitment scheme. Then every language $L \in \mathbf{NP}$ has an admissible proof system that is zero knowledge in the hybrid model.

Proof sketch. The idea behind the proof is to extend the concurrent proof system in [7]. Let x be the common input, and $k = k(|x|)$ be a security parameter. The prover's auxiliary input is an **NP** witness w.

The protocol consists of 2 stages. In the first stage, the prover gets k uniformly distributed bit strings of length k. In the second stage, the prover provides a witness indistinguishable proof that either some of the bit strings is all zero, or the input $x \in L$. The intuition is that, if k is chosen properly, the probability that some of the bit strings is all zero in a real protocol run is negligible. Thus, the protocol constitute a proof system for the language. On the other hand, a simulator can easily force one of the bit strings to be all zero, which allows it to easily simulate the underlying witness indistinguishable protocol (as it gets a witness). Thus, the first stage only serves to provide a trapdoor for the simulator. The details are as follows.

During the first stage of the protocol, the verifier and prover runs a coin-tossing into the well protocol. The verifier picks k uniformly distributed bit strings of length k and commits to them using a perfectly hiding commitment scheme. Denote the random strings by v_i and the commitments by β_i. Then the following two-round interaction goes on for $i = 1, \ldots, k$. First, the prover commits (in a perfectly binding way, see Appendix) to a uniformly distributed k-bit string p_i. Then the verifier opens the commitment β_i. The prover obtains the uniformly distributed string $v_i \oplus p_i$ unknown to the verifier.

During the second stage, the prover provides a witness indistinguishable proof that either $x \in L$, or $v_i = p_i$ for some i.

We will use the witness indistinguishable proof system for Graph 3-coloring used in the proof of Proposition 10 for the second stage. The only modification is that we move all the verifier's commitments (to the edges) to the first stage of the protocol. This makes the resulting protocol admissible.

Straightforward arguments show that the security proof in [7] for the concurrent model can be carried over to the modified protocol for the hybrid model. □

Corollary 13. Assuming the existence of a 2-round perfectly hiding commitment scheme, every language in **NP** has a black-box RZK proof system.

6 The public key model

The *public key model* was introduced in [1] to improve the round efficiency of resettable zero knowledge proof systems. In this model, we assume that each verifier has a public key known by the prover. For simplicity, we assume that all public keys are valid in the sense that some verifier knows the corresponding private key. It is important to note that the public key model is a much weaker assumption than a public key infrastructure assumption. Since our model assumes that everything outside the prover is controlled by an adversary, we actually only assume that there is a fixed set of public keys accessible by the prover, and that somebody knows the corresponding private keys. (Naturally, a zero knowledge prover does not have to care who it is interacting with.) We also assume that the prover cannot break the underlying public-key scheme.

If we consider the RZK protocol for **NP** constructed previously, we see that the only non-constant round part of the protocol is Stage 1 in the proof of Proposition 12. The main purpose of this stage is to provide a *trapdoor* for the simulator. It turns out that this trapdoor can be implemented in a constant number of rounds in the public key model.

The key part of the construction in [1] is the addition of a *trapdoor commitment scheme*, which can be implemented in the public key model. A trapdoor commitment scheme is a 2-round perfectly hiding commitment scheme with the additional feature that the receiver of the commitment posses a trapdoor (that is, the private key) that allows it to open the commitment in an arbitrary way. Note that this feature does not leak information about the actual commitment—it only allows the receiver to open the commitment in a way that looks correct without further interaction with the commitment sender. Since the commitment sender does not know the trapdoor, it is infeasible for him to open the commitment in more than one way. The public keys are not needed for any other part of the construction.

The construction in [1] is essentially identical to the witness indistinguishable proof system in the proof of Proposition 10, except that the verifier commits to the edge sequence using a trapdoor commitment scheme. This modification gives a trapdoor to the simulator, just as Stage 1 in the proof of Proposition 12. A natural modification of the argument in Proposition 12 shows that the resulting protocol is resettable zero knowledge in the hybrid model. An application of Theorem 8 now gives a constant round RZK proof system for **NP**.

We warn the reader that we have ignored several details; see [1] for a more thorough discussion.

The round complexity of RZK protocols in the public key model is further discussed in [5].

It turns out that the *soundness* condition for zero knowledge proofs are more involved in the public key model; see [6] for a discussion of these aspects.

References

[1] R. Canetti, O. Goldreich, S. Goldwasser, and S. Micali. Resettable zero knowledge. In *32th ACM Symposium on Theory of Computing*, 2000. Revised version available online.

[2] U. Feige. *Alternative Models for Zero Knowledge Interactive Proofs.* PhD thesis, Weizmann Institute of Science, 1990.

[3] O. Goldreich. *Foundations of Cryptography: Basic Tools.* Cambridge University Press, 2001.

[4] O. Goldreich and A. Kahan. How to construct constant-round zero knowledge proof systems for NP. *Journal of Cryptology*, 9(2):167–189, 1996.

[5] S. Micali and L. Reyzin. Min-round resettable zero-knowledge in the public-key model. In *Advances in Cryptology—Eurocrypt '01*, pages 373–393, 2001.

[6] S. Micali and L. Reyzin. Soundness in the public-key model. In *Advances in Cryptology—Crypto '01*, pages 542–565, 2001.

[7] R. Richardson and J. Killian. On the concurrent composition of zero-knowledge proofs. In *Advances in Cryptology—Eurocrypt '99*, pages 415–431, 1999.

A Preliminaries

A.1 Commitment schemes

Perfect binding A normal, or perfectly binding, commitment scheme is a commitment scheme with computational secrecy and perfect unambiguity. That is, a commitment scheme is a uniform family of probabilistic polynomial time circuits $\{C_n\}$ such that the following two conditions hold.

- **Perfect unambiguity:** For every n, the supports of $C_n(x)$ and $C_n(y)$, $x \neq y$, $|x| = |y| = \text{poly}(n)$ are disjoint.
- **Computational secrecy:** For all $x \neq y$, $|x| = |y| = \text{poly}(n)$, the commitments $C_n(x)$ and $C_n(y)$ are computationally indistinguishable.

Commitment schemes can be based on one-way functions.

Perfect hiding A perfectly hiding commitment scheme is a commitment scheme with perfect secrecy and computational unambiguity. That is,

- **Computational unambiguity:** It is infeasible to open a commitment in more than one way.
- **Perfect secrecy:** The commitments to two different strings are statistically indistinguishable.