New Notions of Soundness and Simultaneous Resettability in the Public-Key Model[1]

Yunlei Zhao[2 3]

Abstract

In this paper, some new notions of soundness in public-key model are presented. We clarify the relationships among our new notions of soundness and the original 4 soundness notions presented by Micali and Reyzin. Our new soundness notions also characterize a new model for ZK protocols in public key model: weak soundness model. By ``weak" we mean for each common input x selected by a malicious prover on the fly, x is used by the malicious prover at most a-priori bounded polynomial times. The weak soundness model just lies in between BPK model and UPK model. Namely, it is weaker than BPK model but stronger than UPK model. In the weak soundness model (also in the UPK model, since weak soundness model implies UPK model), we get a 3-round black-box rZK arguments with weak resettable soundness for NP.

Note that simultaneous resettability is an important open problem in the field of ZK protocols. And Reyzin has proven that there are no ZK protocols with resettable soundness in the BPK model. It means that to achieve simultaneous resettability one needs to augment the BPK model in a reasonable fashion. Although Barak et al. [BGGL01] have proven that any language which has a black-box ZK arguments with resettable soundness is in BPP. It is the weak soundness that makes us to get simultaneous resettability.

More interestingly, our protocols work in a somewhat ``parallel repetition" manner to reduce the error probability and the verifier indeed has secret information with respect to historical transcripts. Note that in general the error probability of such protocols can not be reduced by parallel repetition. [BIN97]

At last, we give a 3-round non-black-box rZK arguments system with resettable soundness for NP in the preprocessing model in which a trusted third party is assumed. Our construction for such protocol is quite simple. Note that although the preprocessing model is quite imposing but it is still quite reasonable as indicated in [CGGM00]. For example, in many e-commerce setting a trusted third party is often assumed.

The critical tools used in this paper are: verifiable pseudorandom functions, zap and complexity leveraging. To our knowledge, our protocols are also the second application of verifiable pseudorandom functions. The first application is the 3-round rZK arguments with one-time soundness for NP in the public-key model as indicated by Micali and Reyzin [MR01a].

Keywords rZK, arguments, PRF, VRF, zap, complexity leveraging

---

[1] This research is supported by a grant of the City University of Hong Kong (7001023).

[2] Department of Computer Science, City University of Hong Kong, Hong Kong. Email: CSYLZHAO@cityu.edu.hk

[3] Department of computer science, Fudan university, Shanghai, P. R. China.

1 Introduction

The bare public-key (BPK) model was introduced by Canetti, Goldreich, Goldwasser and Micali in the context of resettable zero-knowledge (rZK) [CGGM00]. By the BPK model we mean a model in which all user are assumed to have deposited a public-key in a file that is accessible by all users at all times. The only assumption about this file is that it guarantees that entries in it were deposited before any interaction among the users takes place. No further assumption is made about this file, and so in particular an adversary may deposit many (possibly invalid) public-keys in it (and, in particular, without even knowing corresponding secret keys or whether such exist). The BPK model is very simple and in fact it is a weak version of the frequently used public-key infrastructure (PKI) model, which underlies any public-key cryptosystem or digital signature scheme. Despite its apparent simplicity, the BPK model is quite powerful. While rZK protocols exist both in the standard and in the BPK model, only in the later case can they be constant-round, at least in a black-box sense. For more details regarding this model, the reader are referred to [CGGM00] and [MR01a].

Micali and Reyzin[MR01, R01] first noted and clarified the soundness of protocols in BPK model. In the BPK model, the verifier has a secret key SK, corresponding to its public key PK. Thus, a malicious prover could potentially gain some knowledge about SK from an interaction with the verifier, and this gained knowledge might help him to convince the verifier of a false theorem in a subsequent interaction. In [MR01, R01] four notions of soundness in the public-key model were presented, each of which implies the previous one:

1. **one-time soundness**, when a potential malicious prover is allowed a single interaction with the verifier per theorem statement.
2. **Sequential soundness**, when a potential malicious prover is allowed multiple but sequential interactions with the verifier.
3. **concurrent soundness**, when a potential malicious prover is allowed multiple interleaved interaction with the same verifier.
4. **resettable soundness**, when a potential malicious prover is allowed to reset the verifier with the same random tape and interact with it concurrently.

The above four notions are not only meaningful, but also distinct. That is, for each soundness notion there exists a protocol which satisfies this soundness condition but not satisfy the next one [MR01, R01].

In [MR01a] Micali and Reyzin have proven that any (resettable or not) auxiliary-input ZK protocol (satisfying one-time or stronger soundness) in the BPK model for a language outside of BPP requires at least three rounds. Note that black-box ZK implies auxiliary-input ZK but the opposite direction is not known to be held.[GO94]. Regarding one-time soundness, In [MR01a] Micali and Reyzin also gave a 3-round (optimal according to above result) rZK arguments with one-time soundness for NP in the BPK model under the existence of verifiable pseudorandom functions.

Canetti et al.[CGGM00] first gave a 5-round rZK arguments for NP in the BPK model under the stronger DLP assumption. Their protocol is sequentially sound although they did not explicitly state this. By ``stronger" we mean the intractability assumption of DLP

refers to sub-exponential size circuits rather merely to super-polynomial ones. The round complexity above is further reduced to 4-round by Micali and Reyzin[MR01] using the same assumption as above. It is an interesting open problem whether 3-round rZK protocol with sequential soundness for NP in the BPK model. That is:

Open problem 1: [MR01a] Does there exist a 3-round rZK protocol with sequential soundness for NP in the BPK model?

In [R01} Reyzin also proved that any (resettable or not) black-box ZK protocol satisfying concurrent soundness in the BPK model for a language L outside of BPP requires at least four rounds. Up to now no such protocols have been constructed in the BPK model. In a restricted version of BPK model in which each public key is allowed to use at most a-priori bounded polynomial times (the UPK model) Micali and Reyzin gave a 3-round black-box rZK arguments with concurrent soundness for NP in the UPK model [R01b].

Regarding the resettable soundness, Reyzin [R01] also proved that there is no (resettable or not) black-box ZK protocols satisfying resettable soundness in the BPK model for a langusge outside of BPP. Furthermore, Barak et al. [BGGL01] showed that any language which has a black-box ZK arguments with resettable soundness is in BPP. It means that to get rZK protocols with resettable soundness for NP one needs to either get a non-black-box one or enhance the BPK model in some reasonable fashion. For non-black-box ZK protocols, in [BGGL01] Barak et al. first gave a constant-round resettably-sound non-black-box ZK arguments of knowledge for NP in the standard model assuming the existence of collision-free hash function. But the protocol there is not resettable zero-knowledge and even not concurrent zero-knowledge as well. An interesting important open problem is whether it is possible to simultaneously protect both the prover and the verifier from resetting attack. That is:

Open problem 2: [BGGL01] do language outside of BPP have rZK arguments with resettable soundness?

2. New Notions of Soundness in the BPK Model

In this section, we introduce some new soundness notions in the BPK model and formally define it.

The new soundness notion is ``weak soundness". Roughly speaking, weak soundness means a potential malicious prover is infeasible to convince the verifier of a false statement even he is allowed to interact with the verifier a-priori bounded polynomial times per theorem statement. To get formal definition of weak soundness in BPK model we first give the formal description of BPK model. The following presentation mainly follows [MR01b].

2.1 The Player

Let

- A public file F be a polynomial-size collection of records (id, PK_{id}), where id is a string identifying a verifier, and PK_{id} is its (alleged) public key.
- An (honest) prover P (for a language L) be an interactive deterministic polynomial-time TM that is given as inputs (1) a security parameter 1^n, (2) a n-bit

string $x \in L$, (3) an auxiliary input y, (4) a public file F, (5) a verifier identity id, and (6) a random type w.

- An (honest) verifier V be an interactive deterministic polynomial-time TM that works in two stages. In stage one (the key-generation stage), on input a security parameter 1^n and random tape r, V outputs a public key PK and the corresponding secret key SK. In stage two (the verification stage), on input SK, an n-bit string x and a random tape ρ, V performs an interactive protocol with a prover, and outputs ``accept $x$" or ``reject x".

Completeness for a pair (P, V) is defined the usual way. Consider the following procedure for (P, V), a string $x \in L$ of length n and a string y.

Procedure: Normal-Interaction

1. Run the key-generation stage of V on input 1^n and a random string r to obtain PK, SK.
2. Pick any id, and let F be a public file that contains the record (id, PK_{id}).
3. Pick string w and ρ at random and run P on input 1^n, x, y, id, PK_{id}, w, and the verification stage of V on inputs SK_{id}, x, ρ, letting them interact.

Definition 2.1 A pair (P, V) is complete for an NP-language L if for all n-bit strings $x \in L$ and their NP-witness y, the probability that in an execution of Normal-Interaction V outputs ``accept" differs from 1 by a quantity negligible in n.

2.2 The Various Cheating Provers and Soundness

There are three types of malicious provers. Let

- **A *s*-sequential malicious prover** P^* for a positive polynomial s be a probabilistic polynomial-time TM that, on first input 1^n, runs in at most $s(n)$ stages, so that
  1. In stage 1, P^* receives a public key PK and outputs a string x_1 of length n.
  2. In every even stage, P^* starts in the final configuration of the previous stage and performs a single interactive protocol: it outputs outgoing message and receives incoming message (the machine with which it performs the interactive protocol will be specified below, in the definition of sequential soundness). It can choose to abort an even stage at any point and move on to the next stage by outputting a special message.
  3. In every odd stage $i>1$, P^*starts in the final configuration of the previous stage and outputsa string x_i of length n.
- **A *s*-concurrent malicious prover** P^*, for a positive polynomial s, be a probabilistic polynomial-time TM that, on inputs 1^n and PK, performs at most $s(n)$ interactive protocols as follows:
  1. If P^* is already running i-1 interactive protocols $1 \le i\text{-}1 < s(n)$, it can output a special message ``Start x_i", where x_i is a string of length n.
  2. At any point it can output a message for any of its (at most $s(n)$) interactive protocols (the protocol is unambiguously identified in the outgoing message). It then immediately receives the party's response and continues.
- **A *s*-resetting malicious prover** P^*, for a positive polynomial s, be a probabilistic polynomial-time TM that, on inputs 1^n and PK, gets access to $s(n)$ oracles for the verifier (to be precisely specified below, in the definition of resettable soundness).

Now we define our new soundness notions along with the 4 soundness notions presented by Micali and Reyzin.

Procedure: Sequential-Attack

1. Run the key-generation stage of V on input 1^n and a random string r to obtain PK, SK.
2. Run the first stage of P^* on inputs 1^n and PK to obtain an n-bit string x_1.
3. For i ranging from 1 to $s(n)/2$:
   3.1. Selects a random string ρ_i.
   3.2. Runs the $2i$-th stage of P^*, letting it interact with the verification stage of V with input SK, x_i, ρ_i.
   3.3. Run the (2i+1)-th stage of P^* to obtain an n-bit string x_i.

Definition 2.2 [MR01a] (P, V) satisfies one-time soundness for a language L if for all positive polynomials s, for all s-sequential malicious prover P^*, the probability that in an execution of Sequential-Attack, there exists i such that $1 \le i \le s(n)$, $x_i \notin L$, $x_j \ne x_i$ for all $j<i$ (that is x_i appears once and only once)and V outputs ``accept x_i'' is negligible in n.

Sequential soundness differs from one-time soundness only in that the malicious P^* is allowed to have $x_i=x_j$, for $j<i$.

Definition 2.3 [MR01a] (P, V) satisfies sequential soundness for a language L if for all positive polynomials s, for all s-sequential malicious prover P^*, the probability that in an execution of Sequential-Attack, there exists i such that $1 \le i \le s(n)$, $x_i \notin L$, and V outputs ``accept x_i'' is negligible in n.

Now we give the first new notion: weak sequential soundness. Informally, weak sequential soundness requires that for each common input x, and for any polynomial s, a s-sequential prover can select and use the same x at most a-priori bounded polynomial times in an Sequential Attack.

Definition 2.4 (P, V) satisfies weak sequential soundness for a language L if there exists a-priori bounded polynomial m such that for all positive polynomials s, for all s-sequential malicious prover P^*, the probability that in an execution of Sequential-Attack, there exists i such that $1 \le i \le s(n)$, $x_i \notin L$ and x_i appears at most $m(n)$ times, V outputs ``accept x_i'' is negligible in n.

The difference between weak sequential soundness and sequential soundness lies in that in the definition of sequential soundness each input may appear any polynomial times but in the definition of weak sequential soundness each input is restricted to appear a a-priori bounded polynomial times. Obliviously, the weak sequential soundness just lies in between one time soundness and sequential soundness. In this paper we will give a 3-round rZK arguments with weak sequential soundness for NP in the BPK model, furthermore our protocol also satisfies more stronger soundness, weak concurrent soundness (to be defined below).

Procedure: Concurrent-Attack

1. Run the key-generation stage of V on input 1^n and a random string r to obtain PK, SK.
2. Run P^* on inputs 1^n and PK.
3. Whenever P^* outputs ``Start x_i'', select a fresh random string ρ_i and let the i-th machine with which P^* interacts be the verification stage of V on inputs SK, x_i, ρ_i.

Definition 2.5 [MR01a] (P, V) satisfies concurrent soundness for a language L if for all positive polynomials *s*, for all *s*-concurrent malicious prover P^*, the probability that in an execution of Concurrent-Attack, V ever outputs ``accept *x*'' for $x \notin L$ is negligible in *n*.

Similarly, we get the definition of weak concurrent soundness in the BPK model.

Definition 2.6 (P, V) satisfies weak concurrent soundness for a language L if there exists a-priori bounded polynomial *m* and for all positive polynomials *s*, for all *s*-concurrent malicious prover P^*, the probability that in an execution of Concurrent-Attack, V ever outputs ``accept *x*'' for $x \notin L$ and *x* appears at most $m(n)$ times, is negligible in *n*.

Procedure: Resetting-Attack

1. Run the key-generation stage of V on input 1^n and a random string *r* to obtain *PK*, *SK*.
2. Run P^* on input 1^n and *PK*.
3. Generate $s(n)$ random strings ρ_i, for $1 \le i \le s(n)$.
4. Let P^* interacts with oracles for the second stage of the verifier, the *i*-th oracle having input *SK*, ρ_i.

Definition 2.7 [MR01a] (P, V) satisfies resettable soundness for a language L if for all positive polynomials *s*, for all *s*-resetting malicious prover P^*, the probability that in an execution of Resetting-Attack, P^* ever receives ``accept *x*'' for $x \notin L$ from any of the oracle is negligible in *n*.

Similarly, we get the definition of weak resettable soundness in the BPK model.

Definition 2.8 (P, V) satisfies weak resettable soundness for a language L if there exists a-priori bounded polynomial *m* and for all positive polynomials *s*, for all *s*-resetting malicious prover P^*, the probability that in an execution of Resetting-Attack, P^* ever receives ``accept *x*'' for $x \notin L$ from any of the oracle, and *x* have been queried at most $m(n)$ times, is negligible in *n*.

In this paper we will give a 3-round rZK arguments with weal resettable soundness. Thus we can get some progress towards establishing the open problem 2.

The relationships among all these notions of soundness are presented in the section 4.

Some comments on our new soundness notions are in place. In the subsequent we denote by weak soundness model the BPK model with weak soundness. By ``weak'' we mean for each commom input selected by a malicious prover, the malicious prover is restricted to interact with the verifier V at most a-priori bounded polynomial times using the same common input.

Comment. Weak soundness model versus UPK model:

Note that in UPK model each PK_{id} is restricted to be used at most a-priori bounded polynomial times. This corresponds to in the definitions for soundness notions in BPK model in [MR01a] (that is in definitions 2.2, 2.3, 2.5, 2.7 above) the polynomial *s* is a-priori bounded polynomial. However, in weak soundness model we do not restrict the number of interactions between the malicious prover and the verifier V_{id}. In weak soundness model we allow the malicious prover to interact with V_{id} for any polynomial times just as in the original definitions in [MR01a] but for each theorem statement (common input) we restricted the malicious prover to interact with V_{id} at most a-priori polynomial times using the same common input. Obviously, if each PK_{id} is restricted to be used at most a-priori polynomial times then for each theorem statement (common input) the malicious prover also can interact with V_{id} with PK_{id} at most a-priori

polynomial times. This means that weak soundness model just lies in between BPK model and UPK model. That is, weak soundness model is stronger than BPK model but weaker than UPK model. Any protocol which satisfies weak soundness also satisfies the corresponding soundness in the UPK model. However, a protocol in the UPK model may not satisfy the corresponding weak soundness. Indeed, although the Micali and Reyzin's 3-round rZK protocol in [MR01b] satisfies concurrent soundness in the UPK model but it indeed does not satisfy even weak sequential soundness. The reason is that the concurrent soundness of Micali and Reyzin's protocol is relied on the verifier's *cmax* (which is a-priori bounded polynomial) random strings in his public-key. If the malicious prover can learn all these verifier's random strings then he can easily convince the verifier of a false statement. A malicious prover can learn all these random string by just select *cmax* different inputs on the fly and interact sequentially with the verifier *cmax* times (each time uses a different input, that is each input is uses once and only once). After all these *cmax* sequential interactions the malicious prover can convince V_{id} of a false statement with probability 1. It obviously does not satisfy the weak sequential soundness.

3. Preliminaries

Let us quickly recall the notions, the definitions and the constructions that we utilize in this paper.

Definition 3.1 (Pseudorandom Function PRF) [GGM86] In this paper we will use a stronger version of PRF in which the pseudorandomness is guaranteed against sub-exponential size adversaries rather polynomial size ones.

A function PRF: $\{0,1\}^n \times \{0,1\}^* \to \{0,1\}^n$ is a pseudorandom function if $\exists\ \alpha > 0$ such that for all sufficiently large n and all 2^{n^α} -gate adversaries ADV, the following differences negligible in n:

$$PROB[PRFKey \xleftarrow{R} \{0,1\}^n : ADV^{PRF(PRFKey,\bullet)} = 1] -$$
$$PROB[F \xleftarrow{R} (\{0,1\}^n)^{\{0,1\}^n \times \{0,1\}^*} : ADV^{F(\bullet)} = 1]$$

The value α is the pseudorandomness constant.

Such a PRF can be constructed assuming the security of RSA with large prime exponents against subexponentially-strong adversaries.

Definition 3.2 [BDMP91, BFM88] Non-Interactive Zero-Knowledge (NIZK): Let NIP and NIV be two probabilistic polynomial-time algorithms, and let $NI\sigma\ Len$ be a positive polynomial. We say that (NIP, NIV) is a NIZK arguments system for an NP-language L if

1. **Completeness.** $\forall x \in L$ of lenth n, σ of length $NI\sigma\ Len(n)$, and NP-witness y for x,
$$PROB[\Pi \xleftarrow{R} NIP(\sigma, x, y) : NIV(\sigma, x, \Pi) = YES] = 1$$
2. **Soundness.** $\forall x \in L$ of length n,
$$PROB[\sigma \xleftarrow{R} \{0,1\}^{NI\sigma Len(n)} : \exists \Pi s.t. NIV(\sigma, x, \Pi) = YES]$$
is negligible in n.
3. **Zero-Knowledgeness.** $\exists \alpha > 0$ and a probabilistic polynomial-time simulator NIS such that, $\forall$ sufficient large n, $\forall x \in L$ of length n and NP-witness y for x, the following two distributions are indistinguishable by any 2^{n^α} -gate adversary:
$$[(\sigma', \Pi') \xleftarrow{R} NIS(x) : (\sigma', \Pi')]$$
and

$$PROB[\sigma \xleftarrow{R} \{0,1\}^{NI\sigma Len(n)}; \Pi \xleftarrow{R} NIP(\sigma, x, y) : (\sigma, \Pi)].$$

The value α is the NIZK constant.

Non-interactive zero-knowledge proof system for NP can be constructed based on any one-way permutation. [FLS99, KP98] We note that one-way permutations can be constructed under *RSA* assumption [GB01]. Recent advances in NIZK can be found in [SCOPS01].

3.1 The Malicious Verifier and Resettable Zero-Knowledge in the Public-Key Model

We first define the malicious verifier and the simulator.

- A (s, t)-resetting malicious verifier V^*, for any two positive polynomials s and t, be a TM that runs in two stages so that, on the first input 1^n,
  1. In stage 1, V^* receives $s(n)$ distinct[4] values $x_1,\ldots, x_{s(n)} \in L$ of length n each, and outputs an arbitrary public file F and a list of $s(n)$ identities id_1, …, $id_{s(n)}$.
  2. In stage 2, V^* starts in the final configuration of stage 1, is given oracle access to $s(n)^3$ provers, and then outputs its ``view"of the interactions: its random string and the messages received from the provers.
  3. The total number of steps of V^* in both stages is at most $t(n)$.
- A black-box simulator M be a expected polynomial-time probabilistic algorithm that is given oracle access to an (s, t)-resetting V^* on the $s(n)$ distinct values $x_1,\ldots, x_{s(n)} \in L$.

Definition 3.3 (P, V) is black-box resettable zero-knowledge for an NP-language L if for every pair of positive polynomials (s, t), for every (s, t)-resetting verifier V^*, there exists a black-box simulator M_{V^*} such that the following probability distributions are indistinguishable (in time polynomial in n): Let each distribution be indexed by a sequence of common distinct inputs $\bar{x} = x_1,\ldots, x_{s(n)} \in L$ of length n each and their corresponding NP-witness aux($\bar{x}$)=$y_1,\ldots, y_{s(n)}$:

1. (Distribution 1) The output of V^* obtained from the experiment of choosing w_1, …, $w_{s(n)}$ uniformly at random, running the first stage of V^* to obtain F, and then letting V^* interact in its second stage with the following $s(n)^3$ instances of P: P(x_i, y_i, F, id_k, w_j) for $1 \le i, j, k \le s(n)$.
2. (Distribution 2) The output of M_{V^*} on inputs $x_1, \ldots, x_{s(n)}$.

(P, V) is said to be **resettable witness indistinguishable** (rWI) for an NP-language L if for every pair of positive polynomials (s, t), for every (s, t)-resetting verifier V^*, two distribution ensembles of type 1 above, which are indexed by the same $\bar{x}$ but possibly different sequence of prover's NP-witnesses: $aux^{(1)}(\bar{x}) = y^{(1)}_1,\ldots, y^{(1)}_{s(n)}$ and $aux^{(2)}(\bar{x}) = y^{(2)}_1,\ldots, y^{(2)}_{s(n)}$, are computational indistinguishable.

In [CGGM00] Canetti et al. first gave a 4-round rWI for NP. The round complexity is drastically reduced by Dwork and Naor. In [DN00] they presened a 2-round rWI for NP based on NIZK for NP. Furthermore, more interestingly, their protocol also satisfies resettable soundness.

[4] As noted by Goldreich, if the s(n) common inputs are not distinct then all known rZK protocols are not even rWI.[CGGM00]

Dwork and Naor's 2-round rWI:

The prover P has a private random string s which determines a pseudorandom function f_s. On security parameter 1^n, let p be a positive polynomial and x be the common input and y is the corresponding NP-witness:

Step 1. The verifier V randomly selects (once and fixed for all) $p(n)$ random strings $R_V = r_1, \ldots, r_{p(n)}$ and sends them to prover P.

Step 2. Let $f_s(x, y, R_V)=(r, R_P)$. For each i, $1 \le i \le p(n)$, P uses R_P as the randomness to compute a NIZK proof Π_i with respect to common random string $r \oplus r_i$. At last, P sends r along with all these $p(n)$ NIZK proofs to the verifier.

The following formal definition is almost verbatim from [CGGM00].

Definition 3.4 (perfect-binding bit commitment): A perfect-binding bit commitment is a probabilistic polynomial-time algorithm, denoted C, satisfying:

Computational secrecy: For every v, u of equal poly(n)-length, the random variables $C(1^n, v)$ and $C(1^n, u)$ are computationally indistinguishable by circuits. That is, for every two polynomials p, q, all sufficiently large n's and all $v, u \in \{0,1\}^{p(n)}$ and every distinguishing circuit D of size $q(n)$,

$$\left|\Pr[D(C(1^n, v)) = 1] - \Pr[D(C(1^n, u)) = 1]\right| < \frac{1}{q(n)}$$

Perfect binding: For every v, u of equal poly(n)-length, the random variables $C(1^n, v)$ and $C(1^n, u)$ have disjoint support. That is, for every v, u and m, if $\Pr[C(1^n, v) = m]$ and $\Pr[C(1^n, u) = m]$ are both positive then $u=v$.

The way such a commitment scheme is used should be clear: To commit to a string v, under security parameter n, the sender invokes $C(1^n, v)$ and sends the result as its commitment. The randomness used by C during this computation, is to be recorded and can latter be used as a decommitment.

A one-round commitment scheme as above can be constructed based on any one-way permutation. And since one-way permutation can be constructed under *RSA* assumption [GB01], so the above one-round commitment also can be constructed based on RSA assumption.

In this paper, we will also need another type bit commitment scheme: trapdoor commitment schemes:

Definition 3.5 A trapdoor commitment scheme (TC) is a quintuple of probabilistic polynomial-time algorithms: TCGen, TCCom, TCVer, TCKeyVer and TCFake, such that

1. Completeness. $\forall n, \forall v$,

$$PROB[(TCPK, TCSK) \xleftarrow{R} TCGen(1^n); (c, d) \xleftarrow{R} TCCom(TCPK, v):$$
$$TCKeyVer(TCPK, 1^n) = TCVer(TCPK, c, v, d) = YES] = 1$$

2. Computational Soundness. $\exists \alpha > 0$ such that for all sufficiently large n and for all 2^{n^α}-gate adversaries ADV

$$PROB[(TCPK, TCSK) \xleftarrow{R} TCGen(1^n); (c, v_1, v_2, d_1, d_2) \xleftarrow{R} ADV(1^n, TCPK):$$
$$TCVer(TCPK, c, v_1, d_1) = YES \wedge TCVer(TCPK, c, v_2, d_2) = YES \wedge v_1 \neq v_2] < 2^{-n^\alpha}$$

We call α the soundness constant.

3. Perfect Secrecy. $\forall$ *TCPK* such that TCKeyVer(*TCPK*, 1^n)=YES and $\forall v_1$, v_2 of length of equal length, the following two probability distributions are identical:

$$[(c_1, d_1) \xleftarrow{R} TCCom(TCPK, v_1) : c_1] \text{ and}$$

$$[(c_2, d_2) \xleftarrow{R} TCCom(TCPK, v_2) : c_{21}]$$

4. Trapdoorness. $\forall$ (*TCPK*, *TCSK*) $\in$ {TCGen(1^n)}, $\forall v_1, v_2$ of equal length the following two probability distributions are identical:

$$[(c, d_1) \xleftarrow{R} TCCom(TCPK, v_1); d_2' \xleftarrow{R} TCFake(TCPK, TCSK, c, v_1, d_1, v_2) : (c, d_2')]$$

$$\text{and } [(c, d_2) \xleftarrow{R} TCCom(TCPK, v_2) : (c, d_2)]$$

The above trapdoor bit commitment can be constructed under the strong DLP assumption as follows:

Strong DLP assumption: For every $\alpha > 0$, for every sufficiently large *n*, and every circuit *C* of size at most 2^{n^α}

$$PROB[C(p, g, g^x \bmod p) = x] < 2^{-n^\alpha}$$

where the probability is taken uniformly over all *n*-bit long safe prime *p*, elements *g* of order *q*=(*p*-1)/2, and $x \in Z_q^*$.

3.2 Verifiable Random Functions

The following presentation is mainly followed from [MR01].

A family of verifiable function (*VRF*s), as proposed in [MRV99], is essentially a pseudorandom function family with the additional property that the correct value of a function on an input can not only be computed by the owner of the seed, but also proven to be the unique correct value. The proof can be verified by anyone who knows the public key corresponding to the seed.

More precisely, a *VRF* is a quadruple of functions: (*VRFGen, VRFEval VRFProve, VRFVer*). The function *VRFGen* generates a key pair (*VRFSK, VRFPK*). The function *VRFEval(VRFSK, x)* computes the pseudorandom output *v*. The function *VRFProve(VRFSK, x)* computes pf_x, the proof that *v* is correct. This proof can be verified by anyone who knows the *VRFPK* by using *VRFVer(VRFPK, x, v, pf_x)*. Moreover, no matter how maliciously *VRFPK* is constructed, for each x, there exists at most one v for which a valid proof pf_x exists. The pseudorandomness requirement states that, for all the points for which no proof has been provided, the function *VRFEval(VRFSK, •)* remains indistinguishable from random function even by subexponential-size circuits. The following formal definition is almost verbatim from [MRV99].

Definition 3.6: Let *VRFGen, VRFEval, VRFProve*, and *VRFVer* be polynomial-time algorithms (the first and the last are probabilistic, and the middle two are deterministic). Let *a:* $N \to N \cup \{0,1\}^*$ and *b:* $N \to N$ be any two functions that are computable in time poly(*n*) and bounded by a polynomial in *n* (except when a takes on the value $\{0,1\}^*$).

We say that (*VRFGen, VRFEval, VRFProve, VRFVer*) is a verifiable pseudorandom function (*VRF*) with input *a(n),* and output length *b(n)* if the following properties hold:

1. The following two conditions hold with probability $1 - 2^{-\Omega(n)}$ over the choice of $(VRFPK, VRFSK) \xleftarrow{r} VRFGen(1^n)$:

(a) (Domain-Range correctness):

$$\forall x \in \{0,1\}^{a(n)}, VRFEval(VRFSK, x) \in \{0,1\}^{b(n)}.$$

(b) (Complete Probability): $\forall x \in \{0,1\}^{a(n)}$, if *v=VRFEval(VRFSK, x)* and *pf=VRFProve(VRFSK, x),* then

$$\Pr ob[VRFVer(VRFPK, x, v, pf) = YES] > 1 - 2^{-\Omega(n)}$$

(this probability is over the coin tosses of *VRFVer*).

2. (Unique Probability) For every *VRFPK, x,* v_1, v_2, pf_1, pf_2 such that $v_1 \neq v_2$, the following holds for either *i=1* or *i=2*:

$$\Pr ob[VRFVer(VRFPK, x, v_i, pf_i) = YES] < 2^{-\Omega(n)}$$

(this probability is over the coin tosses of *VRFVer*).

3. (Residual Pseudorandomness): Let $\alpha > 0$ be a constant. Let $T=(T_E, T_J)$ be any pair of algorithms such that $T_E(\bullet, \bullet)$ and $T_J(\bullet, \bullet)$ run for a total of at most 2^{n^α} steps when their first input is 1^n. Then the probability that *T* succeeds in the following experiment is at most $1/2 + 1/2^{n^\alpha}$:

(a) Run $VRFGen(1^n)$ to obtain *(VRFPK, VRFSK).*

(b) Run $T_E^{VRFEval(VRFSK,\bullet),VRF\,\Pr ove(VRFSK,\bullet)}(1^n, VRFPK)$ to obtain the pair *(x, state)*.

(c) Choose $r \xleftarrow{R} \{0,1\}$.

- if *r=0*, let *v=VRFEval(VRFSK, x)*
- if *r=1*, *choose* $v \xleftarrow{R} \{0,1\}^{b(n)}$

(d) Run $T_J^{VRFEval(VRFSK,\bullet),VRF\,\Pr ove(VRFSK,\bullet)}(1^n, VRFPK, v, state)$ to obtain guess.

(e) $T=(T_E, T_J)$ succeeds if $x \in \{0,1\}^{a(n)}$, guess = *r*, and *x* was not asked by either T_E or T_J as a query to *VRFEval(VRFSK, •)* or *VRFProve(VRFSK, •)*.
We call α the pseudorandomness constant.

The above verifiable pseudorandom functions can be constructed assuming the security of *RSA* with large prime exponents against subexponentially-strong adversaries. We refer the reader to [MRV99] for details of the construction.

4 Relations among the soundness notions

In the section we clarify the relations among all the soundness notions presented in section 2.

In [MR01a] Micali and Reyzin have proved the following theorems.

Theorem 4.1[MR01a] If one-way function exist, there is a compiler-type algorithm that, for any language L, and any interactive arguments system for L satisfying one-time soundness, produces another interactive arguments system for the same language L that satisfies one-time soundness but not (**weak**) sequential soundness.

Note that in their proof [MR01a] the malicious prover in the compiled protocol can convince the honest verifier of a false statement if he can invoke the verifier V_{id} using the same common input twice. It means the compiled arguments system does not satisfy weak soundness. Thus, we get:

Theorem 4.2 If one-way function exist, there is a compiler-type algorithm that, for any language L, and any interactive arguments system for L satisfying one-time soundness,

produces another interactive arguments system for the same language L that satisfies one-time soundness but not **weak** sequential soundness.

Theorem 4.3 [MR01a] If one-way functions exist, there is a compiler-type algorithm that, for any language L, and any interactive arguments system for L satisfying sequential soundness, produces another interactive arguments system for the same language L that satisfies sequential soundness but not **(weak)** concurrent soundness.

Theorem 4.4 [MR01a] There exists a compiler-type algorithm that, for any language L, and any interactive proof (or arguments) system for L satisfying concurrent soundness, produces another interactive proof (respectively, arguments) system for the same language L that satisfies concurrent soundness but not **(weak)** resettable soundness.

We stress that in Micali and Reyzin's original proofs they did not explicitly claim the corresponding weak soundness of their compiled protocols. But the compiled protocols in their proof indeed do not satisfy the corresponding weak soundness notions.

The main result of this section is the following theorem:

Theorem 4.5 Assuming the security of RSA with large exponents against subexponentially-strong adversaries, there is a compiler-type algorithm that, for any language L, and any interactive arguments system for L satisfying weak (sequential, concurrent, resettable) soundness, produces another interactive arguments system for the same language L that satisfies weak sequential soundness but not sequential soundness.

Proof. This proof uses ``complexity leveraging" [CGGM00, MR01a].

Let γ be the following constant: for all sufficiently large n, the length of the NP-witness y for $x \in L$ of length n is upper bounded by n^{γ}. We set $\varepsilon > \gamma / \alpha$. For a language $L \in NP$ we construct an arguments system for L on security parameter n while using a PRF with a (larger) security parameter $k = n^{\varepsilon}$. This ensures that one can enumerate all potential NP-witnesses y, in time $2^{n^{\gamma}}$, which is less than the time it would take to break the pseudorandomness of PRF because $2^{n^{\gamma}} < 2^{k^{\alpha}}$.

Let F be a pseudorandom function as defined in definition 3.1: we denote by $F_s(x)$ the output of F with seed s on input x. Note that such functions exist assuming the security of RSA with large exponents against subexponentially-strong adversaries. Let m be the a-priori bounded polynomial guaranteed in the definition of weak (sequential, concurrent, resettable) soundness, we require that: $|F_s(x)| = (m(|x|)+1) \bullet |x|$ and we denote by $F_s(x) = R_1, \ldots R_{m(|x|)+1}$, where for each i, $1 \le i \le m(|x|)+1$, $|R_i| = |x|$.

Let x denote the theorem that the prover is trying to prove to the verifier and suppose $|x| = n$. Given any interactive arguments system (P, V) for a language L in NP satisfying weak (sequential, concurrent, resettable) soundness, we produce another interactive arguments system (P', V') for the same language L that satisfies weak sequential soundness but not sequential soundness.

Add to Key Gen: Generate randomly a n-bit seed s; add s to the key SK_{id}.

Add P step: Set $\beta_1 = \beta_2 = \ldots = \beta_{m(n)+1} = 0$, and sends $(\beta_1 \ldots \beta_{m(n)+1})$ to the verifier.

Add V step: If $F_s(x) = \beta_1 \ldots \beta_{m(n)+1}$, accept and stop. Else randomly selects i in {1, 2, …, $m(n)+1$} and sends R_i to prover.

Note that a sequential malicious prover can get V' to accept with overwhelming probability after sequential $(mn+1)^2$ interactions with the honest verifier V' on the same common input x.

However, if the malicious prover is restricted to use the same common input x at most a-priori bounded polynomial, specifically, $m(n)$, times in his (sequential, concurrent, resettable) interactions with the honest verifier then we will prove in below that it is infeasible for the malicious prover to get the honest verifier accept an x and $x \notin$ L.

First, we assume the F used by V' is a truly random function. It can be easily seen that the malicious prover is infeasible to convince the honest verifier of a false statement if for each common input x, x is used by the malicious prover at most $m(n)$ times in his interactions with the honest verifier. **We stress that we do not restrict the way with which the malicious prover interacts with the honest verifier.** That is: the malicious prover can (sequentially, concurrently, resettably) interact with the honest verifier and the only restriction is that the interactions are ``weak".

Now, we deal with the real case in which the honest verifier uses a pseudorandom function rather truly random function. The subtle problem is that in our contest during the interactions between the malicious prover and the honest verifier the malicious prover selects many common inputs on the fly. The problem is how to distinguish $x \in$ L or $x \notin$ L for each common input selected by the malicious prover on the fly. In [BGGL01] to overcome this problem they required their protocol to be also an arguments of knowledge. What saves us here is the ``complexity leveraging". That is for each common input selected by the malicious prover on the fly we just enumerate all his NP-witnesses in time 2^{n^γ} and decides whether $x \in$ L or not.

Now we claim that the malicious prover is also infeasible to convince the honest verifier of a false statement in his ``weak" interactions with the honest verifier even the function $F$ used by the honest verifier is a pseudorandom function. Otherwise, suppose the malicious prover can convince the honest verifier of a false statement with non-negligible probability in his ``weak" interaction when F is a pseudorandom function. Then we can construct a distinguisher with size lesser than 2^{k^α} and distinguish F from a truly random function with non-negligible probability as follows: The distinguisher runs the malicious prover while oracle access to the honest verifier to simulate the real interactions between the malicious prover and the honest verifier. For each common input x selected by the malicious prover on the fly the simulator decides $x \in$ L or not in time 2^{n^γ}. If during the simulation the distinguisher find that the honest verifier accepts an $x \notin$ L then he says the oracle is using a pseudorandom function. Note that the size of this distinguisher is at most poly(n)• $2^{n^\gamma} < 2^{k^\alpha}$ which violate the pseodorandomness of the PRF F. □

Denote by ``A < B" if there exists a compiler-type algorithm that, for any language L, and any interactive arguments system for L satisfying soundness notion A, produces another interactive arguments system for the same language L that satisfies soundness notion A but not soundness notion B. Then the relations among all the 7 soundness notions in BPK model can be presented as follows:

One-time soundness < weak sequential soundness < **sequential soundness** <weak concurrent sondness <weak resettable soundness< **sequential soundness** < concurrent soundness <resettable soundness.

Note that there is a cycle in the above chain.

5 3-round (optimal) black-box rZK arguments with weak resettable soundness for NP in the BPK model

In this section, we first present a 3-round black-box rZK arguments with weak concurrent soundness, and then transform it into the one satisfying weak resettable soundness. Before present the constructions in detail, we first give a comment on the reasons why our protocol can get simultaneous resettability.

Comment: In [BGGL01] Barak et al. have proven that any language which has a black-box ZK arguments with resettable soundness is in BPP. It seems that our results contradict their results. What saves us here is just the weak soundness. In [BGGL01] their proof relies on two key observations. One is that a resetting malicious prover can emulate the actions of the black-box simulator. Thus, in case $x \in \mathrm{L}$, the resetting malicious prover causes the verifier to accept x with high probability. Another is that by the resettable soundness condition the resetting malicious prover is infeasible to cause the verifier to accept $x \notin \mathrm{L}$. We stress that these two observations do not work in our weak resettable soundness setting. First, in the setting of weak soundness the resetting malicious prover may has no ability to emulate the black-box simulator, otherwise it may violate the weak soundness condition. Actually, in our protocol a malicious resetting prover indeed can not emulate the black-box simulator unless it uses a common input more than $m(n)$ times which violates the weak soundness condition, where m is the priori bounded polynomial guaranteed in the definition of weak soundness. Second, a resetting malicious prover indeed can convince the verifier of a false statement if there is no the weak soundness restriction. What we require is just that if for each $x \notin \mathrm{L}$ x is not used by a resetting malicious prover more than a a-priori bounded polynomial times then this resetting malicious prover is infeasible to cause the verifier accept x. In a nutshell, the observations in [BGGL01] do not work here in our weak soundness setting.

There are three crucial tools in our construction: verifiable pseudorandom functions, Dwork and Naor's 2-round rWI and ``complexity leveraging".

5.1 3-round rZK arguments system with weak concurrent soundness for NP in the BPK model

We use the ``complexity leveraging" as in [MR01a]. Let α be the pseudorandomness constant for VRF (that is the output of VRFEval is indistinguishable from randm for circuit of size $2^{k^{\alpha}}$, where k is the security parameter of VRF). Let γ_1 be the following constant: for all sufficiently large n, the length of the NP-witness y for $x \in \mathrm{L}$ of length n is upper bounded by n^{γ_1}. Let γ_2 be the following constant: for all sufficiently large n, the length of the NIZK proof Π for a statement x' with length poly(n) is upper bounded by n^{γ_2}. We then set $\gamma = \max\{\gamma_1, \gamma_2\}$ and $\varepsilon > \gamma / \alpha$. We use a VRF with a larger security parameter $k = n^{\varepsilon}$. This ensures that one can enumerate all potential NP-witness y, or all potential NIZK proof Π, in time $2^{n^{\gamma}}$, which is less than the time it would take to break the residual pseudorandomness of VRF (because $2^{n^{\gamma}} < 2^{k^{\alpha}}$).

Protocol 5.1

Let x be the common input with length n and m be the priori bounded polynomial as guaranteed in the definition of weak soundness. That is x is not allowed to be used by a malicious prover more than $m(n)$ times during his interactions with the honest verifier V_{id}. We need a verifiable pseudorandom function F with input length n and output length $2m(n) \bullet n^2$. We denote by

$VRFEval(VRFSK, x)= R^1{}_1, R^1{}_2,..., R^1{}_{2m(n)}, R^2{}_1, R^2{}_2,..., R^2{}_{2m(n)},..., R^n{}_1, R^n{}_2,..., R^n{}_{2m(n)}$

the output of VRF on input x of length n, where for each i, $1 \le i \le n$ and each j, $1 \le j \le 2m(n)$, $\left|R^i{}_j\right| = n$.

For a security parameter n, V generates a key pair for the VRF with security parameter k. V then randomly selects $p(n)$ strings ($R_{V_1}, R_{V_2},...R_{V_{p(n)}}$) used as the first round message in a 2-round Dwork and Naor's rWI protocol. *VRFSK* is V's secret key (SK_{id}), and *VRFPK* along with the $p(n)$ random strings is V's public key (PK_{id}).

Public file: A collection F of recoreds (id, PK_{id}), where PK_{id}=($VRFPK$, ($R_{V_1}, R_{V_2},...R_{V_{p(n)}}$)).

Common input: An element $x \in$ L.

P private input: The NP-witness y for $x \in$ L; V's id and the file F; a random string w which determines a PRF f_w.

V private input: A secret key SK.

P step one:

1. Using the string w as a seed for PRF, generates R_P and $2m(n) \bullet n$ strings with length n each: $s^1{}_1, s^1{}_2,..., s^1{}_{2m(n)}, s^2{}_1, s^2{}_2,..., s^2{}_{2m(n)},..., s^n{}_1, s^n{}_2,..., s^n{}_{2m(n)}$, from the inputs x, y and PK_{id}.
2. Selects $2m(n) \bullet n$ arbitrary strings with length $2m(n) \bullet n^2$ each: $t^1{}_1, t^1{}_2,..., t^1{}_{2m(n)}, t^2{}_1, t^2{}_2,..., t^2{}_{2m(n)},..., t^n{}_1, t^n{}_2,..., t^n{}_{2m(n)}$. Let C={$C^{(i,j)}$, where $C^{(i,j)} = C_{s^i{}_j}(t^i{}_j)$ $1 \le i \le n$, $1 \le j \le 2m(n)$}, where C is the one-round perfect binding commitment scheme. P then sends (R_p, C) to V.

V step one: Note that SK_{id}=$VRFSK$.

1. Computes
   R=$VRFEval(SK_{id}, x)$=
   $R^1{}_1, R^1{}_2,..., R^1{}_{2m(n)}, R^2{}_1, R^2{}_2,..., R^2{}_{2m(n)},..., R^n{}_1, R^n{}_2,..., R^n{}_{2m(n)}$, and
   pf_x=$VRFProve(SK_{id}, x)$. **We call each $R^i{}_j$, $1 \le i \le n$, $1 \le j \le 2m(n)$, a block of the pair (x, id).**
2. Randomly selects (j_1, j_2, …j_n), where for each k, $1 \le k \le n$, j_k is uniformly distributed over {1, 2, …, $2m(n)$}. For each k, $1 \le k \le n$, computes V_k= $VRFEval(SK_{id}, R^k{}_{j_k})$ and pf_k=$VRFProve(SK_{id}, R^k{}_{j_k})$. V then sends (($j_1, j_2, \ldots, j_n$), ($V_1$, …, V_n), (pf_1, …, pf_n)) to the prover P.

P step two:

1. Verify that R is correct by invoking $VRFVer(VRFPK, x, R, pf_x)$. If not, abort.
2. For each k, $1 \le k \le n$, verify that V_k is correct by invoking $VRFVer(VRFPK, R^k{}_{j_k}, V_k, pf_k)$. If not , abort.

3. Construct another statement for the Dwork and Naor's rWI protocol: x'=``there exists a NP-witness y such that $(x, y) \in R(x)$ **OR** for each i, $1 \le i \le n$, there exists an $j \in \{1, 2, \ldots, 2m(n)\}$, such that $t^i{}_j = V_i$."
4. Generate and send the second round message of Dwork and Naor's 2-round rWI on the statement x' using y as the witness. The randomness used by P is got by applying his PRF on the transcript so far. That is P sends $\{\text{NIZK}(x', R_P \oplus R_{V_i}), 1 \le i \le p(n)\})$ to the verifier V

V step two: If all these $p(n)$ NIZK proofs above are acceptable then accept, otherwise reject.

Comment: Indeed, our construction above is similar to the one in [RK99, CGGM00, KP01] for non-constant concurrent ZK protocols and rZK protocols for NP. In the protocol presented in [RK99, CGGM00, KP01] it is required that in the first round the verifier sends a message which determines his subsequent actions. The idea there is to let the verifier use a bit commitment scheme with perfect secrecy. However, we do not need this determining first message here since the verifier's action is determined by the VRF he used.

Theorem 5.1 Assuming the security of RSA with large exponents against subexponentially-strong adversaries, the above protocol is 3-round (optimal) black-box rZK arguments with weak concurrent soundness for NP.

Proof. (Sketch)

1**. Completeness**.

The completeness of the above protocol is easily followed from the completeness of the underlying Dwork and Naor's 2-round rWI protocol.

2. Resettable zero-knowledge.

The rZK property can be shown in a way similar to (and simpler than) the way is shown in [CGGM00].

Specifically, for any (s, t)-resetting malicious verifier V^*, and suppose the outputs of the stage one of V^* are: $s(n)$ distinct values $x_1, x_2, \ldots, x_{s(n)} \in L$ of length n each, the public file F and a list of $s(n)$ identities $id_1, id_2, \ldots, id_{s(n)}$. Intuitively, if for each block B_k of pair (x_i, id_j), $1 \le i, j \le n$, $1 \le k \le 2m(n) \bullet n$, the simulator can learn the output of *VRFEval* on B_k, then it is easy for the simulator to generate a transcript which is computational indistinguishable from the real interaction between P and V^*. Since for an (s, t)-resetting verifier V^*, there are at most $s(n)^2 \bullet 2m(n) \bullet n$ blocks, the simulator works as follows while oracle accessing to the V^*:

The simulator works in $s(n)^2 \bullet 2m(n) \bullet n + 1$ phases. In each phase he uses a independent random-type to try to simulate the real interaction between P and V^*. In each phase he either succeeds in getting a simulated transcript which is indistinguishable from the real interaction between P and V^* or learns the output of *VRFEval* for a new block.

3. Weak soundness

We first note that a computational power unbounded prover can easily convince the verifier of a false statement since he can get the corresponding *VRFSK* if his computational power is unbounded. Hence the above protocol constitutes a arguments system rather proof system.

We also note that the above protocol does not satisfy the sequential soundness in the BPK model. Since for a specific input $x \notin L$, and the specific V_{id}, a malicious prover P^*

can convince the V_{id} with overwhelming probability after interacting with V_{id} sequentially expected $(2m(n) \bullet n)^2$ times. However in below we will prove that our protocol does satisfy the weak concurrent soundness and so also satisfies concurrent soundness in the UPK model.

To deal with the soundness of the above protocol we stress that we must be very careful since our protocol works in a somewhat ``parallel repetition" fashion (although it is not a complete parallel repetition). The reason is that our protocol is an arguments system and Bellare et al. have proven that in a 3-round arguments system if the verifier has secret information regarding historical transcripts then parallel repetition can not reduce the error probability in general. [BIN97] Note that in our protocol the verifier indeed has secret information, the *SK*. To our knowledge, our protocol is the first arguments system whose error probability is reduced by an ``parallel repetition".

The following proof uses a standard reduction technique. That is if the above protocol does not satisfy the weak concurrent soundness then we will construct a machine $T=(T_J, T_E)$ to break the residual pseudorandomness of the VRF.

Suppose the above protocol does not satisfy the weak concurrent soundness then in a concurrent attack executed by a *s*-concurrent malicious prover P^* with an honest verifier with *id*, there exists an i, $1 \le i \le s(n)$ such that V accepts x_i, $x_i \notin L$ and x_i is used at most $m(n)$ times by P^*. Now T_E fist guesses this ``i" then simulates the concurrent multiple interactions between P^* and V while running P^*. Note that T_E has oracle access to *VRFEval*(*VRFSK*, $\bullet$) and *VRFProve*(*VRFSK*, $\bullet$) and in V's first step of protocol 5.1, for each k, $1 \le k \le n$, j_k is uniformly distributed over {1, 2, …, $2m(n)$}. Also note that as the Micali and Reyzin's protocol in [MR01b] T_E does not need to rewind the P^*. So, T_E can simulates the multiple concurrent interactions between P^* and V. When it is the time to simulate the *i*-th interaction T_E first determines whether $x_i \notin L$ or not by just enumerating all the NP-witnesses of x_i. If $x_i \notin L$ then T_E runs P^* to get the P^*'s message in the **P step one**. Then T_E randomly selects (j_1, j_2, …j_n) from {1, 2, …, $2m(n)$}just as the honest verifier in **V step one**. Denoted by $B_{j_1},...,B_{j_n}$ the corresponding *n* blocks of the pair (x_i, *id*) selected by T_E. Since x_i has been used at most $m(n)$ times and for each k, $1 \le k \le n$, j_k is uniformly distributed over {1, 2, …, $2m(n)$}, then with probability at least $1\text{-}2^{-n}$ T_E will select a new block from all the $2m(n)n$ blocks of the pair (x_i, *id*). Denote by B_{j_i} the new block selected by T_E. Now T_E outputs (B_{j_i}, *state*), where *state* is T_E's historical view.

Now, T_J receives v, and T_J's job is to find whether v is a random string or *VRFEval*(*VRFSK*, B_{j_i}). T_J then first constructs the new statements x' with respect to (*VRFEval*(*VRFSK*, B_{j_1}), …, *VRFEval*(*VRFSK*, $B_{j_{i-1}}$), *VRFEval*(*VRFSK*, $B_{j_{i+1}}$), v, *VRFEval*(*VRFSK*, B_{j_n})). The key observation is that if v is random then most likely there are no NIZK proofs for x' with respect to (*VRFEval*(*VRFSK*, B_{j_1}), …, *VRFEval*(*VRFSK*, $B_{j_{i-1}}$), *VRFEval*(*VRFSK*, $B_{j_{i+1}}$), v, *VRFEval*(*VRFSK*, B_{j_n})) since $x_i \notin L$ and v is completely unpredictable by the malicious prover P^*. Otherwise, v=*VRFEval*(*VRFSK*, B_{j_i}) then according to our assumption with non-negligible probability there exist acceptable NIZK proofs such that all the $p(n)$ NIZK proofs are all acceptable in the second round of Dwork and Naor's rWI protocol. Note that we can enumerate all the

$p(n)$ NIZK proofs in time $p(n) \bullet 2^{n^\gamma}$. Then we check the fractions of acceptable NIZK proofs in all the NIZK proofs and if this fraction is negligible then we decide that v is a truly random string and if this fraction is non-negligible then we decide that v= $VRFEval(VRFSK, B_{j_i})$. Note that $p(n) \bullet 2^{n^\gamma} < 2^{n^\alpha}$ which violates the residual pseudorandomness of VRF.

According to Reyzin's result [R01] the round complexity of our protocol is indeed optimal. □

5.2 3-round black-box rZK arguments with weak resettable soundness for NP

We transform the 3-round rZK arguments with weak concurrent soundness for NP (protocol 5.1) into one achieving simultaneous resettability. Since we have claimed that weak soundness model is stronger than the UPK model it means that the transformed protocol also remains resettable soundness in the UPK model. Note that in protocol 5.1 the only fresh randomness used by V is to select randomly ($j_1, j_2, \ldots, j_n$) from {1, 2, …, $2m(n)$} in **V step one**. The idea is just let the verifier use a PRF and apply the PRF on (x, id, PK_{id}, SK_{id}, and the message sent by P^* in **P step one** of the current session) to generate the ($j_1, j_2, \ldots j_n$). Suppose a random string s is the seed determines a PRF fs then the verifier in the transformed protocol also adds s to his secret key SK. All the security parameter remains the same as protocol 5.1. And we use the PRF with security parameter and pseudorandomness constant just as the ones of the VRF. That is, for each common input x selected by a malicious prover we can determine $x \in L$ or not, in time 2^{n^γ}, which is less than the time it would take to break the pseudorandomness of the PRF (because $2^{n^\gamma} < 2^{k^\alpha}$). Denote by **Protocol 5.2** the transformed protocol.

Theorem 5.2. Assuming the security of RSA with large exponents against subexponentially-strong adversaries, protocol 5.2 is a 3-round (optimal) black-box rZK arguments with weak resettable soundness for NP.

Proof. (Sketch)

1. Completeness and resettable zero-knowledge

Since zero-knowledge refers to malicious verifier so we do not restrict a malicious verifier to generate ($j_1, j_2, \ldots, j_n$) using a PRF. So the proof for rZK remains the same as the proof for protocol 5.1.

2. Weak resettable soundness

Suppose protocol 5.2 does not satisfy weak resettable soundness. Then in a resetting attack executed by a s-resetting malicious prover P^*, there exists an i, $1 \le i \le s(n)$ such that V accepts x_i, $x_i \notin L$ and x_i is used at most $m(n)$ times by P^*.

We first assume the verifier uses a truly random function rather a pseudorandom one.

Lemma 5.1 Protocol 5.2 satisfies weak resettable soundness assuming V uses a truly random function.

We distinguish two cases in a resetting attach executed by s-resetting malicious prover P^*.

Case 1. For each pair (x, id), P^* never repeats the same **P step one** messages in a resetting attack.

In case 1, since we assume V uses a truly random function and P^* never repeats the same **P step one** message for each pair (x, id), then the output of the truly random function is also truly random and independent since each time the PRF applies to a new

point. That is, in case 1 if V uses a truly random function then protocol 5.2 coincides with protocol 5.1. It means that if the *s*-resetting malicious prover can convince the honest verifier of a false statement with non-negligible probability in a resetting attack against protocol 5.2 then he can also do this in a concurrent attack against protocol 5.1, which contradicts the result that protocol 5.1 is weak concurrent sound.

Case 2. For each pair (x, id), P^* may repeat the same **P step one** messages in a resetting attack.

In case 2 we argue that repeating **P step one** message multiple times does not add power to the malicious prover. The reason is that the bit commitment scheme used by the prover is perfect binding. Suppose, if for an $x \notin L$ P^* can not convince the verifier when his **P step one** message first appears then it also can not convince V afterwards since otherwise it means P^* can decommit to in two different ways with respect to the same **P step one** message, which violates the perfect binding of the underlying commitment scheme. Note that according to the perfect binding of the bit commitment even a computational power unbounded sender can not decommit in two different ways to the same commitment.□

Now we return to deal with the real case in which V uses a pseudorandom function rather a truly random one. The proof is quite similar to the proof presented in proof for theorem 4.5. That is, if protocol 5.2 does not satisfy weak resettable soundness when V uses a pseudorandom function then we can construct a distinguisher with size lesser than $2^{k^{\alpha}}$, where k is the security parameter for PRF. The construction for such a distinguisher is indeed the same as the construction given in proof for theorem 4.5. And the key point is also that for each x_i selected by the malicious prover P^* on the fly during a resetting attack the distinguisher can decide $x_i \notin L$ or not using time much lesser than the time to break the pseudorandomness of the PRF.□

6 3-round Non-Black-box rZK arguments with resettable soundness in the preprocessing model

As indicated by Reyzin, one can not hope to construct black-box rZK protocol with resettable soundness in the BPK model. So, to achieve simultaneous resettability, one needs to enhance the BPK model in some reasonable fashion. In previous sections we have dealt with the weak soundness model which lies in between BPK model and UPK model and we do achieve simultaneous resettability under this model.

There is another more imposing model: preprocessing model [CGGM00]. In general, the preprocessing model postulates that before any interaction among users takes place, the user have to interact with a trusted system manager which issues them certificates in case it did not detect cheating at this stage. In other words, in the preprocessing model each user is guaranteed to know the secret key corresponding to the public-key he registered. As discussed in [CGGM00] although the preprocessing model is a more imposing model it is still quite reasonable in practice. For example, in many e-commerce setting a trusted third party is often assumed.

The organization of this section is just as the one of section 5. That is we first give a 3-round rZK arguments for NP in the preprocessing model and then transform it into the one achieving simultaneous resettability.

6.1 Warm-up: A 3-round non-black-box rZK arguments for NP in the preprocessing model

We use a trapdoor commitment scheme with security parameter K and soundness constant α.

Protocol 6.1

For a security parameter n, V generates a key pair (*TCPK*, *TCSK*) for the trapdoor commitment scheme TC with security parameter k. V then randomly selects $p(n)$ strings $(R_{V_1}, R_{V_2}, ...R_{V_{p(n)}})$ used as the first round message in a 2-round Dwork and Naor's rWI protocol, where p is a positive polynomial. *TCSK* is V's secret key (*SK*), and *TCPK* along with the $p(n)$ random strings is V's public key (*PK*).

Public file: A collection F of records (id, PK_{id}), where PK_{id}=(*TCPK*, $(R_{V_1}, R_{V_2}, ...R_{V_{p(n)}})$).

Common input: An element $x \in$ L.

P private input: The NP-witness y for $x \in$ L; V's id and the file F; a random string w which determines a PRF f_w.

V private input: A secret key *SK*=*TCSK*.

P step one: Using the string w as a seed for PRF, P first generates (R_P, s) from the inputs x, y, PK_{id}. P then selects an arbitrary string v and computes C=*TCCom*(PK, v) suing s as the randomness. P sends (R_P, C) to the verifier V.

V step one: V just randomly selects a string t and sends t to P.

P step two: P

1. Constructs another statement for the Dwork and Naor's rWI protocol: x'=``there exists a NP-witness y such that $(x, y) \in R(x)$ OR v=t''.

2. Using y as the witness, Generates $p(n)$ NIZK proofs on common input x' as the second round message of Dwork and Naor's 2-round rWI. The randomness used by P is got by applying his PRF on the transcript so far. That is P sends {NIZK(x', $R_P \oplus R_{V_i}$), $1 \le i \le p(n)$}) to the verifier V.

V step two: If all the $p(n)$ NIZK proofs above are acceptable then accept, otherwise reject.

The completeness of protocol 7.1 is just followed from the completeness of the underlying NIZK proof systems.

Theorem 6.1. Protocol 6.1 is non-black-box resettable zero-knowledge.

Proof. (sketch) We first consider the simulation for a single session. The simulator S selects an arbitrary string v and also selects two random strings: R_P and s. S then computes C=*TCCom*(PK, v) suing s as the randomness and sends (R_P, C) to the verifier V. After receiving t from V, S uses *TCSK* to decommit to C as t. That is to find s' such that C=*TCCom*(*TCPK*, t) when the randomness is s'. This uses the trapdoor feature of the commitment scheme TC and the hypothesis that the verifier (and so the simulator) knows this trapdoor. S then constructs x' and uses s' as his witness to construct $p(n)$ NIZK proofs and send them to the verifier. Since TC is of perfect secrecy and Dwork and Naor's protocol is rWI we get that the simulated transcripts is computational indistinguishable from a transcript in a real interaction.

Subsequent sessions are simulated in the same way assuming that the inputs of P in current session is different than that in all previous sessions. Otherwise, we simulates **P**

step one by copying the values used in the previous session and simulates P step two just as above.

We stress that the above simulation is not black-box. The reason is that the simulator can not extracts the trapdoor *TCSK* by only oracle accessing to the verifier. □

Theorem 6.2. For each common input x, $x \notin L$, and for each verifier with PK_{id}, any malicious prover of size no more than 2^{k^α} is infeasible to cause the honest verifier V_{id} to accept x in an execution of protocol 6.1.

Proof. (sketch) The proof is quite simple. On one hand, protocol 6.1 is indeed a public-coin one so the responses of V in **V step one** is completely unpredictable by the malicious prover; On the other hand, since the size of the prover is no more than 2^{k^α} then the probability for it cheats by giving two decommitment for a same C sent by him in **P step one** is at most 2^{-k^α} which is negligible; Also note that since $x \notin L$ there no witness y such that $(x, y) \in R(x)$. So, if for a common input x, $x \notin L$, and for a verifier V with PK_{id}, a malicious prover of size no more than 2^{k^α} can cause the honest verifier V to accept x in an execution of protocol 6.1 with non-negligible probability then it will violate the soundness condition of the underlying Dwork and Naor's protocol.□

6.2 3-round non-black-box rZK arguments with resettable soundness for NP in the preprocessing model

We can transform protocol 6.1 into a 3-round non-black-box rZK arguments with resettable soundness. Roughly, the idea is to make the verifier in protocol 6.1 to use a PRF in generating the string t in **V step one**.

Again, we will use the ``complexity leveraging" as in [MR01a]. Let γ be the following constant: for all sufficiently large n, the length of the NP-witness y for $x \in L$ of length n is upper bounded by n^γ. We set $\varepsilon > \gamma / \alpha$ and use a PRF with a (larger) security parameter $k = n^\varepsilon$. This ensures that one can enumerate all potential NP-witnesses y, in time 2^{n^γ}, which is less than the time it would take to break the pseudorandomness of PRF because $2^{n^\gamma} < 2^{k^\alpha}$.

Protocol 6.2

For a security parameter n, V generates a key pair (*TCPK*, *TCSK*) for the trapdoor commitment scheme TC with security parameter k. V then randomly selects $p(n)$ strings $(R_{V_1}, R_{V_2}, ...R_{V_{p(n)}})$ used as the first round message in a 2-round Dwork and Naor's rWI protocol. V also needs to randomly select a string z as the seed of PRF f_z. The pair (*TCSK*, z) is V's secret key (*SK*), and *TCPK* along with the $p(n)$ random strings is V's public key (*PK*).

Public file: A collection F of records (id, PK_{id}), where $PK_{id} = (TCPK, (R_{V_1}, R_{V_2}, ...R_{V_{p(n)}}))$.

Common input: An element $x \in L$.

P private input: The NP-witness y for $x \in L$; V's id and the file F; a random string w.

V private input: A secret key $SK = (TCSK, z)$.

P step one: Using the string w as a seed for PRF, P first generates a (R_P, s) from the inputs x, y, PK_{id}. P then selects an arbitrary string v and computes $C = TCCom(PK, v)$ using s as the randomness. P sends (R_P, C) to the verifier V.

V step one: V computes a string t by applying f_z on (x, PK_{id}, SK_{id}, (R_P, C)) and sends t to P.

P step two: P

1. Constructs another statement for the Dwork and Naor's rWI protocol: x'=``there exists a NP-witness y such that $(x, y) \in R(x)$ OR $v=t$".

2. Using y as the witness, Generates $p(n)$ NIZK proofs on common input x' as the second round message of Dwork and Naor's 2-round rWI protocol. The randomness used by P is got by applying his PRF on the transcript so far. That is P sends {NIZK(x', $R_P \oplus R_{V_i}$), $1 \le i \le p(n)$}) to the verifier V.

V step two: If all the $p(n)$ NIZK proofs above are acceptable then accept, otherwise reject.

Theorem 6.3 Under the strong DLP assumption, protocol 6.2 is a 3-round non-black-box rZK arguments with resettable soundness for NP in the preprocessing model.

Proof. (sketch)

1. Completeness:

The completeness of the protocol 6.2 is just followed from the completeness of the underlying NIZK proofs.

2. Non-black-box resettable zero-knowledge:

Since zero-knowledge refers to malicious verifier so we do not restrict a malicious verifier to generate t using a PRF in **V step one** of protocol 6.1. So the proof for rZK remains the same as proof for theorem 6,1.

3. Resettable soundness:

The proof for resettable soundness is quite similar to the one presented in [BGGL01]. The difference is that the protocol in [BGGL01] is an arguments of knowledge but the protocol 6.2 is not. To overcome this gap we again use the ``complexity leveraging".

First, we consider an imaginary verifier (denoted W_F) who uses a truly random function F rather a pseudorandom one f_z.

Lemma 6.1. A malicious prover is infeasible to convince the imaginary verifier W_F of a false statement even in a resetting attack.

Proof. (sketch)

The proof uses reduction.

We claim that for any polynomial-size s-resetting prover P^* who convinces the verifier W_F to accept some common input $x \notin L$ with non-negligible probability ε, there exists a polynomial-size cheating prover P' of protocol 6.1 that convince the verifier V' with some PK_{id} of protocol 6.1 to accept the same x with probability at least ε / m^2 which is also non-negligible, where m is a bound on the number of messages sent by the prover P^* in an execution of resetting attack, which contradicts the theorem 6.2.

The new cheating prover P' proceeds as follows: It uniformly selects i_1, $i_2 \in \{1, \ldots, m\}$, and invokes (the resetting prover) P^* while emulating an imaginary verifier W_F as follows. If the prefix of the current session transcript is identical to a corresponding prefix of a previous session, then P' answers by copying the same answer it has given in the previous session. If P^* sends a message in **P step one** forms a new transcript prefix, then P' answers according to the following two cases:

1. The index of the current message of P^* does not equal any of the 2 integers i_1, i_2 selected above. In this case, P' provides P^* with a uniformly selected string.

2. Otherwise (i.e. the index of the current message of P^* equals to one of the two integers i_1, i_2), suppose the common input selected by P^* in the current session is x and the verifier in current session has public-key PK_{id}, then P' also make this common input x as the common input for protocol 6.1 and forwards the current message of (P^*) to V' with PK_{id} of protocol 6.1 and feeds P^* with the message it obtains from V'.

Clearly, for any possible choice of the integers i_1, i_2, the distribution of messages seen by P^* when P' emulates an imaginary verifier is identical to the distribution that P' sees when actually interacting with such an imaginary verifier. The reason being that in both cases different prefixes of session transcripts are answered with uniformly and independently distributed strings, while session transcripts with identical prefixes are answered with the same string.

Towards the analysis, we call a message sent by P^* in **P step one** of protocol 7.2 **novel** if it forms a new transcript prefix. The **UrMessage** (prefix Ur means ``the most ancient version of") 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. The **UrMessage** of a novel message is just the message itself. Using this terminology, note that the new prover P' succeeds in cheating V' in an execution of protocol 6.1 if the chosen integers i_1, i_2 equal the indices (within the sequence of all message sent by P^*) of the two UrMessages that corresponds to the two messages sent in a session in which P^* convinced the imaginary verifier of a common input $x \notin L$. Since with probability non-negligible probability ε such a convincing session exists, P' succeeds provided it has guessed its message indices (i.e. 2 indicessout of m).□

We return to deal with the real case in which an honest verifier of protocol 7.2 uses pseudorandom functions rather truly random ones. We claim that a s-resetting malicious prover is infeasible to convince the honest verifier of a false statement in an execution of resetting attack. Otherwise, we can construct a distinguisher with size $poly(n) \bullet 2^{n^\gamma} < 2^{k^\alpha}$ just as the one constructed in proof of theorem 4.5 and the distinguisher can distinguish a pseudorandom function from a truly random one with non-negligible probability, which violates the pseudorandomness condition of the PRF we used. □

Acknowledgement: The author is indebted to Reyzin for his many valuable clarifications and illuminations. The author is also grateful to Lindell for some helpful clarifications.

References

[BDMP91] M. Blum, A. D. Santis, S. Micali and G. Persiano. Noninteractive Zero-Knowledge. SIAM Journal on Computing, 20 (6):1084-1118, 1991.

[BFM88] M. Blum, P. Feldman and S. Micali. Noninteractive Zero-Knowledge and it's Applications. In STOC'88.

[BGGL01] B. Barak, O. Goldreich, S. Goldwasser and Y. Lindell. Resettably-Sound Zero-Knowledge and Its Applications. In FOCS 2001.
Available: http://www.wisdom.weizmann.ac.il/~oded

[BIN97] M. Bellare, R. Impagliazzo and M. Naor. Does Parallel Repetition Lower the Error in Computationally Sound Protocols. In FOCS 1997.
[CGGM00] Ran Canetti, O.Goldreich, S.Goldwasser and S.Micali. Resettable Zero-Knowledge. In STOC'00. . Available from: http://www.wisdom.weizmann.ac.il/~oded/
[DN00] C.Dwork and M. Naor. Zaps and Their Applications. In FOCS 2000.
[FLS99] U. Feige, D. Lapidot and A. Shamir. Multiple Non-Interactive Zero-Knowledge Proofs under General Assumptions. SIAM Journal on Computing, 29(1): 1-28,1999.
[G00] O. Goldreich. Foundation of Cryptography-Fragments of a Book. 2000, Cambridge Publication.
[GB01] S. Goldwasser and M. Bellare. Lecture Notes on Cryptography. 2001, Available from: http://www-cse.ucsd.edu/users/mihir/
[GGM86] O. Goldreich, S. Goldwasser and S. Micali. How to Construct Random Functions. Journal of ACM, 33(4): 792-807, 1986.
[GO94] O.Goldreich and Y. Oren. Definitions and Properties of Zero-Knowledge Proof Systems. Journal of Cryptology, 7(1):1-32, 1994.
[KP98] An Efficient Non-Interactive Zero-Knowledge Proof System for NP with General Assumptions. J. Cryptology, 1998.
[KP01] J. Kilian and E. Petrank. Concurrent and Resettable Zero-Knowledge in Poly-logarithmic Rounds. In STOC'01.
[MR01a] S. Micali and L. Reyzin. Soundness in the Public-Key Model. In Crypto'01. Available from: http://www.cs.bu.edu/~reyzin/
[MR01b] S. Micali and L. Reyzin. Min-Round Resettable Zero-Knowledge in the Public-Key Model. In EuroCrypt'01. Available from: http://www.cs.bu.edu/~reyzin/
[MRV99]: S. Micali, M. Rabin and S. Vadhan. Verifiable Random Functions. In FOCS'99, pp 120-130.
[R01] L. Reyzin. Zero-Knowledge in the Public-Key Model. Ph.D. thesis, MIT, 2001. Available from: http://www.cs.bu.edu/~reyzin/
[RK99] R. Richardson and J. Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. In EuroCrypt'99. 415-431.
[SCOPS01] Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano and Amit Sahai. Robust Non-interactive Zero Knowledge. In Cryoto 2001.