Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 48 (2005)

Constant-Round Concurrently-Secure rZK with (Real) Bare Public-Keys

Moti Yung [*] Yunlei Zhao [†]

Abstract

We present constant-round concurrently secure (sound) resettable zero-knowledge (rZK-CS) arguments with real bare public-keys (i.e., public keys based on reduced complexity assumptions). In particular, for general $\mathcal{NP}$ ZK-arguments we give round-optimal protocols as well as minimal hardness assumption based protocols. We also give highly efficient ZK-arguments for number-theoretic languages. Our constructions employ a number of new techniques/tools which are of possibly independent interest.

1 Introduction

Zero-knowledge (ZK) protocols are remarkable since they allow a prover to validate theorems to a verifier without giving away any other knowledge (i.e., computational advantage). This notion was suggested by Goldwasser, Micali and Rackoff [27] and its generality was demonstrated by Goldreich, Micali and Wigderson [26]. Since its introduction ZK has found numerous and extremely useful applications. ZK protocols can be executed in many environments (models), and resettable zero-knowledge (rZK) is the most restrictive model of zero-knowledge to date. It was put forth by Canetti, Goldreich, Goldwasser and Micali [8], motivated by implementing zero-knowledge provers using smart-cards or other devices that may be (maliciously) reset to their initial conditions and/or cannot afford to generate fresh randomness for each new invocation. rZK also preserves the prover's security when the protocol is executed concurrently in an asynchronous network like the Internet (in fact, rZK is a generalization and strengthening of the notion of concurrent zero-knowledge introduced by Dwork, Naor and Sahai [16]). In a nutshell, an rZK protocol remains secure even if the verifier concurrently interacts with the prover polynomially many times, each time restarting an interaction with the prover using the same configuration and random tape.

ZK protocols for general languages (i.e., $\mathcal{NP}$) constitute an important plausibility result since many important statements are in $\mathcal{NP}$. In addition, ZK has many direct efficient applications (mainly employing number theoretic statements). A major direct application is identification (ID) schemes advocated in [22, 21]. While in many settings the paradigm transforms ZK protocols to ID schemes, these protocols' security fails whenever the prover is resettable.

A major measure of efficiency for interactive protocols is the round-complexity. Unfortunately, there are no constant-round rZK protocols in the standard model, at least for the black-box case, as implied from the works of Canetti, Killian, Petrank and Rosen [9, 10]. To get constant-round resettable zero-knowledge protocols, the work in [8] introduced a simple model with very appealing trust requirement, the *bare public-key* (BPK) model. A protocol in BPK model simply assumes that all verifiers (whether honest or dishonest) have deposited a public key in a public file before any interaction takes place

[2]Department of Computer Science, Columbia University, New York, NY, USA. moti@cs.columbia.edu

[3]Software School, Fudan University, Shanghai 200433, China. Work partially done while the author was at Hewlett-Packard European Research Center, Bristol, UK. 990314@fudan.edu.cn

ISSN 1433-8092

among the users*. The sole assumption is that entries in the public file were deposited before any interaction among the users takes place. But, no assumption is made on whether the public-keys deposited are unique or "nonsensical" or "bad" (e.g., for which no corresponding secret-keys exist or are known) public-keys [8]. Note that an adversary may deposit many (possibly invalid or fake) public keys without any guarantee on the properties of the registered public-keys. In particular, for public-keys registered by an adversary it is *not* guaranteed that one can efficiently verify whether the adversary knows corresponding secret keys or *whether such exist* altogether. What is essentially guaranteed by the BPK model is a limitation of the number of different identities that a potential adversary may assume (note that the adversary may try to impersonate any registered user in the public-file, but it cannot act on behalf of a non-registered user), and, in fact, there are no other assurances.

The BPK model is thus very simple, and it is in fact a weaker version of the frequently used public-key infrastructure (PKI) model, which underlies any public-key cryptosystem or any digital signature scheme. For cryptographic protocols in the BPK model, what kind of keys are allowed is an important *system* parameter. *Thus, the complexity assumption underlying the honest users' keys is an important parameter of the protocol in the BPK model.* We say a protocol is with bare *but self-certified* public-keys if it enforces public-keys to be self-certified (i.e., efficiently and publicly verifiable) for their validity and the prover interacts with (even malicious) verifiers *only* with respect to *valid* public-keys (this essentially might restrict the ability of malicious verifiers in extracting "valuable" knowledge from interactions with the honest prover). Note that the validity of a key implies more than the existence of corresponding secret-key. In particular, a *valid* public-key may *assure* some special properties (other than only an arbitrary string). Formally, the validity requirement implies that a complexity assumption supporting the validity self-certified property of the keys is needed. Recall that the initial work on BPK [8] assumed that the public key merely serves as an identity rather than a valid public-key, thus it is desired to develop protocols and protocol techniques with this property, which we informally call "real bare public keys" that formally implies potentially a weaker complexity assumption for the protocol to be secure. This is crucial for us since we attempt, among other things, to get protocols based on the weakest possible assumption.

Despite its power in achieving round-efficient rZK protocols, the soundness notion in the BPK model turns out to be much more complex and subtler than that of the standard model, as noted by Micali and Reyzin [31]. In public-key models, a verifier V has a secret key SK, corresponding to its public-key PK. A malicious prover P^* could potentially gain some knowledge about SK from an interaction with the verifier. This extra gained knowledge may help this prover convincing the verifier of a false theorem in another interaction. Micali and Reyzin showed that under standard intractability assumptions there are four distinct meaningful notions of soundness, i.e., from weaker to stronger: one-time, sequential, concurrent and resettable soundness. In this paper we focus on concurrent soundness which roughly means, for zero-knowledge protocols, that a malicious prover P^* cannot convince the honest verifier V of a false statement even when P^* is allowed multiple interleaving interactions with V. Micali and Reyzin also showed that any (black-box) rZK protocols with concurrent soundness in the BPK model (for non-trivial languages outside $\mathcal{BPP}$) must run at least four rounds [31].

Constant-round rZK-CS arguments for $\mathcal{NP}$ in the BPK model but with self-certified public-keys were recently achieved in [15] (requiring that ELGamal encryption is secure, i.e., the decisional Diffie-Hellman DDH assumption). Constant-round rZK-CS arguments for $\mathcal{NP}$ were also achieved in some stronger versions of the BPK model [32, 37]. But, it is still an open challenge in this field to achieve constant-round rZK-CS arguments for $\mathcal{NP}$ with real (i.e., reduced complexity assumption) bare public-keys in order to reduce the complexity assumptions needed as well as to simplify the protocol structure. We explore this basic challenge in this work, seeking to achieve solutions to the fundamental issues of

*The BPK model does allow dynamic key registrations and readers are referred to [8] for the details of dealing with dynamic key registrations.

minimal round and minimal complexity assumption protocols.

We further note that in spite of its significance to practice, especially to smart-card based identification and other potential validations, all previous rZK proof systems concentrated on plausibility. The inefficiency in previous rZK systems is in part due to their basic techniques. Namely, (multiple) general $\mathcal{NP}$-reductions, non-interactive zero-knowledge (NIZK), non-black-box technique, etc. As a result, there is a gap between plausibility of rZK and solutions for specific useful number theoretic languages (that are typically used in applications). Therefore, it is an important open question to develop new techniques (bypassing the general $\mathcal{NP}$-reductions) that can be implemented with small constant number of (say) exponentiations under widely used hardness assumptions. We explore this central issue in this work as well.

1.1 Our contributions

We achieve constant-round concurrently-secure rZK (rZK-CS) arguments with *real* bare public-keys, which leads us to new techniques/tools and to answering some fundamental open questions. More specifically we get the following new results and their correspomding new techniques/ applications:

- **Result:** For any $\mathcal{NP}$-language, we achieve 5-round rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under any preimage-verifiable one-way function (OWF), and 4-round (i.e., *optimal*) rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under any one-way permutation OWP and any preimage-verifiable OWF. **Technique:** To this end, we develop a one-round trapdoor commitment scheme (that is based on any OWP). This construction is of independent interest and may be applicable elsewhere. (We note that a OWF f is preimage-verifiable if given a string y one can efficiently verify whether or not there exists an x such that $y = f(x)$. Note that preimage-verifiable OWF is a generic and actually quite weak hardness assumption that includes, in particular, any certified one-way permutation and any 1-1 length-preserving one-way function.)

- **Result:** We give 5-round rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under any OWF and the existence of zaps, and 4-round (i.e., *optimal*) rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under any one-way permutation OWP and the existence of zaps. Note that the existence of (single-theorem) NIZK proofs for $\mathcal{NP}$ implies the existence of zaps. **Technique:** To this end, we involve a novel use of zaps in the public-key setting for implementing a generalized version of preimage-verifiable OWFs in the public-key setting, which might also be of independent value.

- **Result:** We also achieve constant-round rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under the minimal hardness assumptions. Specifically, we present 7-round (resp. 6-round) rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under any subexponentially strong OWF (resp. OWP). **Technique:** To this end, we present constant-round resettable witness indistinguishability (rWI) *arguments* for $\mathcal{NP}$ *in the standard model* under minimal hardness assumptions, a result unknown previously (to the best of our knowledge) that is of independent value.

- **Result:** Beyond plausibility, by a novel use of Σ_{OR}-protocols we present a *generic yet practical* transformation achieving 5-round rZK-CS arguments in the BPK model (*in which the first-round message can be fixed once and for all*). By "generic" we mean applicability to any language that admits Σ-protocols. By "practical", we mean that our construction does not go through general $\mathcal{NP}$-reductions, and if the starting Σ-protocol and the underlying PRF are practical then the transformed rZK-CS protocol is also practical. For example, when instantiated with DL or RSA functions, together with the Naor-Reingold practical PRFs, the transformed rZK-CS protocol (for the languages of DL or RSA respectively) employs a very small constant number of

exponentiations. This is the first protocol of this efficiency nature. **Application:** This directly implies practical DLP or RSA based identification scheme secure against resetting attacks.

In addition, since rZK is a generalization and strengthening of the notion of concurrent ZK and because zero-knowledge plays a central role in modern cryptography, our protocols can be employed as critical building blocks to achieve other round-efficient practical or theoretical (but round-optimal or minimal hardness assumption based) concurretnly/resettablly secure cryptographic schemes with real bare public-keys (under the various assumptions above).

1.2 Comparisons with the recent work of [15]

The rZK-CS protocol of [15] is with bare *but self-certified* public-keys. In the protocol of [15], the verifier's public-key (registered in the public-file) serves as the public-key of the underlying public-key encryption scheme used by the verifier. But, the proof of rZK in [15] critically depends on the following additional assumptions on public-keys generated by even malicious verifiers.

1. The validity of public-keys generated by malicious verifiers can be efficiently verified. That is, the validity of public-keys registered by even malicious verifiers is required to be self-certified, and an adversary cannot deposit "nonsensical" or "bogus" keys without being detected efficiently.

2. Any ciphertext generated with a valid public-key corresponds to at most one plaintext.

The underlying PK encryption is actually used by the verifier as a perfectly-binding commitment scheme in a coin-tossing subprotocol with its output, in turn, serving as the common random string for the underlying NIZK from the prover to the verifier. If the above additional assumptions on public-keys generated by malicious verifiers do not hold, then a malicious verifier will have the potential ability to set the output of the coin-tossing protocol, by which it can extract the prover's private $\mathcal{NP}$-witness in case that the underlying NIZK is NIZKPOK. Thus, for the protocol techniques used there, the requirement seems to be needed.

Given the nature of the keys, this implies that, formally, the hardness assumptions used in [15] are much stronger than those used in our protocols for $\mathcal{NP}$ (and are thus, not generic), and the techniques do not lead to reduced complexity assumption. The protocol of [15] assumes any 1-1 length-preserving one-way function and a special form of public-key encryption with known implementation based on DDH. Note that, in particular, any 1-1 length-preserving one-way function is trivially a preimage-verifiable OWF (more details are given in Section 2).

Note again that the earlier work only considered general plausibility for $\mathcal{NP}$ languages (going through multiple $\mathcal{NP}$-reductions and using general NIZK which is prohibitively expensive for practical adaptation of the techniques).

1.3 Organizations

We recall preliminaries in Section 2. In Section 3, we give the formal definitions of resettable zero-knowledge and concurrent soundness in the BPK model. In Section 4, by a novel use of Σ_{OR}-protocols we present the generic yet practical transformation (without $\mathcal{NP}$-reductions) transforming Σ-protocols to rZK-CS protocols in the BPK model. In Section 5, we first present 5-round rZK-CS arguments for $\mathcal{NP}$ with real bare public-keys under any preimage-verifiable OWF, and then show how the round-complexity can be further reduced to four (that is optimal) by developing a one-round OWP-based trapdoor commitment scheme. In Section 6, we further show how to replace the underlying preimage-verifiable OWFs (used in the above 5-round and 4-round preimage-verifiable OWF based rZK-CS arguments for $\mathcal{NP}$) by any OWFs together with a novel use of zaps in the public-key setting, while remaining almost the same protocol structure. In Section 7, we present constant-round rZK-CS arguments for $\mathcal{NP}$ with real

bare public-keys under minimal hardness assumptions. To this end, we develop constant-round rWI arguments for $\mathcal{NP}$ under minimal hardness assumptions in the standard model. We end this paper by providing concluding remarks and suggesting future investigations in Section 8.

2 Preliminaries

We quickly recall basic definitions and preliminaries in this section.

We use standard notations and conventions below for writing probabilistic algorithms and experiments. If A is a probabilistic algorithm, then $A(x_1, x_2, \cdots; r)$ is the result of running A on inputs $x_1, x_2, \cdots$ and coins r. We let $y \leftarrow A(x_1, x_2, \cdots)$ denote the experiment of picking r at random and letting y be $A(x_1, x_2, \cdots; r)$. If S is a finite set then $x \leftarrow S$ is the operation of picking an element uniformly from S. If α is neither an algorithm nor a set then $x \leftarrow \alpha$ is a simple assignment statement.

Definition 2.1 (preimage-verifiable one-way function) *A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is called a preimage-verifiable one-way function (OWF) if the following conditions hold:*

1. *Easy to compute: There exists a (deterministic) polynomial-time algorithm A such that on input x algorithm A outputs $f(x)$ (i.e., $A(x) = f(x)$).*
2. *Hard to invert: For every probabilistic polynomial-time PPT algorithm A', every positive polynomial $p(\cdot)$, and all sufficiently large n's, it holds $\Pr[A'(f(U_n), 1^n) \in f^{-1}(f(U_n))] < \frac{1}{p(n)}$, where U_n denotes a random variable uniformly distributed over $\{0,1\}^n$. A OWF f is called* sub-exponentially strong *if for some constant c, $0 < c < 1$, for every sufficiently large n, and every circuit C of size at most 2^{n^c}, $\Pr[C(f(U_n), 1^n) \in f^{-1}(f(U_n))] < 2^{-n^c}$.*
3. *Easy to verify the preimage existence: There exists a polynomial-time computable predicate $D_f : \{0,1\}^* \longrightarrow \{0,1\}$ such that for any string y, $D_f(y) = 1$ if and only if there exists an x such that $y = f(x)$.*

We remark preimage-verifiable OWF is a generic and actually quite weak hardness assumption that includes, in particular, any certified one-way permutation and any 1-1 length-preserving one-way function. A permutation family is certified if it is easy to verify (in polynomial time) that a given function belongs to the family. For the formal definition of certified one-way permutations, readers are referred to [4].

Below, with RSA as an example, we give more clarifications about the relationship among normal OWF, preimage-verifiable OWF and 1-1 OWF. (More detailed clarifications can be found in [25].) The difference between normal OWF and length-preserving 1-1 OWF lies in the domain of the function. Given a normal OWF from domain D to Range R, the 1-1 property of the function is defined however with respect to $\{0,1\}^* \longrightarrow \{0,1\}^*$. In other words, length-preserving 1-1 OWF is a single object, while a normal OWF could be a member of a function family. This in particular implies that any length-preserving 1-1 OWF is trivially a preimage-verifiable OWF.

For example, given a function description (N, e) where $N = pq$ for two distinct primes p and q and $gcd(e, \phi(N)) = 1$, define this generic RSA-based function (denoted by f_{GRSA}) to be $f_{GRSA}(x) = x^e \bmod N$. Then, this function f_{GRSA} (specified by (N, e)) is a normal OWF (actually OWP) from Z_N^* to Z_N^*. But, this function is NOT a preimage-verifiable OWF, because given a specific (N, e), one cannot efficiently verify whether $gcd(e, \phi(N)) = 1$ or not.

Now, consider the following *restricted* RSA-based function f_{RRSA} (specified by (N, e) such that N is a composite number and $e > N$ is a prime number). In this case, this function f_{RRSA} is a preimage-verifiable OWF from Z_N^* to Z_N^*, as the additional requirement $e > N$ ensures that $gcd(e, \phi(N)) = 1$. But, this function is clearly NOT a 1-1 function from $\{0,1\}^* \longrightarrow \{0,1\}^*$.

Definition 2.2 (interactive argument system) *A pair of probabilistic polynomial-time interactive machines, $\langle P, V\rangle$, is called an interactive argument system for a language L if the following conditions hold:*

- *Completeness. For every $x \in L$, there exists a string w such that for every string z, $\Pr[\langle P(w), V(z)\rangle(x) = 1] = 1$.*
- *Soundness. For every polynomial-time interactive machine P^*, and for all sufficiently large n's and every $x \notin L$ of length n and every w and z, $\Pr[\langle P^*(w), V(z)\rangle(x) = 1]$ is negligible in n. An interactive protocol is called a* proof *for L, if the soundness condition holds against any (even power-unbounded) P^* (rather than only PPT P^*).*

Definition 2.3 (pseudorandom functions PRF) *On a security parameter n, let $d(\cdot)$ and $r(\cdot)$ be two positive polynomials in n. We say that*

$$\{f_s : \{0,1\}^{d(n)} \longrightarrow \{0,1\}^{r(n)}\}_{s\in\{0,1\}^n}$$

is a pseudorandom function ensemble if the following two conditions hold:

1. *Efficient evaluation: There exists a polynomial-time algorithm that on input s and $x \in \{0,1\}^{d(|s|)}$ returns $f_s(x)$.*
2. *Pseudorandomness: For every probabilistic polynomial-time oracle machine A, every polynomial $p(\cdot)$, and all sufficiently large n's, it holds:*

$$|\Pr[A^{F_n}(1^n) = 1] - \Pr[A^{H_n}(1^n) = 1]| < \frac{1}{p(n)}$$

where F_n is a random variable uniformly distributed over the multi-set $\{f_s\}_{s\in\{0,1\}^n}$, and H_n is uniformly distributed among all functions mapping $d(n)$-bit-long strings to $r(n)$-bit-long strings.

PRFs can be constructed under any one-way function [24, 23]. The current most practical PRFs are the Naor-Reingold implementations under the factoring (Blum integers) or the decisional Diffie-Hellman hardness assumptions [34]. The computational complexity of computing the value of the Naor-Reingold functions at a given point is about two modular exponentiations and can be further reduced to only two multiple products modulo a prime (without any exponentiations!) with natural preprocessing, which is great for practices involving PRFs.

Definition 2.4 (perfectly-binding bit commitment scheme) *A pair of PPT interactive machines, $\langle P, V\rangle$, is called a perfectly-binding bit commitment scheme, if it satisfies the following:*

Completeness. *On a security parameter n, denote by $view_{V(r)}^{P(\sigma,s)}(1^n)$ the view of the commitment receiver V in an interaction with the committer P who is committing to a bit $\sigma \in \{0,1\}$ by using local coins s, which consists of the random coins used by V (i.e., r) and the sequence of messages received from P (denoted by $\bar{m}$). Then for any n, any $s, r \in \{0,1\}^*$, and any $\sigma \in \{0,1\}$, $\Pr[V(1^n, view_{V(r)}^{P(\sigma,s)}(1^n), \sigma, s) = 1] = 1$. Here, $view_{V(r)}^{P(\sigma,s)}(1^n)$ stands for the commitment to σ, and s is the corresponding decommitment information to σ.*

Computationally-hiding. *For all sufficiently large n's, the distributions of $view_{V^*}^{P(0)}(1^n)$ and $view_{V^*}^{P(1)}(1^n)$ (for any arbitrary PPT V^*) are computationally indistinguishable. Namely, for all sufficiently large n's, for any positive polynomial $q(\cdot)$, and every distinguishing circuit D of size $q(n)$, it holds that $|\Pr[D(1^n, view_{V^*}^{P(0)}(1^n)) = 1] - \Pr[D(1^n, view_{V^*}^{P(1)}(1^n)) = 1]| < \frac{1}{q(n)}$. We say the hiding property is* sub-exponentially strong *if the hiding property holds also with respect to subexponential-size circuits (i.e., replace the polynomial $q(\cdot)$ above by a function f of the form $f(n) = 2^{n^c}$, for some constant c, $0 < c < 1$).*

Perfectly-binding. *On a security parameter n, we say that a receiver's view, $(r, \bar{m})$, is a possible σ-commitment if there exists a string s such that $(r, \bar{m}) = view^{P(\sigma,s)}_{V(r)}(1^n)$. Then, for all but a negligible fraction of the random coins of V, $r \in \{0,1\}^{poly(n)}$, there is no $\bar{m}$ such that $(r, \bar{m})$ is both a possible 0-commitment and a possible 1-commitment.*

Definition 2.5 (trapdoor commitment scheme TC) *A (normal) trapdoor commitment scheme (TC) is a quintuple of probabilistic polynomial-time (PPT) algorithms TCGen, TCCom, TCVer, TCKeyVer and TCFake, such that*

- *Completeness.* $\forall n$, $\forall v$, $\Pr[(TCPK, TCSK) \stackrel{R}{\leftarrow} \text{TCGen}(1^n); (c, d) \stackrel{R}{\leftarrow} \text{TCCom}(TCPK, v) : \text{TCKeyVer}(TCPK, 1^n) = \text{TCVer}(TCPK, c, v, d) = 1] = 1$.
- *Computational Binding. For all sufficiently large n's and for any PPT adversary A, the following probability is negligible in n:* $\Pr[(TCPK, TCSK) \stackrel{R}{\leftarrow} \text{TCGen}(1^n); (c, v_1, v_2, d_1, d_2) \stackrel{R}{\leftarrow} \text{A}(1^n, TCPK) :$

  $\text{TCVer}(TCPK, c, v_1, d_1) = \text{TCVer}(TCPK, c, v_2, d_2) = 1 \text{ and } v_1 \neq v_2]$.
- *Perfect (or Computational) Hiding.* $\forall$ *$TCPK$ such that* $\text{TCKeyVer}(TCPK, 1^n) = 1$ *and* $\forall$ *v_1, v_2 of equal length, the following two probability distributions are identical (or computationally indistinguishable):* $[(c_1, d_1) \stackrel{R}{\leftarrow} \text{TCCom}(TCPK, v_1) : c_1]$ *and* $[(c_2, d_2) \stackrel{R}{\leftarrow} \text{TCCom}(TCPK, v_2) : c_2]$.
- *Perfect (or Computational) Trapdoorness.* $\forall$ $(TCPK, TCSK) \in \{\text{TCGen}(1^n)\}$, $\exists v_1$, $\forall v_2$ *such that v_1 and v_2 are of equal length, the following two probability distributions are identical (or computationally indistinguishable):*
  $[(c_1, d_1) \stackrel{R}{\leftarrow} \text{TCCom}(TCPK, v_1); d_2' \stackrel{R}{\leftarrow} \text{TCFake}(TCPK, TCSK, c_1, v_1, d_1, v_2) : (c_1, d_2')]$ *and*
  $[(c_2, d_2) \stackrel{R}{\leftarrow} \text{TCCom}(TCPK, v_2) : (c_2, d_2)]$.

Definition 2.6 (witness indistinguishability WI) *Let $\langle P, V\rangle$ be an interactive system for a language $L \in \mathcal{NP}$, and let R_L be the fixed $\mathcal{NP}$ witness relation for L. That is, $x \in L$ if there exists a w such that $(x, w) \in R_L$. We denote by $view^{P(w)}_{V^*(z)}(x)$ a random variable describing the transcript of all messages exchanged between a (possibly malicious) PPT verifier V^* and the honest prover P in an execution of the protocol on common input x, when P has auxiliary input w and V^* has auxiliary input z. We say that $\langle P, V\rangle$ is witness indistinguishable for R_L if for every PPT interactive machine V^*, and every two sequences $W^1 = \{w^1_x\}_{x\in L}$ and $W^2 = \{w^2_x\}_{x\in L}$ for sufficiently long x, so that $(x, w^1_x) \in R_L$ and $(x, w^2_x) \in R_L$, the following two probability distributions are computationally indistinguishable by any non-uniform PPT algorithm: $\{x, view^{P(w^1_x)}_{V^*(z)}(x)\}_{x\in L,\, z\in\{0,1\}^*}$ and $\{x, view^{P(w^2_x)}_{V^*(z)}(x)\}_{x\in L,\, z\in\{0,1\}^*}$. Namely, for every PPT non-uniform distinguishing algorithm D, every polynomial $p(\cdot)$, all sufficiently long $x \in L$, and all $z \in \{0,1\}^*$, it holds that*

$$|\Pr[D(x, z, view^{P(w^1_x)}_{V^*(z)}(x) = 1] - \Pr[D(x, z, view^{P(w^2_x)}_{V^*(z)}(x) = 1]| < \frac{1}{p(|x|)}$$

We say that $\langle P, V\rangle$ is sub-exponentially strong *witness indistinguishable for R_L, if for some c, $0 < c < 1$, for every sufficiently long $x \in L$ of length n, for every distinguishing circuit D of size at most 2^{n^c}, and every $z \in \{0,1\}^*$, it holds that*

$$|\Pr[D(x, z, view^{P(w^1_x)}_{V^*(z)}(x) = 1] - \Pr[D(x, z, view^{P(w^2_x)}_{V^*(z)}(x) = 1]| < 2^{-n^c}$$

Definition 2.7 (system for proof of knowledge [3, 23]) *Let R be a binary relation and $\kappa : N \to [0, 1]$. We say that a probabilistic polynomial-time (PPT) interactive machine V is a knowledge verifier for the relation R with knowledge error κ if the following two conditions hold:*

- *Non-triviality: There exists an interactive machine P such that for every $(x, w) \in R$ all possible interactions of V with P on common input x and auxiliary input w are accepting.*

- *Validity (with error κ): There exists a polynomial $q(\cdot)$ and a probabilistic oracle machine K such that for every interactive machine P^*, every $x \in L_R$, and every $w, r \in \{0,1\}^*$, machine K satisfies the following condition:*

  Denote by $p(x, w, r)$ the probability that the interactive machine V accepts, on input x, when interacting with the prover specified by $P^*_{x,w,r}$ (where $P^*_{x,w,r}$ denotes the strategy of P^* on common input x, auxiliary input w and random-tape r). If $p(x, w, r) > \kappa(|x|)$, then, on input x and with oracle access to $P^*_{x,w,r}$, machine K outputs a solution $w' \in R(x)$ within an expected number of steps bounded by

  $$\frac{q(|x|)}{p(x, w, r) - \kappa(|x|)}$$

  The oracle machine K is called a knowledge extractor.

An interactive proof system $\langle P, V \rangle$ such that V is a knowledge verifier for a relation R and P is a machine satisfying the non-triviality condition (with respect to V and R) is called a system for proof of knowledge for the relation R.

3 Definitions: rZK and Concurrent Soundness in the BPK Model

In this section we give the formal definitions of resettable zero-knowledge and concurrent soundness in the BPK model.

Honest players in the BPK model.

The BPK model consists of the following:

- F be a public-key file that is 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.

- $P(1^n, x, w, F, id, \gamma)$ be an honest prover that is a *polynomial-time* interactive machine, where 1^n is a security parameter, x is an n-bit string in L, w is an auxiliary input, F is a public-file, id is a verifier identity, and γ is its random-tape.

- V be an honest verifier that is an polynomial-time interactive machine working in two stages.

  1. Key generation stage. V, on a security parameter 1^n and a random-tape r, outputs a public-key PK and remembers the corresponding secret key SK.
  2. Verification stage. V, on inputs SK, $x \in \{0,1\}^n$ and a random tape ρ, performs an interactive protocol with a prover and outputs "accept x" or "reject x".

The malicious resetting verifier and resettable zero-knowledge.

A malicious s-resetting malicious verifier V^*, where s is a positive polynomial, is a PPT Turing machine working in two stages so that, on input 1^n,

Stage-1. V^* receives $s(n)$ *distinct* strings $x_1, \cdots, x_{s(n)}$ of length n each, and outputs an arbitrary public-file F and a list of (without loss of generality) $s(n)$ identities $id_1, \cdots, id_{s(n)}$.

Stage-2. Starting from the final configuration of Stage-1, $s(n)$ random tapes, $\gamma_1, \cdots, \gamma_{s(n)}$, are randomly selected and then fixed for P, resulting in $s(n)^3$ deterministic prover strategies $P(x_i, id_j, \gamma_k)$, $1 \leq i, j, k \leq s(n)$. V^* is then given oracle access to these $s(n)^3$ provers, and finally outputs its "view" of the interactions (i.e., its random tapes and messages received from all its oracles).

Definition 3.1 (black-box resettable zero-knowledge) *A protocol $\langle P, V \rangle$ is black-box resettable zero-knowledge for a language $L \in \mathcal{NP}$ if there exists a PPT black-box simulator S such that for every s-resetting verifier V^*, the following two probability distributions are indistinguishable. Let each distribution be indexed by a sequence of* **distinct** *common inputs $\bar{x} = x_1, \cdots, x_{s(n)} \in L \cap \{0,1\}^n$ and their corresponding NP-witnesses $aux(\bar{x}) = w_1, \cdots, w_{s(n)}$:*

Distribution 1. *The output of V^* obtained from the experiment of choosing $\gamma_1, \cdots, \gamma_{s(n)}$ 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: $P(x_i, w_i, F, id_j, \gamma_k)$ for $1 \leq i, j, k \leq s(n)$. Note that V^* can oracle access to these $s(n)^3$ instances of P.*

Distribution 2. *The output of $S(x_1, \cdots, x_{s(n)})$.*

Remark. In Distribution 1 above, since V^* oracle accesses to $s(n)^3$ instances of P: $P(x_i, w_i, F, id_j, \gamma_k)$, $1 \leq i, j, k \leq s(n)$, it means that V^* may invoke and interact with the same $P(x_i, w_i, F, id_j, \gamma_k)$ multiple times, where each such interaction is called a session. We remark that there are two versions for V^* to work in Distribution 1.

1. Sequential version. In this version, a session must be terminated (either completed or aborted) before V^* initiating a new session. That is, V^* is required to terminate its current interaction with the current oracle $P(x_i, w_i, F, id_j, \gamma_k)$ before starting an interaction with any $P(x_{i'}, w_{i'}, F, id_{j'}, \gamma_{k'})$, regardless of $(i, j, k) = (i', j', k')$ or not. Thus, the activity of V^* proceeds in rounds. In each round it selects one of its oracles and conducts a complete interaction with it.

2. Interleaving version. In this version the above restriction is removed and so V^* may initiate and interact, controlling the schedule of messages being exchanged, with $P(x_i, w_i, F, id_j, \gamma_k)$'s concurrently in many sessions.

However, these two versions are equivalent as shown in [8]. In other words, interleaving interactions do not help the malicious verifier get more advantages on learning knowledge from its oracles than it can do by sequential interactions. Without loss of generality, in the rest of this paper we assume the resetting malicious verifier V^* works in the sequential version.

Definition 3.2 (resettable witness indistinguishability rWI) *A protocol $\langle P, V \rangle$ is said to be resettable witness indistinguishable for an $L \in \mathcal{NP}$ if for every positive polynomial s, for every s-resetting malicious verifier V^*, two distribution ensembles of Distribution 1 (defined in Definition 3.1), which are indexed by the same $\bar{x}$ but possibly different sequences of prover's $\mathcal{NP}$-witnesses: $aux^{(1)}(\bar{x}) = w_1^{(1)}, \cdots, w_{s(n)}^{(1)}$ and $aux^{(2)}(\bar{x}) = w_1^{(2)}, \cdots, w_{s(n)}^{(2)}$, are computationally indistinguishable.*

In [8] Canetti et al. first gave a 4-round rWI *proof* for $\mathcal{NP}$. The round-complexity is dramatically reduced to 2 by Dwork and Naor [17], by a slight modification of zaps that are themselves 2-round public-coin WI *proofs* for $\mathcal{NP}$. An interesting property of zaps is that its first-round message (for NIZK-based zap, it is a random string sent from the verifier to the prover) can be fixed once and for all [17]. It means that one can post the first-round message of a zap as a part of its public-key in the public-key model. We will use this property in our constructions later. As shown in [17], zaps can be constructed in several ways. In particular, they can be based on any (single-theorem) NIZK proof for $\mathcal{NP}$.

Concurrent soundness in the BPK model

For an honest verifier V with public-key PK and secret-key SK, an s-concurrent malicious prover P^* in the BPK model, for a positive polynomial s, is a probabilistic polynomial-time Turing machine that, on a security parameter 1^n and PK, performs concurrently at most $s(n)$ interactive protocols (sessions) with V as follows.

If P^* is already running $i-1$ ($1 \leq i-1 \leq s(n)$) sessions, it can select *on the fly* a common input $x_i \in \{0,1\}^n$ (which may be equal to x_j for $1 \leq j < i$) and initiate a new session with the verification stage of $V(SK, x_i, \rho_i)$. We stress that in different sessions V uses independent random-tapes in its verification stage (that is, $\rho_1, \cdots, \rho_{s(n)}$ are independent random strings).

We then say a protocol satisfies *concurrent soundness* in the BPK model if for any honest verifier V, for all positive polynomials s, for all s-concurrent malicious prover P^*, the probability that there exists an i ($1 \leq i \leq s(n)$) such that $V(SK, x_i, \rho_i)$ outputs "accept x_i" while $x_i \notin L$ is negligible in n.

4 The Generic yet Practical Transformation from any Σ-Protocol

In this section, we first recall the critical tools used for the generic yet practical transformation without $\mathcal{NP}$-reductions, and then present the generic yet practical transformation transforming Σ-protocols to rZK-CS protocols in the BPK model.

4.1 Tools used

The generic yet practical transformation involves, as critical tools, DLP and RSA based practical trapdoor commitment schemes with some additional properties and a novel use of Σ_{OR}-protocols.

4.1.1 Σ-protocols and Σ_{OR}-protocols

The idea of Σ-protocols as an abstract concept is introduced by Cramer in [11]. Informally, a Σ-protocol is itself a 3-round public-coin *special* honest verifier zero-knowledge (SHVZK) protocol with special soundness in the knowledge-extraction sense. Σ-protocols have been proved to be a very powerful cryptographic tool and are widely used in numerous important cryptographic applications including digital signatures, identification schemes, efficient electronic payment and voting systems. We remark that a very large number of Σ-protocols have been developed in the literature (mainly in applied cryptography), and Σ-protocol examples for DLP and RSA are given in below. For a good survey of Σ-protocols and their applications, readers are referred to [14, 12].

Definition 4.1 *A 3-round public-coin protocol $\langle P, V \rangle$ is said to be a Σ-protocol for a relation R if the following hold:*

- *Completeness. If P, V follow the protocol, the verifier always accepts.*
- *Special soundness. From* any *common input x of length n and any pair of accepting conversations on input x, (a, e, z) and (a, e', z') where $e \neq e'$, one can efficiently compute w such that $(x, w) \in R$. Here a, e, z stand for the first, the second and the third message respectively and e is assumed to be a string of length t (that is polynomially related to n) selected uniformly at random in $\{0,1\}^t$.*
- *Special honest verifier zero-knowledge (SHVZK). There exists a probabilistic polynomial-time (PPT) simulator S, which on input x and a random challenge string e, outputs an accepting conversation of the form (a, e, z), with the same probability distribution as the real conversation between the honest P, V on input x.*

Σ-protocol for DLP: The following is a Σ-protocol $\langle P, V\rangle$ proposed by Schnorr [36] for proving the knowledge of discrete logarithm, w, for a common input of the form (p, q, g, h) such that $h = g^w \ mod \ p$, where on a security parameter n, p is a uniformly selected n-bit prime such that $q = (p-1)/2$ is also a prime, g is an element in $\mathbf{Z}_p^*$ of order q. It is also actually the first efficient Σ-protocol proposed in the literature.

- P chooses r at random in $\mathbf{Z}_q$ and sends $a = g^r \ mod \ p$ to V.
- V chooses a challenge e at random in $\mathbf{Z}_{2^t}$ and sends it to P. Here, t is fixed such that $2^t < q$.
- P sends $z = r + ew \ mod \ q$ to V, who checks that $g^z = ah^e \ mod \ p$, that p, q are primes and that g, h have order q, and accepts iff this is the case.

Σ-protocol for RSA: Let n be an RSA modulus and q be a prime. Assume we are given some element $y \in Z_n^*$, and P knows an element w such that $w^q = y \mod n$. The following protocol is a Σ-protocol (proposed in [28]) for proving the knowledge of q-th roots modulo n .

- P chooses r at random in Z_n^* and sends $a = r^q \mod n$ to V.
- V chooses a challenge e at random in Z_{2^t} and sends it to P. Here, t is fixed such that $2^t < q$.
- P sends $z = rw^e \mod n$ to V, who checks that $z^q = ay^e \mod n$, that q is a prime, that $gcd(a, n) = gcd(y, n) = 1$, and accepts iff this is the case.

The OR-proof of Σ-protocols [13].

One basic construction with Σ-protocols allows a prover to show that given two inputs x_0, x_1, it knows a w such that either $(x_0, w) \in R_0$ or $(x_1, w) \in R_1$, but without revealing which is the case. Specifically, given two Σ-protocols $\langle P_b, V_b\rangle$ for R_b, $b \in \{0, 1\}$, with random challenges of, without loss of generality, the same length t, consider the following protocol $\langle P, V\rangle$, which we call Σ_{OR}. The common input of $\langle P, V\rangle$ is (x_0, x_1) and P has a private input w such that $(x_b, w) \in R_b$.

- P computes the first message a_b in $\langle P_b, V_b\rangle$, using x_b, w as private inputs. P chooses e_{1-b} at random, runs the SHVZK simulator of $\langle P_{1-b}, V_{1-b}\rangle$ on input (x_{1-b}, e_{1-b}), and let $(a_{1-b}, e_{1-b}, z_{1-b})$ be the output. P finally sends a_0, a_1 to V.
- V chooses a random t-bit string e and sends it to P.
- P sets $e_b = e \oplus e_{1-b}$ and computes the answer z_b to challenge e_b using (x_b, a_b, e_b, w) as input. It sends (e_0, z_0, e_1, z_1) to V.
- V checks that $e = e_0 \oplus e_1$ and that conversations (a_0, e_0, z_o), (a_1, e_1, z_1) are accepting conversations with respect to inputs x_0, x_1, respectively.

Theorem 4.1 *[13] The protocol Σ_{OR} above is a Σ-protocol for R_{OR}, where $R_{OR} = \{((x_0, x_1), w) | (x_0, w) \in R_0$ or $(x_1, w) \in R_1\}$. Moreover, for any malicious verifier V^*, the probability distribution of conversations between P and V^*, where w is such that $(x_b, w) \in R_b$, is independent of b. That is, Σ_{OR} is perfectly witness indistinguishable proof of knowledge for R_{OR} (the POK property is due to that Σ_{OR}-protocol is itself Σ-protocol).*

4.1.2 Trapdoor commitment TC schemes with additional properties

We recall some perfectly-hiding trapdoor commitment schemes and clarify some additional properties about them which are critical for our purpose.

As shown in Definition 2.5, known trapdoor commitment schemes work in two rounds as follows: In the first round, the commitment receiver generates and sends the $TCPK$ to the commitment sender. In the second round, on $TCPK$ and the value v to be committed, the sender computes $(c, d) \leftarrow TCCom(TCPK, v)$ and sends c as the commitment, while keeping the value v and the decommitment information d in private. The trapdoor is the corresponding $TCSK$ with which one can equivocate the commitment at its wish. But, for our purpose, we need TC schemes that satisfy the following additional requirements:

1. Public-key verifiability. The validity of $TCPK$ (even generated by a malicious commitment receiver) can be efficiently verified. In particular, given any $TCPK$, one can efficiently verify whether or not $TCSK$ exists.

2. Public-key Σ-provability. On common input $TCPK$ and private input $TCSK$, one can prove, by Σ-protocols, the knowledge of $TCSK$.

We call a trapdoor commitment scheme satisfying the above two additional properties a *verifiable and Σ-provable trapdoor commitment* (VPTC, in short) scheme. The first round of a VPTC scheme is denoted by $VPTCPK$ and the corresponding trapdoor is denoted by $VPTCSK$. We note both the DLP-based [7] and the RSA-based [35] perfectly-hiding trapdoor commitment schemes are VPTC schemes with perfect hiding and trapdoorness properties.

Consider the DLP-based perfectly-hiding trapdoor commitment scheme [7]: On a security parameter n, the receiver selects uniformly an n-bit prime p so that $q = (p-1)/2$ is a prime, an element g of order q in $\mathbf{Z}_p^*$. Then the receiver uniformly selects w in $\mathbf{Z}_q$ and sets $h = g^w \ mod \ p$. The receiver sets $TCPK = (p, q, g, h)$ and keeps w as $TCSK$ in secret. To commit a value v in $\mathbf{Z}_q$, the sender first checks that: p, q are primes, $p = 2q + 1$ and g, h are elements of order q in $\mathbf{Z}_p^*$. This guarantees the public-key verifiability property above, and in particular, guarantees that there exists a w such that $h = g^w \ mod \ p$ as there is a unique subgroup of order q in $\mathbf{Z}_p^*$. If the above checking is failed, then the sender halts announcing that the receiver is cheating. Otherwise (i.e., the above testing is successful), then the sender uniformly selects $d \in \mathbf{Z}_q$ (the decommitment information), and sends $c = g^d h^v \mod p$ as its commitment. The public-key Σ-provability is direct from the above Σ-protocol for DLP.

Now, consider the RSA-based perfectly-hiding trapdoor commitment scheme [35]: Let n be a composite number and $q > n$ be a prime number, the receiver randomly chooses w from Z_n^* and computes $y = w^q \mod n$. The $TCPK$ is set to be (n, q, y) and $TCSK = w$. To commit a value $v \in Z_q$, the sender firstly checks that: n is a composite number, $q > n$ is a prime number and y is in Z_n^* (this in particular guarantees the existence of w). If the above checking is successful, then the sender randomly chooses $d \in Z_n^*$ and computes $c = y^v d^q$ as its commitment. The public-key Σ-provability is direct from the above Σ-protocol for RSA.

4.2 The generic yet practical transformation

We next present the generic yet practical transformation (without $\mathcal{NP}$-reductions) transforming Σ-protocols to rZK-CS protocols in the BPK model. Recall that this is the first protocol of this efficiency nature.

Let L be any language that admits Σ-protocols (e.g., the language of 2^n residues or the language of a generated subgroup), f_V be any OWF that admits Σ-protocols, and VPTC be a verifiable Σ-provable trapdoor commitment scheme with $VPTCPK$ as its first round and $VPTCSK$ as the corresponding

Key generation. On a security parameter N, each honest verifier V randomly selects an element x_V of length N, computes $y_V = f_V(x_V)$, publishes y_V as its public-key PK while keeping x_V as its secret-key SK.

Common input. The public-file F, an element $x \in L \cap \{0,1\}^n$ and index i that specifies the i-th entry of F, PK_i. Note that the system security parameter is n.

P **private input.** An $\mathcal{NP}$-witness *wit* for $x \in L$, a pair of random strings (γ_1, γ_2), where γ_1 is a $poly(n)$-bit string that is used to generate $VPTCPK$ and γ_2 is an n-bit string serving as the randomness seed of a PRF.

V **private input.** Private key SK_i such that $PK_i = f_V(SK_i)$.

Complexity-leverage used. Let c_1, $0 < c_1 < 1$, be the constant that the one-wayness of the OWF f_V (used by V) holds against any circuit of size $2^{N^{c_1}}$. Let c_2 be the constant that: for all sufficiently large n's, both the length of the witness *wit* for $x \in L \cap \{0,1\}^n$ and the length of $VPTCSK$ are bounded by n^{c_2}. Then we set $\epsilon > c_2/c_1$ and $N = n^{\epsilon}$. This ensures that one can enumerate all potential witnesses *wit*, or all potential $VPTCSK$s in time $2^{n^{c_2}}$, which is still lesser than the time it would take to break the one-wayness of f_V (because $2^{n^{c_2}} < 2^{N^{c_1}}$).

Phase-1. Phase-1 consists of two stages:

Stage-1. P generates $VPTCPK$ on security parameter n by using the randomness γ_1, and sends $VPTCPK$ to V. *Note that the $VPTCPK$ can be fixed once and for all.*

Stage-2. V first checks the validity of $VPTCPK$ received and aborts if it is not valid (this is guaranteed by the public-key verifiability of the underlying VPTC scheme). Otherwise (i.e., $VPTCPK$ is valid), V randomly chooses a random string e_V of length t and computes $c_V = VPTCCom(VPTCPK, e_V)$ (that is, V commits e_V using the underlying VPTC scheme). V sends c_V to P, and proves to P that it knows either the preimage of PK_i (i.e., SK_i) or $VPTCSK$, by executing the Σ_{OR}-protocol on $(PK_i, VPTCPK)$ in which V plays the role of knowledge prover and P plays the role of knowledge verifier. Denote by a_V the first-round message of this Σ_{OR}-protocol, then all randomness used by P (from then on after receiving (c_V, a_V)) in the remaining computation is got by applying $PRF(\gamma_2, \cdot)$ on the "determining" message $(x, F, c_V, a_V, PK_i, i)$ *that "determines" the behaviors of even malicious verifier in the following interactions (i.e., either abort or providing valid decommitment information).* If V successfully finishes the Σ_{OR}-protocol and P accepts, then goto Phase-2. Otherwise, P aborts.

Phase-2. P proves to V that it knows either the witness *wit* for $x \in L$ or the preimage of PK_i, by executing the Σ_{OR}-protocol on (x, PK_i) in which V sends the second-round message (i.e., the assumed random challenge from V) by just revealing e_V committed to c_V. We also denote by a_P, z_P the first-round message and the third-round message of this Σ_{OR}-protocol, respectively.

Figure 1. Generic yet practical rZK-CS arguments for any language that admits Σ-protocols

trapdoor. Each (honest) verifier randomly selects x_V from the domain of f_V and publishes $y_V = f_V(x_V)$ as its public-key. On a common input $x \in L$, the following is the high-level overview of the transformation that works in two phases: In the first phase, the prover P first sends $VPTCPK$ to the verifier V, and (if $VPTCPK$ is valid) then V proves to P that it knows either $VPTCSK$ or the preimage of y_V, by executing Σ_{OR} on $(VPTCPK, y_V)$. In the second phase, if P accepts the above Σ_{OR}-proof (from V to P), then P proves to V that it knows either the witness of $x \in L$ or the preimage of y_V, by executing Σ_{OR} on (x, y_V) in which the second round (i.e., the random challenge from V to P) is denoted by e_V. Finally, to make the transformed protocol resettable we require that: V first commits e_V on the top of the above transformation using the underlying VPTC scheme, P and V use different security parameters such that breaking $VPTCPK$ does not compromise the security of y_V, and the randomness of P is generated by applying a PRF on some partial transcript. The details are given in Figure 1 (page 13).

The protocol depicted in Figure-1 runs in 7 rounds, but it can be easily combined into a 5-round protocol in which P sends $VPTCPK$ in the first-round (*note that the first-round message, i.e. $VPTCPK$, can be fixed once and for all*), V sends (c_V, a_V) in the second-round and then the remaining interactions

are combined into the other three rounds. Note that the transformation does not go through general $\mathcal{NP}$-reductions, and if the underlying VPTC, PRF and the starting Σ-protocol for L are practical, then the transformed protocol is also a practical protocol for L. When instantiated with the DL function for the underlying VPTC and the starting Σ-protocol, together with the Naor-Reingold practical PRF, the transformed protocol (for the language DL) goes through about 12 exponentiations at the rZK prover side and 9 exponentiations at the verifier side.

Theorem 4.2 *Assuming the existence of PRF, VPTC with perfect hiding and trapdoorness properties, and OWF that admits Σ-protocols and is secure against subexponentially-strong adversaries, the protocol depicted in Figure-1 is a 5-round rZK-CS argument without $\mathcal{NP}$-reductions in the BPK model for any language that admits Σ-protocols.*

Proof (sketch).

Black-box resettable zero-knowledge.

For any s-resetting adversary V^* (as defined in Section 3), first of all, without loss of generality we make the following two simplifying assumptions. Firstly, we assume V^* works in the sequential version (which is equivalent to the interleaving version as discussed in Section 3). Secondly, we assume the real prover uses a truly random function (rather than a PRF), as the proof can be easily extended, by standard hybrid technique, to the case when the real prover uses a PRF.

For any s-resetting adversary V^* who receives $s(n)$ *distinct* strings $x_1, \cdots, x_{s(n)}$ of length n each, and outputs an arbitrary public-file F containing $s(n)$ entries $PK_1, \cdots, PK_{s(n)}$ in its first stage, we say a public-key PK_i in F, $1 \leq i \leq s(n)$, is "covered" if the rZK simulator S has already learnt (extracted) the corresponding secret-key SK_i (if such exists). The rZK simulator S works in $s(n)+1$ phases such that in each phase it either successfully finishes its simulation or "covers" a new public-key in F. In each phase, S runs V^* as a subroutine and works session by session sequentially.

In each session j with a *distinct* "determining" message $(x_k, F, c_V^{(j)}, a_V^{(j)}, PK_i, i)$, $1 \leq i, j, k \leq s(n)$, and with respect to a "*covered*" public-key PK_i, then S uses independent random coins in its remaining computation after that (note that the $VPTCPK$ in the first-round can be fixed once and for all), and uses the (assumed known) secret-key SK_i as its witness in the Σ_{OR}-protocol of Phase-2. For any session with a "determining" message that is identical to that of some previous session, then S just copies what sent in that previous session.

The difficulty occurs in simulating the Phase-2 of a session j with "determining" message $(x_k, F, c_V^{(j)}, a_V^{(j)}, PK_i, i)$, $1 \leq i, j, k \leq s(n)$, but with respect to an *uncovered* public-key PK_i, as S has not yet learned the secret-key SK_i and also S has no the $\mathcal{NP}$-witness of the common input x_k (possessed by the real prover). The approach for S is to extract the corresponding secret-key SK_i in polynomial time (if such exists) and then go to the next phase where it simulates the simulation from scratch again but with one more covered public-key. Recall that although the Σ_{OR} in Stage-2 of Phase-1 is with respect to the common input $(VPTCPK, PK_i)$, but the whole interactions of Phase-1 constitute an *argument of knowledge of the secret-key SK_i* (as $VPTCPK$ is sent by the honest prover). By the proof of knowledge property (i.e., special soundness) of Σ_{OR}, to extract the corresponding SK_i, S needs to send a new random challenge (i.e., the second-round message of the Σ_{OR} of Phase-1). But the problem here is that such random challenge may have been fixed (determined) by a (possibly) past partial transcript (in which the determining message $(x_k, F, c_V^{(j)}, a_V^{(j)}, PK_i, i)$ appeared at the first time) due to the underlying random function used. This problem is bypassed precisely as in [8, 1]. That is, S rewinds V^* to the point that V^* just sent $(x_k, F, c_V^{(j)}, a_V^{(j)}, PK_i, i)$ *at the first time*, and sends back a new random challenge by using a new random function.

The same analysis in [8, 1] shows that with overwhelming probabilities the rZK simulator S will finish its simulation and output a simulated transcript in expected polynomial-time. The indistinguishability between the simulated transcript and the real interaction transcript is from the (perfect) WI property of

the Σ_{OR} of Phase-2. Actually, if we remove the Σ_{OR}-protocol of Phase-1, then the remained interactions constitute a constant-round rWI proof on common input (x, PK_i), as shown by the general paradigm of [8] for achieving rWI from admissible systems. (Readers are referred to [8, 1] for more details of the general paradigm.)

Comments: Note that in the proof of rZK, we require nothing about the public-keys registered by V^* in F. What we need in the simulation is the special soundness of the Σ_{OR}-protocol of Phase-1 that holds with respect to *any* common input (in particular, *any* public-key registered by V^*, whether valid or not). That is, our protocol is with what we informally call real bare public-keys.

Concurrent soundness.

Suppose the transformed protocol depicted in Figure-1 does not satisfy concurrent soundness in the BPK model, then according to the definition of concurrent soundness in the BPK model (described in Section 3), there exists an s-concurrent malicious prover P^* such that in a concurrent attack issued by P^* against an honest verifier V with public-key y_V, with non-negligible probability $q(n)$ there exists a j, $1 \leqslant j \leqslant s(n)$, such that V outputs "accept x_j" in session j while $x_j \notin L$. Then we will construct an algorithm E that takes the public-key y_V as input and outputs the corresponding secret-key x_V with non-negligible probability $\frac{(q(n))^2}{s(n)}$ in time $poly(n) \cdot 2^{n^{c_2}} < 2^{N^{c_1}}$, which breaks the hardness assumption on the OWF f_V used by V.

E randomly chooses j from $\{1, \cdots, s(n)\}$ and runs P^* as a subroutine by playing the role of the honest verifier with public-key y_V. In each session, after receiving a $VPTCPK$, E checks its validity and then extracts the corresponding $VPTCSK$ by brute-force searching in time $2^{n^{c_2}}$ (in case the $VPTCPK$ is valid), then sends a VPTC commitment to 0^t (rather than to a random string of length t as the honest verifier does), uses $VPTCSK$ as the witness in the Σ_{OR}-protocol of Phase-1 of that session, and decommits the VPTC commitment to a random string of length t by using the trapdoor $VPTCSK$ in Phase-2 of that session. In the j-th session with respect to the common input x_j selected by P^* on the fly, denote by $c_V^{(j)}$ the VPTC commitment (that commits to 0^t) sent by E in the second round, then in the Phase-2 of the j-th session after receiving $a_{P^*}^{(j)}$ (the first-round message of Phase-2 of the j-th session), E decommits $c_V^{(j)}$ to a random string $e_V^{(j)}$ of length t. Whenever E receives a valid $z_{P^*}^{(j)}$ such that $(a_{P^*}^{(j)}, e_V^{(j)}, z_{P^*}^{(j)})$ is an accepting conversation of the Σ_{OR} of Phase-2 on (x_j, y_V), E rewinds P^* to the point that it just sent $a_{P^*}^{(j)}$, decommits $c_V^{(j)}$ to a new random string $e_V^{(j)\prime} \neq e_V^{(j)}$ (by using the trapdoor $VPTCSK$), and runs P^* further. Whenever E receives again a valid $z_{P^*}^{(j)\prime}$ and $(a_{P^*}^{(j)}, e_V^{(j)\prime}, z_{P^*}^{(j)\prime})$ is an accepting conversation on (x_j, y_V), then E stops.

Due to the public-key verifiability, the perfect hiding and trapdoorness properties of the underlying VPTC scheme, and the perfect WI property of the Σ_{OR} of Phase-1, with the same probability $q(n)$ there exists a j, $1 \leq j \leq s(n)$, such that P^* can convince E of a false $x_j \notin L$ in the j-th session, *even E sends VPTC commitments to 0^t (rather than to random strings of length t) and uses the extracted $VPTCSK$ (rather than x_V as used by the honest verifier) as its witness in the Σ_{OR} of Phase-1.* Conditioned on E correctly guessed the value j, then with probability $(q(n))^2$ E will extract either the witness *wit* for $x_j \in L$ or the secret-key x_V such that $y_V = f_V(x_V)$, which is guaranteed by the special soundness of the Σ_{OR} of Phase-2. As we assume $x_j \notin L$ and E randomly guesses j from $\{1, \cdots, s(n)\}$, we conclude E outputs x_V with probability $\frac{(q(n))^2}{s(n)}$. Note that E works in $poly(n) \cdot 2^{n^{c_2}} < 2^{N^{c_1}}$ time, which violates the hardness assumption on the OWF f_V used by V.

Comments: The public-key verifiability property of the underlying VPTC scheme is necessary for the above concurrent soundness proof, as otherwise, the simulation of E will be distinguishable from the real interactions between P^* and the instances of the honest verifier. Specifically, for a maliciously-formed $VPTCPK$ if it is hard to verify in polynomial time whether or not the corresponding secret-key exits. Then, for a maliciously formed $VPTCPK$ such that there exists no corresponding secret-key at all, in real interactions the honest verifier still always finishes its interactions with P^* with respect to

such "bad" $VPTCPK$. But, in the simulation of E, E cannot extracts the secret-key corresponding to $VPTCPK$ and thus cannot finish the simulation, which is clearly distinguishable from the *finished* interactions between P^* and the real honest verifier.

We remark that the complexity leveraging technique plays a critical role in the above proof of concurrent soundness. Note that when E rewinds the concurrent malicious P^* in the above proof, E itself is also rewound as P^* is concurrently interacting with it, from which the witness (i.e., $VPTCSK$) used by E in the Σ_{OR} of Phase-1 may be exposed. This is just the place the underlying complexity leveraging plays its role, which says that breaking (or exposing) $VPTCPK$ does not compromise the security of the OWF f_V used by the verifier. Actually, we do not know how to prove concurrent soundness under standard polynomial hardness assumptions without complexity leveraging. □

5 Constant-Round rZK-CS Arguments for $\mathcal{NP}$ with Preimage-Verifiable One-Way Functions

From this section on, we show how to achieve constant-round rZK-CS arguments in the BPK model for any language in $\mathcal{NP}$, with $\mathcal{NP}$-reductions but with much weaker hardness assumptions and reduced (optimal) round-complexity. Firstly, in this section, we achieve constant-round rZK-CS arguments for $\mathcal{NP}$ in the BPK model by employing preimage-verifiable one-way functions.

5.1 5-round rZK-CS arguments for $\mathcal{NP}$ with public-keys secure under any preimage-verifiable one-way function

First note that preimage-verifiable OWF is a generic and much weaker hardness assumption that includes, in particular, any certified one-way permutation and any 1-1 length-preserving OWF. Before describing the construction, we first recall some generic tools used.

5.1.1 Generic tools used

Perfectly-binding commitments. One-round perfectly-binding (computationally-hiding) commitments can be constructed based on any one-way permutation OWP [5, 26]. Loosely speaking, given a OWP f with a hard-core predict b (cf, [23]), on a security parameter N one commits a bit σ by uniformly selecting $x \in \{0,1\}^N$ and sending $(f(x), b(x) \oplus \sigma)$ as a commitment, while keeping x as the decommitment information.

Perfectly-binding commitments can also be constructed based on any one-way function but run in two rounds [33]. On a security parameter N, let $PRG : \{0,1\}^N \longrightarrow \{0,1\}^{3N}$ be a pseudorandom generator, the Naor's OWF-based two-round public-coin perfectly-binding commitment scheme works as follows: In the first round, the commitment receiver sends a random string $R \in \{0,1\}^{3N}$ to the committer. In the second round, the committer uniformly selects a string $s \in \{0,1\}^N$ at first; then to commit a bit 0 the committer sends $PRG(s)$ as the commitment; to commit a bit 1 the committer sends $PRG(s) \oplus R$ as the commitment. Note that the first-round message of Naor's commitment scheme can be fixed once and for all and, in particular, can be posted as a part of public-key in the public-key setting.

For the above perfectly-binding commitment schemes, we remark that if the underlying OWP or OWF are secure against $2^{N^{c_1}}$-time adversaries for some constant $c_1, 0 < c_1 < 1$ on a security parameter N, then the hiding property of corresponding perfectly-binding commitment schemes above also holds against $2^{N^{c_1}}$-time adversaries.

Blum's protocol for HC [6]. The n-parallel repetitions of Blum's basic protocol for proving the knowledge of Hamiltonian cycle on a given directed graph G [6] is just a 3-round public-coin WIPOK for $\mathcal{NP}$ (with knowledge error 2^{-n}) under any one-way permutation (as the first round of it involves

one-round perfectly-binding commitments of a random permutation of G). But it can be easily modified into a 4-round public-coin WIPOK for $\mathcal{NP}$ under any OWF by employing Naor's two-round (public-coin) perfectly-binding commitment scheme [33]. The description of Blum's protocol for HC is given in Appendix A.

We remark that the WI property of Blum's protocol for HC relies on the hiding property of the underlying perfectly-binding commitment scheme (used in its first-round). If the hiding property of the underlying perfectly-binding commitment scheme is secure against $2^{N^{c_1}}$-time adversaries for some constant $c_1, 0 < c_1 < 1$ on a security parameter N, then the WI property of Blum's protocol also holds against $2^{N^{c_1}}$-time adversaries.

Feige-Shamir two-round trapdoor commitments [20]. Based on Blum's protocol for HC, Feige and Shamir developed a generic two-round (computationally-hiding and computationally-binding) trapdoor commitment scheme [20], under either any one-way permutation or any OWF (depending on the underlying perfectly-binding commitment scheme used). The $TCPK$ of the FSTC scheme (i.e., its first-round message) is $(y = f(x), G)$ (for OWF-based solution, the first-round also includes a random string R serving as the first-round message of Naor's OWF-based perfectly-binding commitment scheme), where f is a OWF and G is a graph that is reduced from y by the Cook-Levin $\mathcal{NP}$-reduction. The corresponding trapdoor is x (or equivalently, a Hamiltonian cycle in G). The following is the description of the Feige-Shamir trapdoor commitment (FSTC) scheme, in which, for our purpose, we have assumed the commitment receiver and the committer use different security parameters n and N, respectively.

Round-1. Let f be a OWF, the commitment receiver randomly selects an element x of length n in the domain of f, computes $y = f(x)$, reduces y (by Cook-Levin $\mathcal{NP}$-reduction) to an instance of HC, a graph $G = (V, E)$ with $q = |V|$ nodes, such that finding a Hamiltonian cycle in G is equivalent to finding the preimage of y. Finally, it sends (y, G) to the committer. We remark that to get OWF-based trapdoor commitments, the commitment receiver also sends a random string R of length $3N$, where N is the security parameter used by the committer.

Round-2. The committer first checks the $\mathcal{NP}$-reduction from y to G and aborts if G is not reduced from y. Otherwise, to commit 0, the committer selects a random permutation, π, of the vertices V, and commits (using the underlying perfectly-binding commitment scheme) the entries of the adjacency matrix of the resulting permutated graph. That is, it sends an q-by-q matrix of commitments so that the $(\pi(i), \pi(j))^{th}$ entry is a commitment to 1 if $(i, j) \in E$, and is a commitment to 0 otherwise; To commit 1, the committer commits an adjacency matrix containing a randomly labeled q-cycle only.

Decommitment stage. To decommit to 0, the committer sends π to the commitment receiver along with the revealing of all commitments, and the receiver checks that the revealed graph is indeed isomorphic to G via π; To decommit to 1, the committer only opens the entries of the adjacency matrix that are corresponding to the randomly labeled cycle, and the receiver checks that all revealed values are 1 and the corresponding entries form a simple q-cycle.

The (computational) trapdoorness property of the FSTC scheme is: After sending a commitment to 0 (which is indistinguishable from a commitment to 1), one can decommit to 0 in the normal way. However, it is also possible to decommit it to 1 if one knows a Hamiltonian cycle in G. Furthermore, the distribution of a commitment to 0 *together with the "trapdoor-assistant" decommitment information to 1* is indistinguishable from the distribution of a commitment to 1 *together with the "real" decommitment information to 1* (due to the hiding property of the underlying perfectly-binding commitment scheme). This implies, by standard hybrid technique, that the distribution of commitments to 0^n together with "trapdoor-assistant" decommitment information to a random string $\hat{e}_V$ of length n is indistinguishable

from the distribution of commitments to a random sting e_V of length n together with the "real" decommitment information to e_V (we will use this property in the proof of concurrent soundness below). *Again, if the hiding property of the underlying perfectly-binding commitment scheme is secure against subexponential-time adversaries, then both the hiding property and the trapdoorness property of the FSTC scheme hold also against subexponential-time adversaries.*

5.1.2 The protocol

The idea is to replace the Σ_{OR}-protocols used in the protocol depicted in Figure-1 by Blum's WI protocol for $\mathcal{NP}$, replace the practical Naor-Reingold PRF by a general OWF-based PRF, and replace the underlying VPTC scheme by the Feige-Shamir OWF-based two-round trapdoor commitment. Also, in the key-generation phase, the OWF f_V used by the honest verifier V is not required any longer to be one that admits Σ-protocols and V can use *any* OWF f_V in forming its public-key $y_V = f_V(x_V)$. But, for provable security, we need some cares on the implementation details. Specifically, for the underlying FSTC scheme we require the OWF f used in forming its first-round message be a preimage-verifiable OWF (this is necessary for proving concurrent soundness as discussed in the comments at the end of Section 4.2). Actually, this is also the only place that the theoretical protocol is not based on the minimal hardness assumption of any OWF. For complexity-leveraging, the prover and the verifier use security parameters n and N respectively that are the same as specified in Figure-1. But, here besides that the f_V used in the key-generation phase is required to be secure against $2^{N^{c_1}}$-time adversaries, we also require both the WI property of Blum's WI protocol for $\mathcal{NP}$ (*executed in Phase-1*) and the trapdoorness and hiding properties of the Feige-Shamir trapdoor commitment scheme hold against $2^{N^{c_1}}$-time adversaries.

In more details, in the first-round (i.e., Stage-1 of Phase-1), the prover P sends $(y_P = f_P(x_P), G_P, R_P)$ to the verifier V, where f_P is a preimage-verifiable OWF, x_P is a string of length n chosen randomly from the domain of f_P, G_P is a directed graph that is reduced from y_P by the Cook-Levin $\mathcal{NP}$-reduction, and R_P is a random string of length $3N$ serving as the first-round message of Naor's OWF-based commitment scheme [33]. (*Again, the* (y_P, G_P, R_P) *can be fixed once and for all.*) Then (i.e., in the Stage-2 of Phase-1), V firstly checks the validity of (y_P, G_P) and aborts if it is not valid. Otherwise (i.e., (y_P, G_P) is valid), V randomly chooses a string e_V from $\{0,1\}^n$, computes $c_V = FSTCCom(1^N, (y_P, G_P, R_P), e_V)$, sends c_V to P and proves to P by WIPOK for $\mathcal{NP}$ that it knows either a Hamiltonian cycle of G_P (equivalently, the preimage of y_P) or the preimage of y_V (i.e., its secret-key). After that (i.e., in Phase-2), P proves to V by WIPOK for $\mathcal{NP}$ that it knows either the witness of the common input or the preimage of y_V, in which V sends the random challenge by just revealing the committed e_V. The detailed protocol implementation is depicted in Figure-2 (page 19). It's easy to check the above resultant theoretical protocol still runs in 5 rounds (after round combinations accordingly).

Theorem 5.1 *Under any preimage-verifiable OWF (used by the prover) that is secure against standard polynomial-time adversaries and any OWFs (used by the verifier) that are secure against subexponential-time adversaries, the above protocol is a 5-round concurrently-sound rZK argument for $\mathcal{NP}$ in the BPK model.*

Proof (sketch).

Note that we only require that the verifier uses OWFs that are secure against $2^{N^{c_1}}$-time adversaries, which in turn guarantees that the security of verifier's public-key, the WI property of Blum's WI protocol for $\mathcal{NP}$ (*executed in Phase-1*) and the trapdoorness and hiding properties of the FSTC scheme all hold against $2^{N^{c_1}}$-time adversaries. For the preimage-verifiable OWF f_P used by the prover (in forming the first-round message of the underlying FSTC scheme), it can be only secure against standard polynomial-time adversaries, as the one-wayness of the preimage-verifiable OWF is only used to guarantee the

Key generation. Let f_V be any OWF that is secure against $2^{N^{c_1}}$-time adversaries. On a security parameter N, each honest verifier V randomly selects an element x_V of length N, computes $y_V = f_V(x_V)$, publishes y_V as its public-key PK while keeping x_V as its secret-key SK. (If P uses Naor's OWF-based perfectly-binding commitment scheme in Phase-2, V also deposits a random string R_V of length $3n$ as a part of its public-key, which serves the first-round message of Naor's commitment scheme.)

Common input. The public-file F, an element $x \in L \cap \{0,1\}^n$ and index i that specifies the i-th entry of F, PK_i. Note that the system security parameter is n.

P **private input.** An $\mathcal{NP}$-witness *wit* for $x \in L$, a pair of random strings (γ_1, γ_2), where γ_1 is a $poly(n)$-bit string and γ_2 is an n-bit string serving as the randomness seed of a PRF.

V **private input.** Private key SK_i such that $PK_i = f_V(SK_i)$.

Complexity-leverage used. Let c_1, $0 < c_1 < 1$, be the constant that the one-wayness of the OWF f_V, the WI property of the underlying Blum's protocol for HC (executed in Stage-2 of Phase-1), the hiding and trapdoorness properties of the underlying Feige-Shamir trapdoor commitment scheme all hold against any circuit of size $2^{N^{c_1}}$. Let c_2 be the constant that: for all sufficiently large n's, both the length of the witness *wit* for $x \in L \cap \{0,1\}^n$ and the size of G_P (reduced from y_P) are bounded by n^{c_2}. Then we set $\epsilon > c_2/c_1$ and $N = n^{\epsilon}$. This ensures that one can enumerate all potential witnesses *wit*, or all potential Hamiltonian cycles of G_P in time $2^{n^{c_2}}$, which is still lesser than the time it would take to break the one-wayness of f_V, or the WI property of the Blum's protocol for HC (executed in Stage-2 of Phase-1), or the hiding and trapdoorness properties of the underlying trapdoor commitment scheme, because $2^{n^{c_2}} < 2^{N^{c_1}}$.

Phase-1. Phase-1 consists of two stages:

Stage-1. Let f_P be any preimage-verifiable OWF. On security parameter n, P randomly selects an element x_P of length n in the domain of f_P, computes $y_P = f_P(x_P)$, reduces y_P to a directed graph G_P by Cook-Levin $\mathcal{NP}$-reduction such that finding a Hamiltonian cycle in G_P is equivalent to finding the preimages of y_P. For OWF-based solution, P also randomly selects a random string R_P of length $3N$ serving as the first-round message of Naor's OWF-based perfectly-binding commitment scheme used by V. Finally, P sends (y_P, G_P, R_P) to V. The randomness used by P in this process is γ_1, *which means that the* (y_P, G_P, R_P) *is fixed once and for all.*

Stage-2. V first checks the validity of (y_P, G_P) (i.e., whether or not G_P is reduced from y_P) and aborts if it is not valid. If it is valid, V randomly chooses a string e_V from $\{0,1\}^n$ and computes $c_V = FSTCCom(1^N, (G_P, R_P), e_V)$ (that is, V commits e_V using the underlying Feige-Shamir trapdoor commitment FSTC scheme). V sends c_V to P, and proves to P that it knows either the preimage of PK_i (i.e., SK_i) or a Hamiltonian cycle in G_P (equivalently, the preimage of y_P), by executing the (n-parallel repetitions of) Blum's WI protocol for $\mathcal{NP}$ on common input (PK_i, G_P, R_P) in which V plays the role of knowledge prover and P plays the role of knowledge verifier. Denote by a_V the first-round message (that is from knowledge prover V to knowledge verifier P) of the n-parallel repetitions of Blum's protocol for $\mathcal{NP}$, then all randomness used by P (from then on after receiving (c_V, a_V)) in the remaining computation is got by applying $PRF(\gamma_2, \cdot)$ on the "determining" message $(x, F, c_V, a_V, PK_i, i)$ *that "determines" the behaviors of even malicious verifier in the following interactions (i.e., either abort or providing valid decommitment information).* If V successfully finishes the Blum's WI protocol for $\mathcal{NP}$ in this stage and P accepts, then goto Phase-2. Otherwise, P aborts.

Phase-2. P proves to V that it knows either the witness *wit* for $x \in L$ or the preimage of PK_i, by executing the (n-parallel repetitions of) Blum's WI protocol for $\mathcal{NP}$ on common input (x, PK_i), in which V sends the assumed random challenge by just revealing e_V committed to c_V.

Figure 2. 5-round rZK-CS arguments for $\mathcal{NP}$ under any preimage-verifiable OWF

computationally-binding property of the underlying FSTC scheme against malicious polynomial-time verifiers (in proving black-box resettable zero-knowledge).

Black-box resettable zero-knowledge.

The proof of black-box rZK for the above theoretical protocol is almost the same as that for the protocol of Figure-1. Again, we remark that the above theoretical protocol is with real bare public-keys as we require nothing about public-keys registered by malicious verifiers.

Concurrent soundness.

The proof of concurrent soundness is a bit more complicated than that for the protocol of Figure-1.

Suppose in the concurrent attack issued by an s-concurrent malicious P^* against an honest verifier V with public-key y_V, with non-negligible probability $q(n)$ there exists a j, $1 \leqslant j \leqslant s(n)$, such that V outputs "accept x_j" in session j while $x_j \notin L$. The knowledge-extractor E (that on common input y_V runs P^* as a subroutine to output the preimage of y_V) works in the same way as described in Theorem 4.2. Now, we want to argue that P^* will also convince E of a false statement in one of the $s(n)$ sessions with probability $p(n)$ that is negligibly close to $q(n)$. This is trivial in Theorem 4.2 due to the perfect hiding and trapdoorness properties of the underlying VPTC scheme and the perfect WI property of Σ_{OR}. But, for the above theoretical protocol case, both the hiding and trapdoorness properties of the underlying FSTC scheme and the WI property of the Blum's WIPOK for $\mathcal{NP}$ are only *computationally* secure (against $2^{N^{c_1}}$-time adversaries). This is overcome by standard hybrid technique with a critical use of the underlying complexity-leveraging.

Specifically, we consider a hybrid experiment, in which an probabilistic polynomial-time (PPT) algorithm $\hat{E}$ takes (y_V, x_V) as input such that $y_V = f_V(x_V)$ (that is, $\hat{E}$ takes both the verifier's public-key and the corresponding secret-key as its input) and works in the same way as the knowledge-extractor E does but with the following modification: $\hat{E}$ uses x_V as its witness in Stage-2 of Phase-1 of any session just as the honest verifier does. Note that the difference between the interactions between P^* and the honest verifier V and the interactions between P^* and $\hat{E}$ is that: in the real interactions between P^* and V, V always commits (and accordingly decommits to) a random string of length n using the underlying FSTC scheme, but in the interactions between P^* and $\hat{E}$, $\hat{E}$ always commits 0^n and then decommits to a random string of length n by using the brute-force extracted trapdoor x_{P^*} (just as E does). The difference between the interactions between P^* and E and the interactions between P^* and $\hat{E}$ is that: E always uses the brute-force extracted x_{P^*} as its witness in Stage-2 of Phase-1 of each session, but $\hat{E}$ always uses the verifier's secret-key x_V as its witness (just as the honest verifier does).

Denote by $\hat{q}(n)$ the probability that P^* can convince $\hat{E}$ of a false statement in one of the $s(n)$ sessions, then if $|q(n) - \hat{q}(n)|$ is non-negligible, we can break the hiding and trapdoorness properties of the underlying FSTC scheme in the following way: We run P^* as a subroutine and interact with a player (who is either the honest verifier V or $\hat{E}$), and for each common statement selected by P^* we verify its verity by just working in $2^{n^{c_2}}$ time. Whenever we find P^* successfully convinces a false statement we output 1, otherwise we output 0. Clearly, by standard hybrid technique, if $|q(n) - \hat{q}(n)|$ is non-negligible we can break the hiding and trapdoorness properties of the underlying FSTC scheme in time $poly(n) \cdot 2^{n^{c_2}} < 2^{N^{c_1}}$. Similarly, we can also prove $|\hat{q}(n) - p(n)|$ is negligible, as otherwise we can break the WI property of Blum's protocol for $\mathcal{NP}$ in time $poly(n) \cdot 2^{n^{c_2}} < 2^{N^{c_1}}$. Finally, conditioned on $|q(n) - p(n)|$ is negligible, the rest of the proof is the same as described in Theorem 4.2. □

5.2 Round-optimal rZK-CS arguments for $\mathcal{NP}$

For the 5-round theoretical protocol developed in Section 5.1, if the verifier V uses a OWP-based one-round perfectly-binding commitment scheme then the prover only needs to send (y_P, G_P) in the first-round. To further reduce the round-complexity, we want to combine (y_P, G_P) into the third-round of the 5-round protocol (that is from the prover to the verifier), thereby obtaining 4-round (that is optimal) rZK-CS arguments for $\mathcal{NP}$. Recall that (y_P, G_P) is used by V in two ways: On one hand, it forms the $\mathcal{NP}$-statement (to be precise, a directed graph reduced from (y_P, y_V) by $\mathcal{NP}$-reduction) to be proved by V by Blum's WIPOK for HC in Stage-2 of Phase-1; On the other hand, it serves $TCPK$ of the underlying FSTC scheme with x_P (equivalently, a Hamiltonian cycle of G_P) as the trapdoor $TCSK$. To combine (y_P, G_P) into the third-round while remaining the same protocol structure, we need the following cryptographic tools.

1. A 3-round OWP-based WIPOK for HC, in which the prover sends the first-round message without knowing the $\mathcal{NP}$-statement (i.e., a directed graph) to be proved, other than the lower and upper bounds of the size of the graph (guaranteed by the underlying $\mathcal{NP}$-reduction).

2. A *one-round* OWP-based trapdoor commitment scheme based on HC, in which the committer sends the one-round commitments without knowing the HC graph G_P other than the lower and upper bounds of its size (guaranteed by the underlying $\mathcal{NP}$-reduction from y_P to G_P), and G_P is only sent in the decommitment stage after the commitment stage is finished.

For the first cryptographic tool of above, we note that the Lapidot-Shamir OWP-based 3-round WIPOK for HC [30] (also described in [18]) is just the protocol of the type we need. In the Lapidot-Shamir protocol, the prover sends the first-round message with only the knowledge of the size of the Hamiltonian graph to be proved. But, it can be easily extended to the case that the prover knows only the lower and upper bounds of the size of the graph to be proved. Below, we recall the details of the Lapidot-Shamir WIPOK for HC, which is the n-parallel repetitions of the following basic protocol.

Round-1. The prover P commits a adjacency matrix for a randomly-labeled cycle C of size q (without knowing the Hamiltonian graph to be proved). The commitment is done bit-by-bit using the one-round OWP-based perfectly-binding commitment scheme.

Round-2. The verifier V responds with a randomly chosen bit b

Round-3. Now, P is given the Hamiltonian graph $G = (V, E)$ with size $q = |V|$ to be proved and a Hamiltonian cycle C_G in G as its private input. If $b = 0$, then P opens all commitments (and V checks the revealed graph is indeed a q-cycle). If $b = 1$, then P sends a random permutation π mapping C_G (i.e., its private witness) to C (committed to its first-round message), and for each non-edge of G $(i, j) \notin E$ $(1 \leq i, j \leq q)$, P opens the value (that should be 0) committed to the $(\pi(i), \pi(j))$ entry of the adjacency matrix sent in the first-round message (and V checks all revealed values are 0 and the *unrevealed* entries in the committed adjacency matrix constitute a graph that is isomorphic to G via the permutation π).

In the above description, we have assumed P knows the size of the graph G to be proved. But, it can be easily extended to the case that P only knows the lower-bound $l(n)$ and the upper-bound $u(n)$ of the size of G. In this case, in the first-round P commits $(u(n) - l(n) + 1)$ many adjacency matrices for $(u(n) - l(n) + 1)$ many cycles with sizes ranging from $l(n)$ to $u(n)$. In the third-round, after the size of G is clear, P only decommits with respect to the unique cycle of according size.

Thus, the big challenge here is to develop a one-round trapdoor commitment scheme of the above described type, which however, to our knowledge, is unknown in the literature previously. To our purpose, we develop the trapdoor commitment of such type in this work, which is described below:

One-round commitment stage. To commit a bit 0, the committer sends a q-by-q adjacency matrix of commitments with each entry of the adjacency matrix committing to 0. To commit a bit 1, the committer sends a q-by-q adjacency matrix of commitments such that the entries committing to 1 constitute a randomly-labeled cycle C. We remark that the underlying commitment scheme used in this stage is the one-round OWP-based perfectly-binding commitment scheme.

Two-round decommitment stage. The commitment receiver sends a Hamiltonian graph $G = (V, E)$ with size $q = |V|$ to the committer. Then, to decommit to 0, the committer sends a random permutation π, and for each non-edge of G $(i, j) \notin E$, the committer reveals the value (that is 0) that is committed to the $(\pi(i), \pi(j))$ entry of the adjacency matrix sent in the commitment stage (and the receiver checks all revealed values are 0 and the unrevealed positions in the adjacency matrix constitute a graph that is isomorphic to G via the permutation π). To decommit to 1, the committer only reveals the committed cycle (and the receiver checks that all revealed values are 1 and the revealed entries constitute a q-cycle).

The computationally-hiding property of the above scheme is directly from that of the underlying perfectly-binding commitment scheme. The computationally-binding property of the above scheme is from the fact that the ability to decommit to both 0 and 1 for the same commitment-stage message implies extracting a Hamiltonian cycle of G. The trapdoorness property is from the following observation: After sending a commitment to 1, one can decommit to 1 in the normal way. However, it is also possible to decommit it to 0 if one knows the Hamiltonian cycle of G. Finally, note that in the above description we have assumed the committer knows the size of the graph G sent by the commitment receiver in the decommitment stage. But it can be easily extended to the case that the committer only knows the lower-bound $l(n)$ and the upper-bound $u(n)$ of the size of G. In this case, in commitment stage P sends $(u(n) - l(n) + 1)$ many adjacency matrices with vertex-sizes ranging from $l(n)$ to $u(n)$. In the decommitment stage, after the size of G is clear, P only decommits with respect to the unique adjacency matrix of according size.

Comments: Although the above one-round trapdoor commitment scheme is developed here to reduce round-complexity for rZK, but we remark that it is of independent value and, in particular, can be used to reduce round-complexity of other cryptographic protocols involving trapdoor commitments.

Finally, using almost the same proof procedure of Theorem 5.1, we can prove the following theorem:

Theorem 5.2 *Under the BPK model and any preimage-verifiable OWF (used by the prover) that is secure against standard polynomial-time adversaries and any OWF and OWP (used by the verifier in the key-generation phase and the protocol main-body respectively) that are secure against subexponential-time adversaries, any language in $\mathcal{NP}$ has a 4-round (that is optimal) rZK-CS argument.*

In particular, this implies that *round-optimal* rZK-CS arguments with real bare public-keys for $\mathcal{NP}$ can be implemented with *any certified one-way permutation*.

6 Constant-Round rZK-CS Arguments for $\mathcal{NP}$ with Zaps

In this section, we further show how to replace the underlying preimage-verifiable OWFs (used in the 5-round and 4-round rZK-CS arguments for $\mathcal{NP}$ developed in Section 5) by any OWFs *together with a novel use of zaps*, while remaining almost the same protocol structure.

Recall that zaps are themselves 2-round public-coin WI proofs for $\mathcal{NP}$. Furthermore, the first-round message of a zap can be fixed once and for all. Below, we first highlight the differences between the zap-based solutions and the previous preimage-verifiable OWF based protocols. The details of the zap-based constant-round rZK-CS arguments for $\mathcal{NP}$ with (real) bare public-keys are depicted in Figure 3 (page 24).

Key generation. In the zap-based solutions, besides publishing $y_V = f_V(x_V)$ where f_V is any OWF used by the verifier, the verifier V also posts the first-round message, denoted ξ, of the underlying zap as a part of its public-key PK. That is, the public-key of the verifier consists of (y_V, ξ). The secret-key of V is still x_V.

P's random-tape. P's random tape consists of a *single* random string γ of length n that serves as the randomness seed of the underlying PRF (rather than a pair of random strings as in the preimage-verifiable OWF-based protocols).

Phase-1. Let f_P be any OWF (rather than a preimage-verifiable OWF). In Phase-1 of any session with respect to a public-key $PK_i = (y_V^{(i)}, \xi^{(i)})$, rather than sending (y_P, G_P) (as does in the previous preimage-verifiable OWF based theoretical protocols), the prover does the following: on a security parameter n, it randomly selects $x_P^{(0)} \in \{0,1\}^n$ and $x_P^{(1)} \in \{0,1\}^n$, computes $y_P^{(0)} = f_P(x_P^{(0)})$ and

$y_P^{(1)} = f_P(x_P^{(1)})$, reduces $(y_P^{(0)}, y_P^{(1)})$ to a directed graph G_P by Cook-Levin $\mathcal{NP}$-reduction such that finding a Hamiltonian cycle in G_P is equivalent to finding one of the preimages of $(y_P^{(0)}, y_P^{(1)})$. Then, on input G_P, P computes the second-round message, denoted Π, of the zap showing the existence of a Hamiltonian cycle in G_P (equivalently, one of the preimages of $(y_P^{(0)}, y_P^{(1)})$). Finally, P sends $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$ to V. The randomness used by P in this process is got by applying $PRF(\gamma, \cdot)$ on PK_i. Note the randomness used by P in the remaining computations is got by applying $PRF(\gamma, \cdot)$ on the "determining" message of the current session.

Note that in previous preimage-verifiable OWF based theoretical protocols, the (y_P, G_P) is fixed once and for all sessions. But, in the zap-based solutions, the $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$ is determined by the public-key used in the current session. That is, the $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$ is fixed once and for all sessions *with respect to the same verifier's public-key*. But, for sessions with respect to a different verifier's public-key, the $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$ will also be different. Also, note that in this case, the WIPOK for $\mathcal{NP}$ on input $(G_P, y_V^{(i)})$ executed in Stage-2 of Phase-1 (for showing the knowledge of either the preimage of $y_V^{(i)}$ or a Hamiltonian cycle in G_P) can be equivalently viewed as a protocol for showing the knowledge of either the preimage of y_V or one of the preimages of $(y_P^{(0)}, y_P^{(1)})$.

Similar to that of the preimage-verifiable OWF-based protocols, the OWF-based implementation of the protocol depicted in Figure 3 runs in 5 rounds (after round combinations accordingly), and the OWP-based implementation runs in 4 rounds (that is optimal), by using the 3-round OWP-based Lapidot-Shamir WIPOK for $\mathcal{NP}$ and the one-round OWP-based Yung-Zhao trapdoor commitment scheme (developed in Section 5.2).

We now proceed to the security analysis of the zap-based solutions. Recall that in the previous preimage-verifiable OWF based rZK-CS arguments for $\mathcal{NP}$ (developed in Section 5), the prover sends $y_P = f_P(x_P)$ and G_P (that is reduced from y_P by the Cook-Levin $\mathcal{NP}$-reduction) to the verifier, where f_P is assumed to be a preimage-verifiable OWF. In the security proof, the proof of concurrent soundness requires the preimage-verifiability property of f_P. And the proof of rZK employs the one-wayness property of f_P to guarantee that the WI protocol on (G_P, y_V) (equivalently, (y_P, y_V)) executed in Stage-2 of Phase-1 is actually an argument of knowledge of the preimage of y_V (where y_V is the public-key of the verifier).

For provable security of the above zap-based solutions, we need the $(y_P^{(0)}, y_P^{(1)}, G, \Pi)$ sent by the prover satisfy the following properties:

Preimage-verifiability with respect to well-formed public-key. For any *well-formed* verifier's public-key (in particular, any well-formed the first-round message of the underlying zap ξ), any (possibly maliciously formed) $(y_P^{(0)}, y_P^{(1)})$ and G_P that is reduced from $(y_P^{(0)}, y_P^{(1)})$ by $\mathcal{NP}$-reduction such that finding a Hamiltonian cycle in G_P is equivalent to finding one of the preimages of $(y_P^{(0)}, y_P^{(1)})$, and any (possibly maliciously formed) Π, with overwhelming probabilities it is the case that Π is a valid second-round message of the zap if and only if G_P contains a Hamiltonian cycle (equivalently, there exists an element x_P such that either $y_P^{(0)} = f_P(x_P)$ or $y_P^{(1)} = f_P(x_P)$.

We just observe that the preimage-verifiability property of the (possibly maliciously formed) $(y_P^{(0)}, y_P^{(1)}, G, \Pi)$ with respect to well-formed verifier's public-key is direct from the fact that the underlying zap is a *proof* for $\mathcal{NP}$.

One-wayness with respect to even maliciously formed public-key. For any (*even maliciously formed*) verifier's public-key (in particular, any maliciously formed the first-round message of the underlying zap), and any *well-formed* $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$, no PPT algorithm can compute out a

Key generation. Let f_V be any OWF that is secure against $2^{N^{c_1}}$-time adversaries. On a security parameter N, each honest verifier V randomly selects an element x_V of length N, computes $y_V = f_V(x_V)$. V also forms the first-round message, denoted ξ, of the underlying zap. Then, V posts (y_V, ξ) as its public-key PK. The secret-key of V is still x_V. (If P uses Naor's OWF-based perfectly-binding commitment scheme in Phase-2, V also deposits a random string R_V of length $3n$ as a part of its public-key, which serves the first-round message of Naor's commitment scheme.)

Common input. The public-file F, an element $x \in L \cap \{0,1\}^n$ and index i that specifies the i-th entry of F, $PK_i = (y_V^{(i)}, \xi^{(i)})$. Note that the system security parameter is n.

P **private input.** An $\mathcal{NP}$-witness *wit* for $x \in L$, a single n-bit random string γ serving as the randomness seed of a PRF.

V **private input.** Private key SK_i such that $y_V^{(i)} = f_V(SK_i)$.

Complexity-leverage used. Let c_1, $0 < c_1 < 1$, be the constant that the one-wayness of the OWF f_V, the WI property of the underlying WI protocol for $\mathcal{NP}$ (executed in Stage-2 of Phase-1), the hiding and trapdoorness properties of the underlying trapdoor commitment scheme all hold against any circuit of size $2^{N^{c_1}}$. Let c_2 be the constant that: for all sufficiently large n's, both the length of the witness *wit* for $x \in L \cap \{0,1\}^n$ and the size of G_P (reduced from $(y_P^{(0)}, y_P^{(1)})$) are bounded by n^{c_2}. Then we set $\epsilon > c_2/c_1$ and $N = n^{\epsilon}$. This ensures that one can enumerate all potential witnesses *wit*, or all potential Hamiltonian cycles of G_P in time $2^{n^{c_2}}$, which is still lesser than the time it would take to break the one-wayness of f_V, or the WI property of the WI protocol for $\mathcal{NP}$ (executed in Stage-2 of Phase-1), or the hiding and trapdoorness properties of the underlying trapdoor commitment scheme, because $2^{n^{c_2}} < 2^{N^{c_1}}$.

Phase-1. Phase-1 consists of two stages:

Stage-1. Let f_P be any OWF. On security parameter n, P randomly selects two elements $x_P^{(0)}$ and $x_P^{(1)}$ of length n each in the domain of f_P, computes $y_P^{(b)} = f_P(x_P^{(b)})$ for $b \in \{0,1\}$, reduces $(y_P^{(0)}, y_P^{(1)})$ to a directed graph G_P by Cook-Levin $\mathcal{NP}$-reduction such that finding a Hamiltonian cycle in G_P is equivalent to finding one of the preimages of $(y_P^{(0)}, y_P^{(1)})$. (For OWF-based solution, P also randomly selects a random string R_P of length $3N$ serving as the first-round message of Naor's OWF-based perfectly-binding commitment scheme used by V.) Then, on common input G_P and with $\xi^{(i)}$ as the first-round message of the underlying zap, P computes the second-round message, denoted Π, of the underlying zap showing the existence of a Hamiltonian cycle in G_P (equivalently, one of the preimages of $(y_P^{(0)}, y_P^{(1)})$). Finally, P sends $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi, R_P)$ to V. The randomness used by P in this process is got by applying $PRF(\gamma, \cdot)$ on PK_i.

Stage-2. V first checks the validity of $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$ (i.e., whether or not G_P is reduced from $(y_P^{(0)}, y_P^{(1)})$ and $(\xi^{(i)}, \Pi)$ is valid zap proof on common input G_P), and aborts if it is not valid. If it is valid, V randomly chooses a string e_V from $\{0,1\}^n$ and computes $c_V = TCCom(1^N, (G_P, R_P), e_V)$ (that is, V commits e_V using the underlying trapdoor commitment scheme that is either the Feige-Shamir scheme or the Yung-Zhao scheme developed in Section 5.2). V sends c_V to P, and proves to P that it knows either the preimage of $y_V^{(i)}$ (i.e., SK_i) or a Hamiltonian cycle in G_P (equivalently, one of the preimages of $(y_P^{(0)}, y_P^{(1)})$), by executing (n-parallel repetitions of) the underlying WIPOK for $\mathcal{NP}$ on common input (PK_i, G_P) in which V plays the role of knowledge prover and P plays the role of knowledge verifier. Denote by a_V the first-round message (that is from knowledge prover V to knowledge verifier P) of the n-parallel repetitions of the underlying WIPOK for $\mathcal{NP}$, then all randomness used by P (from then on after receiving (c_V, a_V)) in the remaining computation is got by applying $PRF(\gamma, \cdot)$ on the "determining" message $(x, F, c_V, a_V, PK_i, i)$. If V successfully finishes the WI protocol for $\mathcal{NP}$ in this stage and P accepts, then goto Phase-2. Otherwise, P aborts.

Phase-2. P proves to V that it knows either the witness *wit* for $x \in L$ or the preimage of $y_V^{(i)}$ in PK_i, by executing (n-parallel repetitions of) of the underlying WIPOK for $\mathcal{NP}$ on common input (x, PK_i), in which V sends the assumed random challenge by just revealing e_V committed to c_V.

Figure 3. Constant-round rZK-CS arguments for $\mathcal{NP}$ with zaps

Hamiltonian cycle in G_P (equivalently, one of the preimages of $(y_P^{(0)}, y_P^{(1)})$) with non-negligible probabilities.

The one-wayness of *well-formed* $(y_P^{(0)}, y_P^{(1)}, G_P, \Pi)$ with respect to even maliciously formed public-key is from the pseudorandomness of the output of the underlying PRF, and the observation that

in this case WI implies witness hiding (WH).

Now, for the proof of concurrent soundness of the zap-based solution, note that if the $(y^{(0)}_{P^*}, y^{(1)}_{P^*}, G_{P^*}, \Pi)$ sent by a malicious prover P^* holds the above preimage-verifiability property with respect to well-formed verifier's public-key, then the security proof of concurrent soundness of the zap-based solution remains the same as that of the previous preimage-verifiable OWF based theoretical protocols.

For the proof of rZK, the rZK simulator, denoted S_{ZAP}, of the zap-based solution works in almost the same way as the rZK simulator for the preimage-verifiable OWF based theoretical protocols, but with the following modifications: For each public-key PK_i in the public-key file ($1 \leq i \leq s(n)$), S_{ZAP} generates $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i}, G_{PK_i}, \Pi_{PK_i})$ by using truly random coins, and the $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i}, G_{PK_i}, \Pi_{PK_i})$ is then fixed once and for all sessions with respect to the same public-key PK_i (here we have assumed the real honest prover uses a truly random function). Note that in the previous preimage-verifiable OWF based protocols, the (y_P, G_P) is fixed once and for all sessions, regardless the public-keys used in the sessions.

For the rZK simulation to be successful, we need to argue that with overwhelming probabilities the rZK simulator can either output an indistinguishable simulated transcript or extract all secret-keys corresponding to public-keys deposited in the public file (in case such exist) in polynomial time. The proof procedure is almost identical to that of the previous preimage-verifiable OWF based protocols, but with a subtlety to be addressed here. Specifically, to successfully extract the secret-keys in polynomial time we need to require the WIPOK on (G_P, PK_i) (equivalently, $(y^{(0)}_P, y^{(1)}_P, y^{(i)}_{V^*})$) executed in Stage-2 of Phase-1 of the zap-based solution is actually an *argument of the knowledge* of the preimage of $y^{(i)}_{V^*}$. Note that this is obvious in the preimage-verifiable OWF based protocols due to the one-wayness of the (y_P, G_P) (that is fixed once and for all sessions) and a malicious verifier V^* does not have any additional assistant information to help it to break the one-wayness of G_P (equivalently, y_P). But, the case of the zap-based solution is different. Specifically, a malicious verifier V^* not only sees G_P (equivalently $(y^{(0)}_P, y^{(1)}_P)$) but also accesses a piece of (seemingly very helpful) assistant information (i.e., a WI-proof Π showing the existence of Hamiltonian cycles in G_P). This is overcome by standard reduction argument.

Specifically, consider the following PPT algorithm $\hat{S}$. The input of $\hat{S}$ consists of the public-key file $F = \{PK_1, \cdots, PK_{s(n)}\}$ generated by V^* in its first stage and a set of inputs $\{(y^{(0)}_{PK_1}, y^{(1)}_{PK_1}, G_{PK_1}, \Pi_{PK_1}), \cdots, (y^{(0)}_{PK_{s(n)}}, y^{(1)}_{PK_{s(n)}}, G_{PK_{s(n)}}, \Pi_{PK_{s(n)}})\}$, where each $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i}, G_{PK_i}, \Pi_{PK_i})$ ($1 \leq i \leq s(n)$) is generated with respect to public-key PK_i by using independent truly random coins. $\hat{S}$ runs V^* as a subroutine and works in the same way as the rZK simulator S_{ZAP} does with the following modification: In each session with respect to a public-key PK_i, $\hat{S}$ just sends $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i}, G_{PK_i}, \Pi_{PK_i})$ in its input set as the Stage-1 message in Phase-1 of that session. Clearly, from the viewpoint of V^*, the behavior of $\hat{S}$ is identical to that of S_{ZAP}. Now, suppose in the interactions between $\hat{S}$ and V^*, in the Phase-1 of a session with respect to a (uncovered) public-key PK_i that consists of $y^{(i)}_{V^*}$ and possibly maliciously formed the first-round message of the zap, V^* successfully finishes the WIPOK on $(G_{PK_i}, y^{(i)}_{V^*})$ (equivalently, $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i}, y^{(i)}_{V^*})$) but $\hat{S}$ extracts in polynomial-time one of the preimages of $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i})$ (rather than the expected preimage of $y^{(i)}_{V^*}$), then this will violate the above assumption that no PPT algorithm can compute out one of the preimages of $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i})$ from $(y^{(0)}_{PK_i}, y^{(1)}_{PK_i}, G_{PK_i}, \Pi_{PK_i})$ for any PK_i, $1 \leq i \leq s(n)$.

After solving the above subtlety in the rZK simulation of the zap-based solution, the rest of the rZK proof procedure is identical with that of the preimage-verifiable OWF based theoretical protocols. Thus, we have the following theorem:

Theorem 6.1 *Under the existence of zaps that are secure against standard polynomial-time adversaries and any OWF (resp. any OWP) used by the verifier that is secure against subexponential-time adver-*

saries, any language in $\mathcal{NP}$ has a 5-round (resp. 4-round that is optimal) rZK-CS argument in the BPK model.

Note that the existence of (single-theorem) NIZK proofs for $\mathcal{NP}$ implies the existence of zaps.

Comments: Actually, the process of generating $(y_P^{(0)}, y_P^{(1)}, G, \Pi)$, together with publishing the first-round message of the zap as a public-key, can be viewed as a generalized version of preimage-verifiable OWFs *in the public-key model*. Such a novel use of zaps in the public-key setting might also be of independent value. We also note that the existence of zaps should actually be a different assumption than the assumption of preimage-verifiable OWFs, and it is hard to say one is weaker than another.

7 Constant-Round rZK-CS Arguments for $\mathcal{NP}$ under Minimal Hardness Assumptions

In this section, we further modify the above zap-based rZK-CS protocols (presented in Section 6) into constant-round rZK-CS for $\mathcal{NP}$ in the BKP model under minimal hardness assumptions (i.e., any subexponentially strong OWF or OWP). The idea is to replace the underlying zaps in the above zap-based protocols by constant-round rWI arguments for $\mathcal{NP}$. But, as we shall see, the security proof in this case turns out to be much more complicated and subtler. To this end, we also present constant-round rWI *arguments* for $\mathcal{NP}$ *in the standard model* under minimal hardness assumptions, a result unknown previously to our knowledge that is of independent value. The detailed protocol description is depicted in Figure 4 (page 33).

The OWF-based protocol depicted in Figure-3 runs in 7 rounds after round combinations accordingly. In particular, the first two rounds of Phase-4 can be combined into previous phases. The round-complexity can be further reduced to 6 if we aim for OWP-based protocol implementation, by using the OWP-based Yung-Zhao one-round trapdoor commitment scheme developed in Section 5.2, OWP-based Lapidot-Shamir WIPOK for $\mathcal{NP}$, and reversing the order of Stage-1 and Stage-2 in Phase-1. Now, for the protocol depicted in Figure-4, we have the following theorem:

Theorem 7.1 *Under any (subexponentially strong) OWF (resp. OWP), any language in $\mathcal{NP}$ has a 7-round (resp. 6-round) rZK-CS argument in the BPK model.*

Proof (sketch).

Concurrent soundness.

We remark that if a PPT adversary P^* can only successfully convince the honest verifier V of a false G_{P^*} (such that there exists no Hamiltonian cycle in G_{P^*}) with negligible probabilities in Phase-2 of any session, then the proof of concurrent soundness here still remains the same as that for previous preimage-verifiable OWF based protocols.

Now, we show that if a PPT concurrent s-adversary P^* can convince V (with public-key $y_V = f_V(x_V)$) of a false G_{P^*} with non-negligible probability $q(n)$ in Phase-2 of one of the $s(n)$ sessions, then this will violate the hiding property of the underlying perfectly-binding commitment scheme, denoted Com, used by V in Phase-1 that is run on security parameter N. Note that according to the hiding property of the underlying perfectly-binding commitment scheme used by V in Phase-1, given two random strings $\hat{e}_0$ and $\hat{e}_1$ of length n each and $C = Com(1^N, R_{P^*}, \hat{e}_b)$ for a randomly chosen bit $b \in \{0,1\}$, no $2^{N^{c_1}}$-time non-uniform algorithm can distinguish whether C commits to $\hat{e}_0$ or to $\hat{e}_1$ (i.e., guess the bit b correctly) with non-negligible advantage over $1/2$, even with $\hat{e}_0$, $\hat{e}_1$ and the secret-key of V (i.e., x_V) as its non-uniform inputs.

We construct a non-uniform algorithm E who takes $(1^n, (\hat{e}_0, \hat{e}_1, x_V), C)$ as input and wants to guess b with a non-negligible advantage over $1/2$, where $\hat{e}_0$ and $\hat{e}_1$ are taken uniformly at random from $\{0,1\}^n$ and $C = Com(1^N, R_{P^*}, \hat{e}_b)$ for a randomly chosen bit $b \in \{0,1\}$. E randomly selects i from $\{1, \cdots, s(n)\}$, runs P^* as a subroutine by playing the role of the honest verifier V with x_V as its secret-key. In the i-th session, after receiving G_{P^*} from P^* at Stage-1 of Phase-1, E first checks whether there exists a Hamiltonian cycle in G_{P^*} or not by brute-force searching in time $2^{n^{c_2}}$. If E finds a Hamiltonian cycle in G_{P^*}, then E randomly guesses the bit b and stops. Otherwise (i.e., there exists no Hamiltonian cycle in G_{P^*}), E runs P^* further and continues the interactions of the i-th session as follows: E gives C to P^* (as the assumed commitment to $e_V^{(0)}$) at Stage-2 of Phase-1. After receiving the first-round message of the Blum's protocol for HC in Phase-2 (which contains n committed adjacency matrices), E first opens all the committed adjacency matrices by brute-force in time $poly(n) \cdot 2^{n^{c_2}}$ (note that E can do this as the underlying perfectly-binding commitment scheme used by the prover is run on security parameter n). For each revealed graph G_k $(1 \leq k \leq n)$ (described by the corresponding opened adjacency matrix) , we say G_k is a 0-valid graph if it is isomorphic to G_{P^*}, or a 1-valid graph if it contains a Hamiltonian cycle of the same size of G_{P^*}. We say the set of revealed graphs $\{G_1, \cdots, G_n\}$ is $\hat{e}_b$-valid ($b \in \{0,1\}$) if for all k, $1 \leq k \leq n$, G_k is a $\hat{e}_b^{(k)}$-valid graph, where $\hat{e}_b^{(k)}$ denotes the k-th bit of $\hat{e}_b$. Note that for the set of revealed graphs $\{G_1, \cdots, G_n\}$, E can determine whether it is $\hat{e}_0$-valid or $\hat{e}_1$-valid in time $poly(n) \cdot 2^{n^{c_2}}$. Then, E outputs 0 if the set $\{G_1, \cdots, G_n\}$ is $\hat{e}_0$-valid but not $\hat{e}_1$-valid. Similarly, E outputs 1 if the set $\{G_1, \cdots, G_n\}$ is $\hat{e}_1$-valid but not $\hat{e}_0$-valid. In other cases, E just randomly guesses the bit b.

The key observation here is that if G_{P^*} is false (i.e., containing no Hamiltonian cycle), then for each revealed graph it cannot be both a 0-valid graph and a 1-valid graph. Similarly, for false G_{P^*}, the set of revealed graphs $\{G_1, \cdots, G_n\}$ cannot be both $\hat{e}_0$-valid and $\hat{e}_1$-valid for different $\hat{e}_0 \neq \hat{e}_1$. Furthermore, suppose C commits to $\hat{e}_b$ ($b \in \{0,1\}$), then for false G_{P^*}, with probability $1 - 2^{-n}$ the set of revealed graphs $\{G_1, \cdots, G_n\}$ is not $\hat{e}_{1-b}$-valid. As the value i is randomly chosen from $\{1, \cdots, s(n)\}$, we conclude that E can successfully guess the bit b with probability at least $(1-2^{-n}) \cdot \frac{q(n)}{s(n)} + \frac{1}{2}(1 - \frac{q(n)}{s(n)}) = \frac{1}{2} + \frac{1}{2} \cdot \frac{q(n)}{s(n)} - 2^{-n} \cdot \frac{q(n)}{s(n)}$ in time $poly(n) \cdot 2^{n^{c_2}} < 2^{N^{c_1}}$. That is, E successfully guesses the bit b with non-negligible advantage over $1/2$ in time $poly(n) \cdot 2^{n^{c_2}} < 2^{N^{c_1}}$, which violates the hiding property of the underlying perfectly-binding commitment scheme used by V that is run on the security parameter N.

Black-box resettable zero-knowledge.

The rZK simulator works in the same way as the rZK simulator for the preimage-verifiable OWF based protocols. Note that in the rZK simulation, we get into trouble only if the malicious verifier V^* successfully finishes Phase-3 in one session with respect to a yet "uncovered" public-key. In such a case, in order to continue, we need to extract the corresponding secret-key of an uncovered public-key in expected polynomial time, by rewinding V^* to the point (possibly, back to a past partial transcript) that V^* just sent the same Stage-2 message of Phase-1 of that session *at the first time* and redefining the random function (we have assumed the honest prover uses a truly random function rather than a pseudorandom one). Note that, in particular, during each such rewinding, the rZK simulator needs also to provide "fresh" Phase-2 messages (i.e., the first-round and the third-round messages of Phase-2) on the fixed common input $(y_P^{(0)}, y_P^{(1)}, G_P)$ and with respect to the same Stage-2 message of Phase-1 (determined by the rewinding), using "fresh" randomness by redefining the underlying random function.

So, pending on our ability to cover new uncovered public-keys within time inversely proportional to the probability that we encounter a success of Phase-3 relative to a yet uncovered public-key, the rZK simulation will be done in expected polynomial time and the indistinguishability between the simulated transcript and the real (resetting) interaction transcript is from the rWI property of Phase-3 combined with Stage-2 of Phase-1. To successfully extract the secret-keys in polynomial time we need to

require the underlying Lapidot-Shamir or Blum's protocol on (G_P, PK_i) (equivalently, $(y_P^{(0)}, y_P^{(1)}, PK_i)$) executed in Stage-2 of Phase-1 together with Phase-3 is actually an *argument of the knowledge* of the preimage of PK_i (i.e., the secret-key SK_i). But, the subtle and complicated situation here is that before V^* finishes Phase-3, the honest prover has already proved the knowledge of the Hamiltonian cycle of G_P (equivalently, the preimage of either $y_P^{(0)}$ or $y_P^{(1)}$) in Phase-2, and furthermore V^* is resettingly interacting with the instances of the honest prover (which means that V^* can rewind the honest prover instances arbitrarily at its wish). So, one may argue that by rewinding the honest prover instances arbitrarily, V^* may potentially forge the interactions on $(y_P^{(0)}, y_P^{(1)}, G_P)$ provided by the honest prover in Phase-2 of one session into successful but "false" interactions on (G_P, PK_j) in Stage-2 of Phase-1 and Phase-3 of another session with respect to public-key PK_j, in the sense that although the interactions are valid but V^* actually does not know the corresponding secret-key of PK_j. This means that, in such case, the interactions on (G_P, PK_i) (equivalently, $(y_P^{(0)}, y_P^{(1)}, PK_i)$) executed in Phase-3 together with Stage-2 of Phase-1 are not any longer an *argument of the knowledge* of the preimage of PK_i, although it is still a system for proof of knowledge of the Hamiltonian cycle in G_P or the preimage of PK_i. What save us here is the concurrent (not resettable) WI property of the Blum's protocol for HC.

Below, we construct an algorithm $\hat{S}$ that emulates the real rZK simulator while *concurrently* (not resettingly) running the Blum's protocol for HC. That is, by playing the role of knowledge verifier $\hat{S}$ concurrently interacts with instances of the knowledge prover, denoted $\hat{P}$, of Blum's protocol for HC on common input $(y_P^{(0)}, y_P^{(1)}, G_P)$, where G_P is reduced from $(y_P^{(0)}, y_P^{(1)})$ by $\mathcal{NP}$-reduction; At the same time, $\hat{S}$ runs the s-resetting malicious V^* as a subroutine by playing the role of the honest prover, and sets $(y_P^{(0)}, y_P^{(1)}, G_P)$ as the part of the Stage-1 message of Phase-1 (that is fixed once and for all). $\hat{S}$ emulates the real rZK simulator but with the following modification: whenever $\hat{S}$ needs to sends a "fresh" first-round message of Blum's protocol for HC on $(y_P^{(0)}, y_P^{(1)}, G_P)$ in Phase-2 with respect to a "determining" Stage-2 message of Phase-1 (this happens due to either V^* sends a distinct "determining" Stage-2 message in one session or $\hat{S}$ needs rewinding V^* and redefining the underlying random function to extract knowledge used by V^* in a successful execution of Stage-2 of Phase-1 together with Phase-3 with respect to an uncovered public-key), it initiates a new session with $\hat{P}$ on $(y_P^{(0)}, y_P^{(1)}, G_P)$, and forwards the first-round message received from $\hat{P}$ to V^*. Then, $\hat{S}$ runs V^* further and in case V^* successfully reveals the assumed challenge (that is committed to the "determining" Stage-2 message in question) then $\hat{S}$ returns back the revealed challenge to $\hat{P}$ as its own challenge in the according simultaneous execution of Blum's protocol for HC, and returns back the third-round message received from $\hat{P}$ to V^*. For a session with "determining" message identical to that of some previous message, $\hat{S}$ just copies what sent in the previous session (note that in this case, although $\hat{S}$ may still possibly need to get some third-round messages in some *existing* concurrent sessions with $\hat{P}$, but it does not need to initiate a new concurrent session with $\hat{P}$).

Note that from the viewpoint of V^*, the behavior of $\hat{S}$ is identical to the behavior of the real rZK simulator, where the real rZK simulator generates $(y_P^{(0)}, y_P^{(1)}, G_P)$ and provides the corresponding messages of Phase-2 by itself (rather than interacting with instances of a knowledge prover $\hat{P}$ of the Blum's protocol for HC on $(y_P^{(0)}, y_P^{(1)}, G_P)$). The key observation here is that although V^* is actually resettingly interacting with $\hat{S}$, but $\hat{S}$ only concurrently interacts with $\hat{P}$ and never rewinds $\hat{P}$ (the underlying reason is just that in each session Phase-2 interactions take place only after V^* sent the "determining" Stage-2 message of Phase-1 that determines the behaviors of V^* in the following interactions of that session). Note that in this case, the (concurrent) WI property of the Blum's protocol for HC on common input $(y_P^{(0)}, y_P^{(1)}, G_P)$ actually implies witness hiding (WH), which means no PPT algorithm can output a Hamiltonian cycle in G_P (equivalently, the preimage of either $y_P^{(0)}$ or $y_P^{(1)}$) from concurrent interactions with $\hat{P}$. Also note that on common input (G_P, PK_i), Phase-3 together with Stage-2 of Phase-1 is always a system for proof of knowledge of either a Hamiltonian cycle in G_P or the preimage of PK_i,

which means that with overwhelming probabilities $\hat{S}$ (or the real rZK simulator) always can extract either a Hamiltonian cycle in G_P or the corresponding secret-key within time inversely proportional to the probability that V^* successfully finishes Phase-3 (by rewinding V^* and redefining the underlying random function). But, the WH property of Blum's protocol for HC shows that with overwhelming probabilities, $\hat{S}$ (or the real rZK simulator) never outputs a Hamiltonian cycle of G_P in its simulation that is done in expected polynomial-time. □

7.1 Constant-round rWI arguments for $\mathcal{NP}$ under minimal hardness assumptions in the standard model

We remark the protocol depicted in Figure 4 actually implies constant-round rWI *arguments* for $\mathcal{NP}$ under minimal hardness assumptions *in the standard model* (specifically, the Phase-2 together with Stage-2 of Phase-1 on the top), a result unknown previously that is of independent value. The rWI protocol is reproduced in Figure 5 (page 34).

The OWF-based implementation of the protocol depicted in Figure 5 runs in 5 rounds and the OWP-based implementation runs in 4 rounds. The (computational) soundness of the protocol depicted in Figure 5 is direct from the proof of concurrent soundness for the rZK-CS protocol depicted in Figure 4, and the rWI property is direct from the general paradigm of [8] for achieving rWI from admissible system. Thus, we have the following theorem:

Theorem 7.2 *Under any (subexponentially strong) OWF (resp. OWP), any language in $\mathcal{NP}$ has a 5-round (resp. 4-round) rWI argument in the standard model.*

8 Concluding Remarks and Future Investigations

In this work, we show how to achieve both highly practical constant-round rZK-CS arguments (for specific number-theoretic languages) and theoretical constant-round rZK-CS arguments for $\mathcal{NP}$ (with optimal round-complexity or under minimal hardness assumptions) in the BPK model. Our protocols directly imply efficient identifications secure against resetting attacks. More importantly, as rZK is a generalization and strengthening of the notion of concurrent ZK and because zero-knowledge plays a central role in modern cryptography, our protocols can be employed as critical building blocks to achieve other round-efficient practical or theoretical (but round-optimal or minimal hardness assumption based) concurretnly/resettablly secure cryptographic schemes with real bare public-keys.

We remark that there is an interesting phenomenon with our approaches for constant-round rZK-CS arguments in BPK. Specifically, to weaken the underlying hardness assumptions, the round complexity of our protocols increases accordingly (or in other words, to reduce the round-complexity, the underlying hardness assumptions will be strengthened). For example, 4-round rZK-CS arguments for $\mathcal{NP}$ in BPK exist under any certified OWP, 5-round protocols can be based on any preimage-verifiable OWF (that is weaker than certified OWP), 6-round protocols are based on any OWP and 7-round protocols are based on any OWF. We do not know whether this phenomenon is intrinsic or relative to the technical approaches we employed. We suggest it as an interesting open problem to show whether or not round-optimal rZK-CS arguments for $\mathcal{NP}$ in BPK exist under minimal hardness assumptions (i.e., any OWF or OWP).

Acknowledgments. We would like to thank Giovanni Di Crescenzo for very helpful discussions with him.

References

[1] B. Barak, O. Goldreich, S. Goldwasser and Y. Lindell. Resettably-Sound Zero-Knowledge and Its Applications. In *IEEE Symposium on Foundations of Computer Science*, pages 116-125, 2001.

[2] M. Bellare, M. Fischlin, S. Goldwasser and S. Micali. Identification protocols secure against reset attacks. In *B. Pfitzmann (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 2001, LNCS 2045*, pages 495–511. Springer-Verlag, 2001.

[3] M. Bellare and O. Goldreich. On Defining Proofs of Knowledge In *E. F. Brickell (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1992, LNCS 740*, pages 390-420, Springer-Verlag, 1992.

[4] M. Bellare and M. Yung. Certifying Cryptographic Tools: The Case of Trapdoor Permutations. In *E. F. Brickell (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1992, LNCS 740*, pages 442-460, Springer-Verlag, 1992.

[5] M. Blum. Coin Flipping by Telephone. In *proc. IEEE Spring COMPCOM*, pages 133-137, 1982.

[6] M. Blum. How to Prove a Theorem so No One Else can Claim It. In Proceedings of the International Congress of Mathematicians, Berkeley, California, USA, 1986, pp. 1444-1451.

[7] Brassard, D. Chaum and C. Crepeau. Minimum Disclosure Proofs of Knowledge. *Journal of Computer Systems and Science*, 37(2): 156-189, 1988.

[8] R. Canetti, O. Goldreich, S. Goldwasser and S. Micali. Resettable Zero-Knowledge. In *ACM Symposium on Theory of Computing*, pages 235-244, 2000. Available from: http://www.wisdom.weizmann.ac.il/∼oded/

[9] R. Canetti, J. Kilian, E. Petrank and A. Rosen. Black-Box Concurrent Zero-Knowledge Requires $\tilde{\Omega}(log\ n)$ Rounds. In *ACM Symposium on Theory of Computing*, pages 570-579, 2001.

[10] R. Canetti, J. Kilian, E. Petrank and A. Rosen. Black-Box Concurrent Zero-Knowledge Requires (Almost) Logarithmically Many Rounds. In *SIAM Journal on Computing*, 32(1): 1-47, 2002.

[11] R. Cramer. Modular Design of Secure, yet Practical Cryptographic Protocols, PhD Thesis, University of Amsterdam, 1996.

[12] R. Cramer and I. Damgard. On Electronic Payment Systems. A lecture note for the course of Cryptographic Protocol Theory at Aarhus University, 2003. Available from: http://www.daimi.au.dk/∼ivan/CPT.html

[13] R. Cramer, I. Damgard and B. Schoenmakers. Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In *Y. Desmedt (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1994, LNCS 839*, pages 174-187. Springer-Verlag, 1994.

[14] I. Damgard. On Σ-protocols. A lecture note for the course of Cryptographic Protocol Theory at Aarhus University, 2003. Available from: http://www.daimi.au.dk/∼ivan/CPT.html

[15] G. Di Crescenzo, G. Persiano and I. Visconti. Constant-Round Resettable Zero-Knowledge with Concurrent Soundness in the Bare Public-Key Model In *M. Franklin (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2004, LNCS 3152*, pages 237-253. Springer-Verlag, 2004.

[16] C. Dwork, M. Naor and A. Sahai. Concurrent Zero-Knowledge. In *ACM Symposium on Theory of Computing*, pages 409-418, 1998.

[17] C. Dwork and M. Naor. Zaps and Their Applications. In *IEEE Symposium on Foundations of Computer Science*, pages 283-293, 2000.

[18] U. Feige. Alternative Models for Zero-Knowledge Interactive Proofs. Ph.D Thesis, Weizmann Institute of Science, 1990.

[19] U. Feige, D. Lapidot and A. Shamir. Multiple Non-Interactive Zero-Knowledge Based on a Single Random String. In *IEEE Symposium on Foundations of Computer Science*, pages 308-317, 1990.

[20] U. Feige and Shamir. Zero-Knowledge Proofs of Knowledge in Two Rounds. In *G. Brassard (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1989, LNCS 435*, pages 526-544. Springer-Verlag, 1989.

[21] U. Feige, A. Fiat and A. Shamir. Zero-knowledge Proof of Identity. *Journal of Cryptology*, 1(2): 77-94, 1988.

[22] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *A. Odlyzko (Ed.): Advances in Cryptology-Proceedings of CRYPTO'86, LNCS 263*, pages 186-194. Springer-Verlag, 1986.

[23] O. Goldreich. *Foundation of Cryptography-Basic Tools*. Cambridge University Press, 2001.

[24] O. Goldreich, S. Goldwasser and S. Micali. How to Construct Random Functions. *Journal of the Association for Computing Machinery*, 33(4):792–807, 1986.

[25] O. Goldreich, L. A. Levin and N. Nisan. On Constructing 1-1 One-Way Functions. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(29), 1995.

[26] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All language in $\mathcal{NP}$ Have Zero-Knowledge Proof Systems. *Journal of the Association for Computing Machinery*, 38(1): 691-729, 1991.

[27] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof-Systems In *ACM Symposium on Theory of Computing*, pages 291-304, 1985.

[28] L. Guillou and J. J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing both Transmission and Memory. In *C. G. Gnther (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 1988, LNCS 330* , pages 123-128, Springer-Verlag, 1988.

[29] J. Katz and R. Ostrovsky. Round-Optimal Secure Two-Party Computation. In *M. K. Franklin (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2004, LNCS 3152*, pages 335-354. Springer-Verlag, 2004.

[30] D. Lapidot and A. Shamir. Publicly-Verifiable Non-Interactive Zero-Knowledge Proofs. In *A.J. Menezes and S. A. Vanstone (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1990, LNCS 537*, pages 353-365. Springer-Verlag, 1990.

[31] S. Micali and L. Reyzin. Soundness in the Public-Key Model. In *J. Kilian (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2001, LNCS 2139*, pages 542–565. Springer-Verlag, 2001.

[32] S. Micali and L. Reyzin. Min-Round Resettable Zero-Knowledge in the Public-Key Model. In *B. Pfitzmann (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 2001, LNCS 2045*, pages 373–393. Springer-Verlag, 2001.

[33] M. Naor. Bit Commitment Using Pseudorandomness. *Journal of Cryptology*, 4(2): 151-158, 1991.

[34] M. Naor and O. Reingold. Number-Theoretic Constructions of Efficient Pseudo-Random Functions. *Journal of the ACM*, 1(2): 231-262 (2004).

[35] T. Okamoto. Provable Secure and Practical Identification Schemes and Corresponding Signature Schemes In *E. F. Brickell (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1992, LNCS 740*, pages 31-53. Springer-Verlag, 1992.

[36] C. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3): 24, 1991.

[37] Y. Zhao, X. Deng, C. H. Lee and H. Zhu. Resettable Zero-Knowledge in the Weak Public-Key Model. In *E. Biham (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 2003, LNCS 2656* , pages 123-140. Springer-Verlag, 2003.

Appendix A. Blum's Protocol for HC [6]

In the main text, we use the n-parallel repetitions of the following basic proof system for the directed Hamiltonian Cycle (HC) problem which is $\mathcal{NP}$-Complete.

Common input. A directed graph $G = (V, E)$ with $q = |V|$ nodes.

Prover's private input. A directed Hamiltonian cycle C_G in G.

Round-1. The prover selects a random permutation, π, of the vertices V, and commits (using a perfectly-binding commitment scheme) the entries of the adjacency matrix of the resulting permutated graph. That is, it sends a q-by-q matrix of commitments so that the $(\pi(i), \pi(j))^{th}$ entry is a commitment to 1 if $(i, j) \in E$, and is a commitment to 0 otherwise.

Round-2. The verifier uniformly selects a bit $b \in \{0, 1\}$ and sends it to the prover.

Round-3. If $b = 0$ then the prover sends π to the verifier along with the revealing of all commitments (and the verifier checks that the revealed graph is indeed isomorphic to G via π); If $b = 1$, the prover reveals to the verifier only the commitments to entries $(\pi(i), \pi(j))$ with $(i, j) \in C_G$ (and the verifier checks that all revealed values are 1 and the corresponding entries form a simple q-cycle).

Key generation. Let f_V be any OWF that is secure against $2^{N^{c_1}}$-time adversaries. On a security parameter N, each honest verifier V randomly selects an element x_V of length N, computes $y_V = f_V(x_V)$, publishes y_V as its public-key PK while keeping x_V as its secret-key SK. (If P uses Naor's OWF-based perfectly-binding commitment scheme in Phase-2 or Phase-4, V also deposits a random string R_V of length $3n$ as a part of its public-key, which serves the first-round message of Naor's commitment scheme.)

Common input. The public-file F, an element $x \in L \cap \{0,1\}^n$ and index i that specifies the i-th entry of F, PK_i. Note that the system security parameter is n.

P **private input.** An $\mathcal{NP}$-witness *wit* for $x \in L$, a pair of random strings (γ_1, γ_2), where γ_1 is a $poly(n)$-bit string and γ_2 is an n-bit string serving as the randomness seed of a PRF.

V **private input.** Private key SK_i such that $PK_i = f_V(SK_i)$.

Complexity-leverage used. Let c_1, $0 < c_1 < 1$, be the constant that the one-wayness of the OWF f_V, and the hiding property of the underlying perfectly-binding commitment scheme used by the verifier all hold against any circuit of size $2^{N^{c_1}}$ (which in turn guarantees that the WI property of the underlying WI protocol for $\mathcal{NP}$ executed in Stage-2 of Phase-1 and Phase-3, the hiding and trapdoorness properties of the underlying trapdoor commitment scheme all hold against any circuit of size $2^{N^{c_1}}$). *Note that the perfectly-binding commitment scheme used by the prover runs on a security parameter n and thus can be brute-force decommitted in time 2^n.* Let c_2 be the constant that: for all sufficiently large n's, both the length of the witness *wit* for $x \in L \cap \{0,1\}^n$ and the size of G_P (reduced from $(y_P^{(0)}, y_P^{(1)})$) are bounded by n^{c_2}. Then we set $\epsilon > c_2/c_1$ and $N = n^{\epsilon}$. This ensures that one can open the committed adjacency matrices (sent by prover in the underlying Blum's WI protocol for $\mathcal{NP}$ executed in Phase-2), enumerate all potential witnesses *wit*, or all potential Hamiltonian cycles of G_P in time $2^{n^{c_2}}$, which is still lesser than the time it would take to break the one-wayness of f_V, or the WI property of the underlying WI protocol for $\mathcal{NP}$ (executed in Stage-2 of Phase-1 and Phase-3), or the hiding and trapdoorness properties of the underlying trapdoor commitment scheme, because $2^{n^{c_2}} < 2^{N^{c_1}}$.

Phase-1. Phase-1 consists of two stages:

Stage-1. Let f_P be any OWF. On security parameter n, P randomly selects two elements $x_P^{(0)}$ and $x_P^{(1)}$ of length n each in the domain of f_P, computes $y_P^{(b)} = f_P(x_P^{(b)})$ for $b \in \{0,1\}$, reduces $(y_P^{(0)}, y_P^{(1)})$ to a directed graph G_P by Cook-Levin $\mathcal{NP}$-reduction such that finding a Hamiltonian cycle in G_P is equivalent to finding the preimage of either $y_P^{(0)}$ or $y_P^{(1)}$. For OWF-based solution, P also randomly selects a random string R_P of length $3N$ serving as the first-round message of Naor's OWF-based perfectly-binding commitment scheme. Finally, P sends $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)$ to V. The randomness used by P in this process is γ_1, *which means that the $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)$ is fixed once and for all.*

Stage-2. V first checks the validity of $(y_P^{(0)}, y_P^{(1)}, G_P)$ (i.e., whether or not G_P is reduced from $(y_P^{(0)}, y_P^{(1)})$). If it is valid, V randomly chooses two random strings $e_V^{(0)}$ and $e_V^{(1)}$ from $\{0,1\}^n$, computes $c_V^{(0)} = Com(1^N, R_P, e_V^{(0)})$ *by using the underlying perfectly-binding commitment scheme Com*, and $c_V^{(1)} = TCCom(1^N, (G_P, R_P), e_V^{(1)})$ by using the underlying trapdoor commitment scheme. Then, on common input (G_P, R_P, PK_i) V computes the first-round message, denoted a_V, of (n-parallel repetitions of) the Lapidot-Shamir or Blum's WI protocol for $\mathcal{NP}$ for showing the knowledge of either SK_i or a Hamiltonian cycle in G_P (equivalently, the preimage of either $y_P^{(0)}$ or $y_P^{(1)}$). Finally, V sends $(c_V^{(0)}, c_V^{(1)}, a_V)$ to P. *From then on, all randomness used by P in the remaining computation is got by applying $PRF(\gamma_2, \cdot)$ on the "determining" message $(x, F, c_V^{(0)}, c_V^{(1)}, a_V, PK_i, i)$.*

Phase-2. P proves to V that it knows a Hamiltonian cycle in G_P (equivalently, the preimage of either $y_P^{(0)}$ or $y_P^{(1)}$) by executing the (n-parallel repetitions of) Blum's WI protocol for $\mathcal{NP}$, in which V sends the assumed random challenge by just revealing $e_V^{(0)}$ committed to $c_V^{(0)}$. Note that the first-round message of Phase-2 (from P to V) consists of n committed adjacency matrices committed by running the underlying perfectly-binding commitment scheme *on security parameter n*. If P successfully finishes this phase and V accepts, then goto Phase-3. Otherwise, V aborts.

Phase-3. V and P continue the WI protocol for $\mathcal{NP}$ suspended at Stage-2 of Phase-1. If V successfully convinces P of the knowledge of either SK_i or a Hamiltonian cycle in G_P, then goto Phase-4. Otherwise, P aborts.

Phase-4. P proves to V that it knows either the witness *wit* for $x \in L$ or the preimage of PK_i, by executing the (n-parallel repetitions of) Blum's WI protocol for $\mathcal{NP}$ on common input (x, PK_i), in which V sends the assumed random challenge by just revealing $e_V^{(1)}$ committed to $c_V^{(1)}$.

Figure 4. Constant-round rZK-CS arguments for $\mathcal{NP}$ under minimal hardness assumptions

Common input. A directed graph $G = (V, E)$ with $|V| = n$. Note that the system security parameter is n.

P **private input.** A Hamiltonian cycle C_G in G, a random string γ that is an n-bit string serving as the randomness seed of a PRF. (For OWF-based implementation, P's randomness also consists of a random string R_P of length $3N$ that serves the first-round message of Naor's OWF-based perfectly-binding commitment scheme used by V.)

Complexity-leverage used. Let c_1, $0 < c_1 < 1$, be the constant that the hiding property of the underlying perfectly-binding commitment scheme used by the verifier holds against any circuit of size $2^{N^{c_1}}$. *Note that the perfectly-binding commitment scheme used by the prover runs in a security parameter n and thus can be brute-force decommitted in time* 2^n. Then we set $\epsilon > 1/c_1$ and $N = n^{\epsilon}$. This ensures that one can open the committed adjacency matrices (sent by prover in the first-round of the underlying Blum's protocol for HC), or enumerate all potential Hamiltonian cycles of G in time $poly(n) \cdot 2^n$, which is still lesser than the time it would take to break the hiding property of the underlying perfectly-binding commitment scheme used by the verifier that is run on security parameter N, because $2^n < 2^{N^{c_1}}$.

Phase-1. V randomly chooses a random string e_V from $\{0,1\}^n$, computes $c_V = Com(1^N, e_V)$ by using the underlying perfectly-binding commitment scheme Com that is run on security parameter N. (Note that if Com is Naor's OWF-based 2-round perfectly-binding commitment scheme, P also sends R_P on the top that is fixed once and for all.) If P uses Naor's OWF-based perfectly-binding commitment scheme in Phase-2, V also sends to P a random string R_V of length $3n$. That is, V sends (c_V, R_V) to P in this phase. *From then on, all randomness used by P in the remaining computation is got by applying $PRF(\gamma, \cdot)$ on the "determining" message* (G, c_V, R_V).

Phase-2. P proves to V that it knows a Hamiltonian cycle in G by executing the (n-parallel repetitions of) Blum's protocol for HC, in which V sends the second-round message (i.e., the assumed random challenge) by just revealing e_V committed to c_V. Note the first-round message from P to V consists of n committed adjacency matrices committed by running the underlying perfectly-binding commitment scheme *on security parameter n*.

Figure 5. Constant-round rWI arguments for $\mathcal{NP}$ with complexity-leveraging in the standard model

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'