# Generic and Practical Resettable Zero-Knowledge in the Bare Public-Key Model[⋆]

Moti Yung[1] and Yunlei Zhao[2]

[1] RSA Laboratories and Department of Computer Science, Columbia University, New York, NY, USA. moti@cs.columbia.edu

[2] Contact author. Software School, Fudan University, Shanghai 200433, China. ylzhao@fudan.edu.cn

**Abstract.** We present a generic construction for constant-round concurrently sound resettable zero-knowledge (rZK-CS) arguments for $\mathcal{NP}$ in the bare public-key (BPK) model under any (sub-exponentially strong) one-way function (OWF), which is a traditional assumption in this area. The generic construction in turn allows round-optimal implementation for $\mathcal{NP}$ still under general assumptions, and can be converted into a highly practical instantiation (under specific number-theoretic assumptions) for any language admitting $\Sigma$-protocols. Further, the rZK-CS arguments developed in this work also satisfy a weak (black-box) concurrent knowledge-extractability property as proofs of knowledge, in which case some super-polynomial-time assumption is intrinsic.

## 1 Introduction

Resettable zero-knowledge (rZK) is the strongest version of the remarkable notion of zero-knowledge (ZK) [13] to date. It was put forth by Canetti, Goldreich, Goldwasser and Micali [5], 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 (cZK) introduced by Dwork, Naor and Sahai [10].

A major measure of efficiency for interactive protocols is the round-complexity. Unfortunately, there are no constant-round rZK in the standard model, at least for the black-box case, as implied by the works of Canetti, Kilian, Petrank and Rosen [6]. To get constant-round rZK protocols, [5] introduced a simple model with very appealing trust requirement, the *bare public-key* (BPK) model.

A protocol in the BPK model simply assumes that all verifiers have deposited a public key in a public file before any interaction takes place among the users. (Actually, the BPK model also allows dynamic key registration with a reasonable amount time between key registration and key usage [5].) But, no

⋆ The second author is supported by 973 project No. 2007CB807900.

assumption is made on whether the public-keys deposited are unique or valid. That is: no trusted third party is assumed, preprocessing is reduced to users non-interactively posting public-keys in a public file, and the underlying communication network is assumed to be adversarially asynchronous. In many cryptographic settings, availability of a public key infrastructure (PKI) is assumed or required, in which case the BPK model that is weaker than PKI is natural.

Soundness in public-key models, when verifiers register public-keys, turns out to be more complicated and subtle than in other models as was shown by Micali and Reyzin. They 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 [16]. In this work, we focus on concurrent soundness, which roughly means that a malicious prover $P^*$ cannot convince the honest verifier $V$ of a *false* statement even when $P^*$ is allowed multiple interleaving interactions with $V$. They also showed that any (resettable or not) black-box ZK protocols with concurrent soundness in the BPK model (for non-trivial languages outside $\mathcal{BPP}$) must run at least four rounds [16]. The recent work of [21] formulates a new concurrent verifier security in the public-key model, named concurrent knowledge-extraction (CKE), and shows that CKE is strictly stronger than concurrent soundness in the public-key model when proofs of knowledge are considered.

A direct application of rZK is to achieve (smartcard based) identification schemes secure against resetting attacks [5, 2]. Despite its significant importance to practice, especially to smartcard based e-commerce over the Internet, most existing rZK systems are only theoretical feasible solutions which cannot be directly employed in practice and are not implementable by smartcards. That is, there is a gap between the significant importance and motivation for rZK as a mode suitable for practice and the present theoretical constructions of rZK systems. (Note that it is natural to investigate general feasibility prior to practical solutions.) Given the state of protocols, it is an important issue to develop highly practical rZK systems (say, with only a very small constant number of exponentiations, which are within reach for coming smartcard environments) for languages widely used in cryptography.

## 1.1 Our contributions

The main result of this work is a generic construction for constant-round concurrently sound rZK (rZK-CS) arguments for $\mathcal{NP}$ in the BPK model under any generic sub-exponentially strong OWF (sub-exponential assumptions in order to enable ZK protocols in this highly constrained resettable setting have been employed from the introduction of the model). The structure and techniques of the generic rZK-CS construction, in turn, allow round-optimal (still under general assumptions) and highly practical instantiation (under specific number-theoretic assumptions) implementations. Further, the rZK-CS arguments developed in this work also satisfy a weak (black-box) concurrent knowledge-extractability (CKE) property in the public-key model. (Roughly, a malicious prover not only cannot convince of a *false* statement by concurrent interactions as required by concurrent soundness, but also cannot convince of a true statement in its concurrent interactions without knowing a witness if the underlying language is sub-exponentially hard.) This answers several open problems left over in the field of round-efficient rZK in the BPK model [16, 22, 9].

Specifically, the generic construction allows the following round-optimal or highly practical implementations, which involve novel uses of a number of cryptographic tools:

– Round-optimal (i.e., 4-round) rZK-CS arguments for $\mathcal{NP}$ in the BPK model under any sub-exponentially strong one-way permutation (OWP) and any (standard polynomially secure) preimage-verifiable OWF. 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. This implies, in particular, that round-optimal rZK-CKE arguments for $\mathcal{NP}$ in the BPK model can be based on any certified one-way permutation.
– A *generic practical* transformation achieving 5-round rZK-CS arguments in the BPK model. By "generic" we mean applicability to any language that admits $\Sigma$-protocols. By "practical", we mean that the transformation does not go through general $\mathcal{NP}$-reductions, and if the starting $\Sigma$-protocol and the underlying pseudorandom function (PRF) are practical then the transformed rZK-CS arguments are also practical. For example, when instantiated with DL or RSA functions, together with the Naor-Reingold practical PRFs [18], the transformed rZK-CS arguments (for the languages of DL or RSA respectively) employ a very small constant number of exponentiations.

Discussions on related works are deferred to the full version.

## 2 Preliminaries

We briefly recall some basic definitions and tools, with detailed presentations deferred to the full version.

**Preimage-verifiable one-way functions.** A OWF $f$ is called preimage-verifiable if 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 $y = f(x)$.

**Statistically-binding commitment schemes.** We employ both the OWP based one-round perfectly-binding commitment scheme [12], and Naor's OWF-based 2-round scheme [17]. 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 part of a public-key in the public-key setting. *We remark that if the underlying OWP or OWF are secure against $2^{n^c}$-time adversaries for some constant $c$, $0 < c < 1$, on a security parameter $n$, then the hiding property of the corresponding commitment schemes above also holds against $2^{n^c}$-time adversaries.*

**Public-coin witness indistinguishability (WI) proof of knowledge (POK) systems for $\mathcal{NP}$.** One is Blum's protocol for directed Hamiltonian cycle DHC [3], and another is the Lapidot-Shamir protocol for DHC [15]. The salient feature of the Lapidot-Shamir protocol is that the prover sends the first-round message without knowing the statement to be proved other than its size. We remark that the WI property of Blum's protocol or the Lapidot-Shamir protocol for HC relies on the hiding property of the underlying statistically-binding commitment scheme (used in its first-round). If the hiding property of the underlying statistically-binding commitment scheme is secure against $2^{n^c}$-time adversaries for some constant $c$, $0 < c < 1$, on a security parameter $n$, then the WI property also holds against $2^{n^c}$-time adversaries.

**Trapdoor commitment schemes.** Normal trapdoor commitment schemes run in two rounds, in which the commitment receiver generates and sends the trapdoor commitment public key ($TCPK$) in the first-round (while keeping the trapdoor secret key $TCSK$ in private). For the Feige-Shamir trapdoor commitment scheme (FSTC) [11], $TCPK$ consists of $(y = f(x), G)$ (for OWF-based solution, the $TCPK$ also includes a random string $R$ serving as the first-round message of Naor's OWF-based statistically-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 $G$). Note that the first-round message, i.e., $TCPK$, can be fixed once and for all. The commitment sender forms the second-round message by using (either OWP-based one-round or Naor's OWF-based two-round) statistically-binding commitment scheme. Again, if the hiding property of the underlying statistically-binding commitment scheme is secure against sub-exponential-time adversaries, then both the hiding property and the trapdoorness property of the FSTC scheme hold also against sub-exponential-time adversaries.

**$\Sigma$-protocols and $\Sigma_{OR}$-protocols.** Informally, a $\Sigma$-protocol is itself a 3-round public-coin *special* honest verifier zero-knowledge (SHVZK) protocol with special soundness in the knowledge-extraction sense. A very large number of $\Sigma$-protocols have been developed in the literature. One basic construction with $\Sigma$-protocols is the OR of a real and simulated transcript, called $\Sigma_{OR}$, that 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 [7] (i.e., witness indistinguishable WI). For a good survey of $\Sigma$-protocols and their applications, the reader is referred to [8].

**The malicious resetting verifier and rZK in the BPK model.** A malicious $s$-resetting malicious verifier $V^*$ in the BPK model, 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 $\bar{\mathbf{x}} = \{x_1, \cdots, x_{s(n)}\}$ of equal length $poly(n)$ 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 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 1 (black-box resettable zero-knowledge [5]).** *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 $\bar{\boldsymbol{x}} = \{x_1, \cdots, x_{s(n)}\}$, $x_i \in L \cap \{0,1\}^{poly(n)}$ for $1 \leq i \leq s(n)$, and their corresponding NP-witnesses $aux(\bar{\boldsymbol{x}}) = \{w_1, \cdots, w_{s(n)}\}$:*

**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 $S(\bar{\boldsymbol{x}})$.*

**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 $P(x_i, w_i, F, id_j, \gamma_k)$ in multiple protocols (sessions). We remark that, as clarified in [5], in the resettable setting interleaving interactions do not help the malicious resetting 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.

## 3 The Generic rZK-CS Construction

**The high-level overview of the protocol.** We first convey basic ideas and a high-level overview of the protocol. Let $f_V$ be any (*sub-exponentially strong*) OWF, each (honest) verifier $V$ randomly selects an element $x_V$ from the domain of $f_V$, and publishes $y_V = f_V(x_V)$ as its public-key with $x_V$ as its secret-key. Let $L$ be an $\mathcal{NP}$-language and $x \in L$ be the common input, the main-body of the protocol goes as follows: The honest prover $P$ first generates and sends a hard-instance using a standard *polynomially-secure* OWF $f_P$. The hard-instance is then fixed once and for all. Then, $P$ proves to $V$ the existence of the preimage of the hard-instance, by executing a OWF-based resettable witness-hiding (rWH) protocol. After that, $V$ proves to $P$ that it knows either the preimage of $y_V$ (i.e., its secret-key $x_V$) or the preimage of the hard-instance generated by $P$, by executing a OWF-based constant-round WIPOK protocol for $\mathcal{NP}$. Finally, $P$ proves to $V$ that it knows either a witness for $x \in L$ or the preimage of $y_V$ (i.e., $V$'s secret-key), by executing another OWF-based constant-round rWI argument for $\mathcal{NP}$. The detailed protocol description is depicted in Figure 1 (page 6).

**The underlying complexity-leveraging.** For provable security and for the weak CKE security, we employ the complexity-leveraging technique (originally introduced in [5]). Specifically, the verifier $V$ uses a security parameter $N$ (in

**Key generation.** On the system 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 statistically-binding commitment scheme in Phase-2 or Phase-4 (that is run on security parameter $n$), $V$ also deposits a random string $R_V$ of length $3n$.

**Common input.** An element $x \in L \cap \{0,1\}^{poly(N)}$, the public-file $F$ and an index $j$ that specifies the $j$-th entry of $F$, i.e., $PK_j = (y_V^{(j)}, R_V^{(j)})$.

$P$ **private input.** An $\mathcal{NP}$-witness $w$ for $x \in L$, a pair of random strings $(\gamma_1, \gamma_2)$, where $\gamma_1$ is a $poly(n)$-bit string and $\gamma_2$ is the $n$-bit randomness seed of a PRF.

$V$ **private input.** $SK_j$. For presentation simplicity, we denote $PK_j = f_V(SK_j)$.

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

**Stage-1.** Let $f_P$ be any polynomially-secure 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 string $R_P$ of length $3N$ serving as the first-round message of Naor's OWF-based statistically-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 $\gamma_1$, *which means* $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)$ *is fixed once and for all.*

**Stage-2.** $V$ first checks whether or not $G_P$ is reduced from $(y_P^{(0)}, y_P^{(1)})$ and $R_P$ is of length $3N$. If the checking is successful, $V$ 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 statistically-binding commitment scheme* $Com$, and $c_V^{(1)} = TCCom(1^N, (G_P, R_P), e_V^{(1)})$ by using the underlying FSTC trapdoor commitment scheme. Then, on common input $((y_P^{(0)}, y_P^{(1)}, G_P, R_P), PK_j)$ $V$ computes the first-round message, denoted $a_V$, of ($n$-parallel repetitions of) Blum's WIPOK for $\mathcal{NP}$ for showing the knowledge of either $SK_j$ 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* $D = (x, F, (j, PK_j), (y_P^{(0)}, y_P^{(1)}, G_P, R_P), (c_V^{(0)}, c_V^{(1)}, a_V))$.

**Phase-2.** $P$ proves to $V$ the existence of a Hamiltonian cycle in $G_P$ by executing the ($n$-parallel repetitions of) Blum's WI protocol for $\mathcal{NP}$ on $(y_P^{(0)}, y_P^{(1)}, G_P, R_V^{(j)})$, 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 statistically-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 WIPOK protocol for $\mathcal{NP}$ suspended at Stage-2 of Phase-1. If $V$ successfully convinces $P$ of the knowledge of either $SK_j$ or a Hamiltonian cycle in $G_P$, then goto Phase-4. Otherwise, $P$ aborts. We denote by $e_V$, $z_V$, the first-round message and the second-round message of Phase-3 respectively.

**Phase-4.** $P$ proves that it "knows" either the witness $w$ for $x \in L$ or the secret-key $SK_j$, by executing Blum's WI protocol for $\mathcal{NP}$ on common input $(x, PK_j)$, in which $V$ sends the assumed random challenge by just revealing $e_V^{(1)}$ committed to $c_V^{(1)}$.

**Fig. 1.** The generic rZK-CS argument $\langle P, V\rangle$ for $\mathcal{NP}$

generating messages from it) that is also the system security parameter. But, the prover $P$ uses a relatively smaller security parameter $n$ (still polynomially related to $N$). The justification and discussions of the complexity-leveraging technique are given in [5]. *Here, we additionally remark that, pragmatically speaking, letting the verifier and the prover use different security parameters is quite reasonable in the resettable setting, in which the prover is implemented by smart-cards or clients that have relatively limited computational resources and power and the verifier is normally implemented by servers that have much more computational resources and power.*

Specifically, the security parameters are set as follows. On the system parameter $N$, suppose $f_V$ is secure against $2^{N^{c_V}}$-time adversaries for some constant $c_V$, $0 < c_V < 1$. This implies that the hiding property of the underlying statistically-binding commitment scheme used by the verifier holds also against any $2^{N^{c_V}}$-time adversary, 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, and the hiding and trapdoorness properties of the underlying trapdoor commitment scheme all hold against any $2^{N^{c_V}}$-time adversary. The prover uses a relatively smaller security parameter $n$ and uses a standard polynomially-secure OWF $f_P$ that can be broken (brute-force wise) in time $2^{n^{c_P}}$ for some constant $c_P$, $c_P \geq 1$. Specifically, $c_P$ is the constant that: for all sufficiently large $n$'s, the size of $G_P$ (reduced from $(y_P^{(0)}, y_P^{(1)})$ at Stage-1 of Phase-1) is bounded by $n^{c_P}$, which in turn implies that the statistically-binding commitment scheme used by the prover (that is run on the security parameter $n$) can be brute-force decommitted in time $poly(n) \cdot 2^{n^{c_P}}$. Let $c_L$, $0 < c_L \leq 1$, be a constant specific to the underlying language $L$ (the use of $c_L$ is specified in Section 3.1 for the weak CKE property). Let $c$ be any constant such that $0 < c < \min\{c_V, c_L\}$, in other words, $\min\{c_V, c_L\} = c + c'$ for another constant $c'$, $0 < c' < 1$. Let $\varepsilon$ be any constant such that $\varepsilon > \frac{c_P}{c}$, then we set $N = n^{\varepsilon}$. Note that $N$ and $n$ are polynomially related. That is, any quantity that is a polynomial of $N$ is also another polynomial of $n$. This complexity leveraging guarantees that although any $poly(n) \cdot 2^{n^{c_P}}$-time adversary can break $f_P$ on a security parameter $n$, it is still infeasible to break the one-wayness of $f_V$, because $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^c} \ll 2^{N^{c_V}}$ (also note that $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_L}}$).

The OWF-based protocol depicted in Figure 1 (page 6) runs in 7 rounds after some round combinations. In particular, the first two rounds of Phase-4 can be combined into previous phases. Actually, the round-complexity can be further reduced to 6 but under any (sub-exponentially strong) OWP.

**Theorem 1.** *Assuming the OWF $f_P$ (used by the prover) is secure against standard polynomial-time adversaries, and the OWF $f_V$ (used by the verifier) is secure against sub-exponential-time adversaries, the protocol depicted in Figure 1 is a constant-round rZK-CS argument for $\mathcal{NP}$ in the BPK model.*

**Proof (sketch).**

**Black-box resettable zero-knowledge.**

For any $s$-resetting adversary $V^*$ who receives $s(N)$ *distinct* strings $\bar{\mathbf{x}} = \{x_1, \cdots, x_{s(N)}\}$, $x_i \in L \cap \{0,1\}^{poly(N)}$ for each $i$ $(1 \leq i \leq s(N))$, 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 $PK_j$ in $F$, $1 \leq j \leq s(N)$, is "covered" if the rZK simulator $S$ has already learned (extracted) the corresponding secret-key $SK_j$ (if such exists). In its second stage, $V^*$ is given oracle access to $(s(N))^3$ prover instances $P(x_i, PK_j, \gamma_k)$, $1 \leq i, j, k \leq s(N)$. We denote by $D_t = (x_i, F, (j, PK_j), (y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k, (c_{V^*}^{(0)}, c_{V^*}^{(1)}, a_{V^*})^t)$ the "determining" message of the $t$-th session with respect to common input $x_i$ and public-key $PK_j$ and the honest prover instance $P(\cdot, \cdot, \gamma_k)$, $1 \leq i, j, k \leq s(N)$ and $1 \leq t \leq (s(N))^3$. As discussed in [5], w.l.o.g., we use the convention that $V^*$ works in the sequential version in its second stage, and the rZK simulator utilizes a truly random function rather than a pseudorandom one.

The rZK simulation procedure is similar to, but more complicated than, that of [5]. Specifically, the rZK simulator $S$ runs $V^*$ as a subroutine, and works in at most $s(N) + 1$ phases such that in each phase it either successfully finishes its simulation or "covers" a new public-key in $F$. In each phase, $S$ makes a simulation attempt from scratch with a new truly random function that is to be defined adaptively, and works session by session sequentially in at most $(s(N))^3$ sessions. The difficulties lie in that for such rZK simulation to be successful, the rZK simulator $S$ needs to have the ability to cover new uncovered public-keys within time inversely proportional to the probability that it encounters a success of Phase-3 relative to a yet uncovered public-key in its simulation. Pending on $S$'s such ability, the rZK property follows from the pseudorandomness of PRF and the rWI property of Phase-4 combined with Phase-1 (according to the CGGM general paradigm for achieving rWI [5]).

Specifically, we want to argue that the underlying Blum's WIPOK protocol on $((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k, PK_j)$ (executed in Stage-2 of Phase-1 and Phase-3) is actually an *argument of knowledge* of the preimage of $PK_j$ (i.e., the secret-key $SK_j$). But, the subtle and complicated situation here is that before $V^*$ finishes Phase-3, $S$ has already proved the knowledge of the Hamiltonian cycle of $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ in Phase-2. Note that the $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ is fixed once and for all (*that can be viewed as the public-key of the honest prover instance* $P(\cdot, \cdot, \gamma_k)$), and furthermore $V^*$ is resettingly (more than concurrently) interacting with the honest prover instances. As demonstrated in [21], normal argument of knowledge and even concurrent soundness do not guarantee *correct* knowledge-extractability in such setting. In particular, one may argue that, by rewinding the honest prover instances arbitrarily, $V^*$ may potentially malleate the interactions on $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ provided by the honest prover in Phase-2 of one session into successful but "false" interactions on $((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k, PK_j)$ 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 $SK_j$. This means that, in such a case the interactions on $((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k, PK_j)$ executed in Phase-3 together with Stage-2 of Phase-1 are no longer *arguments of knowledge* of the preimage of $PK_j$, although it is always a system for proof of knowledge of either $SK_j$ or a

Hamiltonian cycle of $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$. What save us here is the (concurrent) 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, on common inputs $\{(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^1, \cdots, (y_P^{(0)}, y_P^{(1)}, G_P, R_P)^{s(N)}\}$ $\hat{S}$ concurrently interacts with $s(N)$ instances of the knowledge prover, denoted $\hat{P}$, of Blum's protocol for HC by playing the role of knowledge verifier. We denote each of the $s(N)$ instances of $\hat{P}$ by $\hat{P}((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k)$, $1 \leq k \leq s(N)$; At the same time, $\hat{S}$ runs the $s$-resetting malicious $V^*$ as a subroutine by playing the role of the honest prover, and sends $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ as the Stage-1 message of Phase-1 whenever $V^*$ initiates a session with the honest prover instance $P(\cdot, \cdot, \gamma_k)$. $\hat{S}$ emulates the rZK simulator $S$ but with the following modification: whenever $\hat{S}$ needs to send a "fresh" first-round message of Blum's protocol for HC on $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ in Phase-2 with respect to a "determining" message, it initiates a new session with $\hat{P}((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k)$, and forwards the first-round message received from $\hat{P}((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k)$ to $V^*$. This "fresh" message happens due to either $V^*$ sends a distinct "determining" message in one session or $\hat{S}$ needs rewinding $V^*$ and redefining the underlying random function $f$ to extract knowledge used by $V^*$ in a successful execution of Stage-2 of Phase-1 and Phase-3 with respect to an uncovered public-key. Then, $\hat{S}$ runs $V^*$ further, and in case $V^*$ successfully reveals the assumed challenge (that is *statistically-bindingly* committed to the underlying "determining" message in question) then $\hat{S}$ returns back the revealed challenge to $\hat{P}$ as its own challenge in the corresponding simultaneous session of Blum's protocol for HC, and returns back the third-round message received from $\hat{P}((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k)$ to $V^*$. For a session with a "determining" message that is identical to that of some previous sessions, $\hat{S}$ just copies what was sent in the previous sessions. Note that in this case, $\hat{S}$ may still possibly need to interact with $\hat{P}$ in some *existing* concurrent session to get some third-round message (in case $V^*$ did not reveal or invalidly revealed the random challenge statistically-bindingly committed to the underlying "determining" message in all previous sessions, but correctly reveals it in the current session). However, the key point here is that in this case $S$ 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 $S$ generates $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$'s and provides the corresponding Phase-2 messages by itself (rather than get them by externally interacting with the knowledge prover instances $\hat{P}((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k)$'s). The key observation here is that although $V^*$ is actually resettingly interacting with $\hat{S}$, $\hat{S}$ only concurrently interacts with the instances of $\hat{P}$ and never rewinds $\hat{P}$. *The underlying reason is just that in any session, Phase-2 interactions take place only after $V^*$ sent the "determining" message at Stage-2 of Phase-1 that determines the subsequent behaviors of $V^*$ in that session.* Note that in this case, the (concurrent) WI property of the Blum's protocol for HC on common input $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$

actually implies witness hiding (WH), which means that no PPT algorithm can output a Hamiltonian cycle in $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ even by concurrently interacting with $\hat{P}((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k)$'s. Also note that on common input $((y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k, PK_j)$, Phase-3 together with Stage-2 of Phase-1 is always a system for proving the knowledge of either a Hamiltonian cycle in $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ or the preimage of $PK_j$ (i.e., $SK_j$), which means that with overwhelming probability $\hat{S}$ (or the real rZK simulator $S$) can always extract either a Hamiltonian cycle in $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ or the corresponding secret-key $SK_j$ within time inversely proportional to the probability that $V^*$ successfully finishes Phase-3 (by rewinding $V^*$ and redefining the underlying random function as is done in [5]). But, the WH property of Blum's protocol for HC shows that with overwhelming probability, $\hat{S}$ (or the real rZK simulator $S$) never outputs a Hamiltonian cycle in $(y_P^{(0)}, y_P^{(1)}, G_P, R_P)^k$ in its simulation that is done in expected polynomial-time. Here, a subtle point needs to be further addressed. Specifically, the normal WH property is defined with respect to probabilistic (strict) polynomial-time algorithms, but here $\hat{S}$ works in expected polynomial-time. But, by Markov inequality, it is easy to see that if the WH property of a protocol holds with respect to any strict polynomial-time algorithms, then it also holds with respect to any expected polynomial-time algorithms.

**Concurrent soundness.**

We show that for any (whether true or not) common input $x \in \{0,1\}^{poly(N)}$, if a PPT $s$-concurrent malicious $P^*$, on a public-key $PK$, can convince an honest verifier $V$ (with public-key $PK$ and secret-key $SK$) of the statement "$x \in L$" with non-negligible probability $p_x$ in one of the $s(N)$ concurrent interactions, then there exists an algorithm $E$ that, on the *same* public-key $PK$ with oracle accessing $P^*$, works in $poly(n) \cdot 2^{n^{c_P}}$-time and outputs either a witness for $x \in L$ or the preimage of $y_V$ also with non-negligible probability. Note that according to the underlying complexity leveraging on the security parameters $N$ and $n$, no $poly(n) \cdot 2^{n^{c_P}}$-time algorithm can break the one-wayness of $f_V$ used by $V$ in forming its public-key on security parameter $N$ (because $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_V}}$). This implies that $x \in L$.

On the same public-key $PK$, $E$ runs $P^*$ as a subroutine by playing the role of the honest verifier with public-key $PK$. *Note that $E$ does not know the corresponding secret-key $SK$.* In each session $t$, $1 \le t \le s(N)$, after receiving the Stage-1 message of Phase-1, denoted $((y_{P^*}^{(0)}, y_{P^*}^{(1)})^t, G_{P^*}^t, R_{P^*}^t)$, $E$ first checks whether or not $G_{P^*}^t$ is $\mathcal{NP}$-reduced from $(y_{P^*}^{(0)}, y_{P^*}^{(1)})^t$ and $R_{P^*}^t$ is of length $3N$. If the checking is successful, then $E$ tries to find a Hamiltonian cycle in $G_{P^*}^t$ by brute-force searching in $2^{n^{c_P}}$-time.

– If $E$ finds a Hamiltonian cycle in $G_{P^*}^t$, then $E$ sets the Stage-2 message of Phase-1 of the $t$-th session, denoted $((c_V^{(0)})^t, (c_V^{(1)})^t, a_V^t)$, as follows: it randomly chooses one random string $(e_V^{(0)})^t$ from $\{0,1\}^n$, computes $(c_V^{(0)})^t = Com(1^N, R_{P^*}^t, (e_V^{(0)})^t)$ by using the underlying Naor's statistically-binding

commitment scheme $Com$, and computes $(c_V^{(1)})^t = TCCom(1^N, (G_{P*}^t, R_{P*}^t), 0^n)$ by using the underlying Feige-Shamir trapdoor commitment scheme (note that, $(c_V^{(1)})^t$ commits to $0^n$ rather than a random string in $\{0,1\}^n$ as the honest verifier does). Then, on common input $(((y_{P*}^{(0)}, y_{P*}^{(1)})^t, G_{P*}^t, R_{P*}^t), PK)$ $E$ computes the first-round message, denoted $a_V^t$, of ($n$-parallel repetitions of) Blum's WIPOK for $\mathcal{NP}$ for showing the knowledge of either $SK$ or a Hamiltonian cycle in $G_{P*}^t$. *Note that the first-round message of Blum's WIPOK for $\mathcal{NP}$ is computed without using any witness knowledge (i.e., either $SK$ or a Hamiltonian cycle in $G_{P*}^t$)*; In case $P^*$ successfully finishes Phase-2 of the $t$-th session, $E$ moves into Phase-3. After receiving the first-round message of Phase-3 of the $t$-th session, denoted $e_V^t$, $E$ computes the second-round message of Phase-3, denoted $z_V^t$ (i.e., the third-round message of Blum's WIPOK for showing the knowledge of either $SK$ or a Hamiltonian cycle in $G_{P*}^t$), *by using the extracted Hamiltonian cycle in $G_{P*}^t$ as its witness*; Finally, in Phase-4 of the $t$-th session, $E$ decommits $(c_V^{(1)})^t$ to a random string $(e_V^{(1)})^t$ of length $n$, *by using the extracted Hamiltonian cycle in $G_{P*}^t$ as the trapdoor.*
– If there exists *no* Hamiltonian cycle in $G_{P*}^t$, then $E$ sets and sends the Stage-2 message of Phase-1 of the $t$-th session, i.e., $((c_V^{(0)})^t, (c_V^{(1)})^t, a_V^t)$, just as above. But, whenever $P^*$ successfully finishes Phase-2 of the $t$-th session and sends to $E$ the first-round message of Phase-3 of the $t$-th session (i.e., $e_V^t$), $E$ aborts with an error message (as it has no witness for generating the next message).

Whenever $P^*$ stops, $E$ also stops and outputs the simulated transcript $str$ (i.e., the view of $P^*$ interacting with $E$). Denote by $view_{P^*}^{E(PK)}(1^n, PK)$ the view of $P^*$ (i.e., $str$) in the above run of $E(1^n, PK)$. We first establish that the simulated transcript is indistinguishable from the view of $P^*$ in real execution with honest verifier instances. The purpose of $E$ is to extract witnesses to all accepting sessions in $str$, which will be demonstrated later.

**Lemma 1.** *For any sufficiently large $n$, and for all (except for a negligible fraction of) $(PK, SK)$ outputted by the key-generation stage of the honest verifier, the view of $P^*$ in the run of $E(1^n, PK)$ (i.e., $view_{P^*}^{E(PK)}(1^n, PK)$) is indistinguishable from the view of $P^*$ in real execution with honest verifier instances.*

**Proof. (of Lemma 1)** This is done by establishing a series of hybrid experiments.

We first consider a mental experiment in which $P^*$ concurrently interacts with an imaginary verifier $\widehat{V}$ with the same public-key $PK$ and secret-key $SK$. $\widehat{V}$ mimics the real honest verifier $V$ with public-key $PK$ and secret-key $SK$ but with the following modifications: For any session $t$, $1 \leq t \leq s(N)$, in case $P^*$ successfully finishes Phase-2 and sends to $\widehat{V}$ the first-round message of Phase-3, $\widehat{V}$ enumerates all possible Hamiltonian cycles of $G_{P*}^t$ by brute-force searching in $2^{n^{c_P}}$-time, where $((y_{P*}^{(0)}, y_{P*}^{(1)})^t, G_{P*}^t, R_{P*}^t)$ is the Stage-1 message of Phase-1 of the $t$-th session. If there exists *no* Hamiltonian cycle in $G^t_{P*}$, $\widehat{V}$ aborts with an error message, *although it can continue the execution with $SK$ as its witness!*

Note that the only difference between the interactions between $P^*$ and $\widehat{V}$ and the interactions between $P^*$ and the real honest verifier $V$ is that: for any session $t$, $1 \leq t \leq s(N)$, the real honest verifier always continues the execution of Phase-3 by using $SK$ as its witness in forming the second-round message of Phase-3, in case $P^*$ successfully finished Phase-2 and sent the first-round message of Phase-3; but $\widehat{V}$ may abort in this case if it finds that $G^t_{P^*}$ is "false" (i.e. there exists no Hamiltonian cycle in $G^t_{P^*}$) by brute-force searching in $2^{n^{c_P}}$-time. That the view of $P^*$ interacting with $\widehat{V}$ is indistinguishable from its view in real execution with honest verifier instances is from the following lemma.

**Lemma 2.** *For all positive polynomials $s(\cdot)$ and all $s$-concurrent malicious $P^*$, the probability that there exists a $t$, $1 \leq t \leq s(N)$, such that $P^*$ can successfully finish Phase-2 with respect to a false $G^t_{P^*}$ (i.e., $G^t_{P^*}$ contains no Hamiltonian cycle) in the $t$-th session of the $s(N)$ concurrent sessions (against the real honest verifier $V$ with public-key $PK$) is negligible in $n$.*

**Proof (of Lemma 2).** We show that if a PPT $s$-concurrent adversary $P^*$ can convince $V$ (with public-key $PK$) of a false $G^t_{P^*}$ with non-negligible probability $p'(n)$ in Phase-2 of one of the $s(N)$ concurrent sessions, then this will violate the hiding property of the underlying *statistically-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 statistically-binding commitment scheme $Com$, given two strings $\hat{e}_0$ and $\hat{e}_1$ that are taken uniformly at random from $\{0,1\}^n$ and $C = Com(1^N, R^t_{P^*}, \hat{e}_b)$ for a randomly chosen bit $b \in \{0,1\}$, no $2^{N^{c_V}}$-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., $SK$) as its non-uniform inputs.

We construct a (non-uniform) algorithm $A$ that takes $(1^n, (\hat{e}_0, \hat{e}_1, SK), C)$ as input and attempts 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 $j$ from $\{1, \cdots, s(N)\}$, runs $P^*$ as a subroutine by playing the role of the honest verifier $V$ with secret-key $SK$ in any session other than the $j$-th session. In the $j$-th session, after receiving $G^j_{P^*}$ from $P^*$ at Stage-1 of Phase-1, $E$ first checks whether there exists a Hamiltonian cycle in $G^j_{P^*}$ or not by brute-force searching in time $2^{n^{c_P}}$. If $E$ finds a Hamiltonian cycle in $G^j_{P^*}$, then $E$ randomly guesses the bit $b$ and stops. Otherwise (i.e., there exists no Hamiltonian cycle in $G^j_{P^*}$), $E$ runs $P^*$ further and continues the interactions of the $j$-th session as follows: $E$ gives $C$ to $P^*$ as the assumed commitment to $(e^{(0)}_V)^j$ at Stage-2 of Phase-1. After receiving the first-round message of Phase-2 (i.e., the first-round of Blum's protocol for proving the existence of a Hamiltonian cycle in $G^j_{P^*}$) that contains $n$ committed adjacency matrices, $E$ first opens all the committed adjacency matrices by brute-force in

$poly(n) \cdot 2^{n^{c_P}}$-time (note that $E$ can do this since the underlying statistically-binding commitment scheme used by the prover in forming these $n$ committed adjacency matrices is run on security parameter $n$). For each revealed graph $G_k^j$ ($1 \leq k \leq n$) (described by the corresponding opened adjacency matrix entries) we say that $G_k^j$ is a 0-valid graph if it is isomorphic to $G_{P*}^j$, or a 1-valid graph if it contains a Hamiltonian cycle of the same size of $G_{P*}^j$. We say that the set of revealed graphs $\{G_1^j, \cdots, G_n^j\}$ is $\hat{e}_b$-valid ($b \in \{0,1\}$) if for all $k$, $1 \leq k \leq n$, $G_k^j$ 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 $\{G_1^j, \cdots, G_n^j\}$, $E$ can determine whether it is $\hat{e}_0$-valid or $\hat{e}_1$-valid in time $poly(n) \cdot 2^{n^{c_P}}$. Then, $E$ outputs 0 if the set $\{G_1^j, \cdots, G_n^j\}$ is $\hat{e}_0$-valid but not $\hat{e}_1$-valid. Similarly, $E$ outputs 1 if the set $\{G_1^j, \cdots, G_n^j\}$ 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 $G_{P*}^j$ 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 $G_{P*}^j$, the set of revealed graphs $\{G_1^j, \cdots, G_n^j\}$ 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 $G_{P*}^j$ with probability $1-2^{-n}$ the set of revealed graphs $\{G_1^j, \cdots, G_n^j\}$ is not $\hat{e}_{1-b}$-valid (since $\hat{e}_{1-b}$ is taken uniformly at random from $\{0,1\}^n$). Since the value $j$ is randomly chosen from $\{1, \cdots, s(N)\}$, we conclude that $E$ can successfully guess the bit $b$ with probability at least $(1-2^{-n}) \cdot \frac{p'(n)}{s(N)} + \frac{1}{2}(1 - \frac{p'(n)}{s(N)}) = \frac{1}{2} + \frac{1}{2} \cdot \frac{p'(n)}{s(N)} - 2^{-n} \cdot \frac{p'(n)}{s(N)}$ in time $poly(n) \cdot 2^{n^{c_P}}$. That is, $E$ successfully guesses the bit $b$ with non-negligible advantage over 1/2 in time $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_V}}$, which violates the hiding property of the underlying statistically-binding commitment scheme $Com$ used by $V$ that is run on the security parameter $N$. This finishes the proof of Lemma 2.

Now, we want to show that the view of $P^*$ with $\widehat{V}$ is indistinguishable the view of $P^*$ with $E$. This is established by conducting another hybrid experiment.

Specifically, we consider the following hybrid experiment. An algorithm $\widehat{E}$ takes $(PK, SK)$ as its input (that is, $\widehat{E}$ takes both the verifier's public-key and the corresponding secret-key as its input), and runs $P^*$ as a subroutine by mimicking the knowledge-extractor $E$ (who only takes $PK$ as input) but with the following modification: For any session $t$, $1 \leq t \leq s(N)$, in case $P^*$ successfully finishes Phase-2 and sends to $\widehat{E}$ the first-round message of Phase-3, $\widehat{E}$ enumerates all possible Hamiltonian cycles of $G_{P*}^t$ by brute-force searching in $2^{n^{c_P}}$-time, where $((y_{P*}^{(0)}, y_{P*}^{(1)})^t, G_{P*}^t, R_{P*}^t)$ is the Stage-1 message of Phase-1 of the $t$-th session. If there *exists* a Hamiltonian cycle in $G_{P*}^t$, then $\widehat{E}$ continues the execution by forming the second-round message of Phase-3 of the $t$-th session (for showing the knowledge of either $SK$ or a Hamiltonian cycle of $G_{P*}^t$) *but using $SK$ as its witness just as the real honest verifier does* (note that in this case $E$ continues the execution with the extracted Hamiltonian cycle of $G_{P*}^t$ as the corresponding witness). If there exists *no* Hamiltonian cycle in $G_{P*}^t$, then $\widehat{E}$ aborts with an error message just as $E$ (or $\widehat{V}$) does (*although in this case $\widehat{E}$ can continue the execution with $SK$ as its witness*).

Note that the difference between the interactions between $P^*$ and the imaginary verifier $\widehat{V}$ in the first hybrid experiment and the interactions between $P^*$ and $\widehat{E}$ is that: in any session $t$, $1 \leq t \leq s(N)$, of the interactions between $P^*$ and $\widehat{V}$, $\widehat{V}$ always commits (and accordingly decommits to) a random string of length $n$ (i.e., $(e_V^{(1)})^t$) by using the underlying FSTC scheme (just as the honest verifier $V$ does), but in the interactions between $P^*$ and $\widehat{E}$, $\widehat{E}$ always commits $0^n$ and then decommits to a random string of length $n$ by using the brute-force extracted Hamiltonian cycle of $G^t_{P^*}$ as the trapdoor (just as $E$ does). We show the view of $P^*$ with $\widehat{E}$ is indistinguishable from the view of $P^*$ with $\widehat{V}$. Otherwise, by hybrid arguments, we can construct a $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_V}}$-time algorithm that breaks the hiding and trapdoorness properties of FSTC.

The difference between the interactions between $P^*$ and $\widehat{E}$ and the interactions between $P^*$ and $E$ is that: $E$ always uses the brute-force extracted Hamiltonian cycle of $G^t_{P^*}$ as its witness in Phase-3 of any session $t$, $1 \leq t \leq s(N)$, but $\widehat{E}$ always uses the verifier's secret-key $SK$ as its witness (just as the honest verifier does). Similarly, the view of $P^*$ with $\widehat{E}$ is indistinguishable from the view of $P^*$ with $E$ is indistinguishable. Otherwise, by hybrid arguments, we can break the WI property of Blum's protocol for $\mathcal{NP}$ in time $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_V}}$. This finishes the proof of Lemma 1.

Now, $E$ wants to extract the corresponding witness to each accepting session in the simulated transcript $str$. For any $t$, $1 \leq t \leq s(N)$, suppose the $t$-session is accepting in $str$, we define an experiment $E_t$ that emulates $E$ with the fixed random coins of $E$, but with the following exception: the random $n$-bit string $(e_V^{(1)})^t$ (i.e., the decommitted value to $(c_V^{(1)})^t$) is no longer emulated internally, but received externally. Note that the experiment $E_t$ actually amounts to the *stand-alone* execution of the Blums's WIPOK of Phase-4 on common input $x_t$ between a (stand-alone) sub-exponential-time prover (combining all internal emulation of $E$ with running $P^*$ as the subroutine, except for $(e_V^{(1)})^t$ to be received externally) and a public-coin honest verifier that sends $(e_V^{(1)})^t$. By applying the stand-alone knowledge-extractor on $E_t$, except for the probability $2^{-n}$ we can get one of the following within time $poly(n) \cdot 2^n \ll 2^{n^{c_P}}$ (actually, within expected polynomial-time): a witness $w_t$ for $x_t \in L$ or the corresponding secret-key $SK$ such that $PK = f_V(SK)$. As $E_t$ runs in $poly(n) \cdot 2^{n^{c_P}}$-time, we conclude that $E$ can extract either $w_t$ or $SK$ within time $poly(n) \cdot 2^{n^{c_P}}$ in total. As $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_V}}$ and $f_V$ is secure against any $2^{N^{c_V}}$-time adversary, we know with overwhelming probability (except for a negligible fraction of $(PK, SK)$'s output by the key-generation stage of $V$) the extracted witness must be $w_t$. This means within time $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^{c_V}}$ $E$ will output all witnesses to common inputs of accepting sessions in $str$ with overwhelming probability.

As the simulated transcript $str$ is indistinguishable from the view of $P^*$ in real execution with honest verifier instances, this implies that for any $x$ and for all (except for a negligible fraction of) $(PK, SK)$ outputted by the key-generation stage of $V$, if $P^*$ can convince the honest verifier $V(SK)$ of "$x \in L$" in one of the $s(N)$ sessions with non-negligible probability $p_x$, then $P^*$ will also convince

$E(PK)$ of this statement with probability negligibly close to $p_x$. According to the knowledge-extraction ability of $E$, $E$ will output a witness to $x \in L$ with probability negligibly close to $p_x$. Then, the concurrent soundness of the protocol depicted in Figure 1 follows. This finishes the proof of Theorem 1.

### 3.1 Discussion: on the weak concurrent knowledge-extractability

We remark that the above proof for concurrent soundness actually establishes a (*black-box*) weak CKE property, roughly as follows: there exists a *sub-exponential-time* (specifically, $poly(n) \cdot 2^{n^{c_P}}$-time) *black-box* simulator/extractor $E$ such that for any concurrent malicious PPT prover $P^*$ against verifier instances with public-key $PK$, on the *same* public-key $PK$ $E$ outputs a simulated indistinguishable transcript, together with all witnesses to common inputs of accepting sessions in $str$. Note that, according to the parameter specifications in Section 3, $poly(n) \cdot 2^{n^{c_P}} \ll 2^{N^c} \ll 2^{N^{c_L}}$. Suppose the underlying language $L$ is $2^{N^{c_L}}$-hard for some constant $c_L$, $0 < c_L < 1$, such weak CKE property essentially says that $P^*$ "knows" witnesses to common inputs whose validations are successfully conveyed by concurrent interactions, rather than only convincing the verity (i.e., membership) of common inputs. Formal formulation of the weak CKE property and detailed discussions are deferred to the full version (in particular, the weak CKE property is strictly stronger than concurrent soundness in the public-key model under any sub-exponentially strong OWF).

We remark that super-polynomial-time is intrinsic to *black-box* knowledge-extraction for rZK arguments, as rZK (black-box) arguments of knowledge exist only for $\mathcal{BPP}$ languages [1]. Also, we believe that the weak CKE property is still very useful in practice. In particular, it allows highly practical rZK implementations for specific languages, e.g., DLP and RSA, that are widely assumed to be sub-exponentially hard.

## 4 Simplified, Practical, Round-Optimal Implementations

### 4.1 Simplified implementation

We further investigate the interactions combining Phase-1 and Phase-2 of the OWF-based rZK-CS protocol (depicted in Figure 1) when the messages $c_V^{(1)}$ and $a_V$ are removed from Stage-2 of Phase-1 (i.e., $V$ only sends $c_V^{(0)}$ at Stage-2 of Phase-1). The key observation here is that if the OWF $f_P$ used by the prover is *preimage-verifiable*, then such interactions can be replaced by only letting $P$ send, at the start, the initialization messages $(y_P, G_P, R_P)$: a *unique* value $y_P = f_P(x_P)$ (rather than a pair of values $(y_P^{(0)}, y_P^{(1)})$), the graph $G_P$ (reduced from $y_P$ by $\mathcal{NP}$-reduction) and the random string $R_P$. Note that the initialization messages $(y_P, G_P, R_P)$ is fixed once and for all. Thereby, we obtain a much more simplified 5-round implementation. In this case, the proof of Theorem 1 remains essentially unchanged (other than being simplified). *We remark that the preimage-verifiability property plays a critical role in the proof of concurrent* *soundness, as otherwise the malicious $P^*$ can distinguish whether it is interacting with honest verifier instances (who always continue the interactions w.r.t. a false $G_{P^*}$ in which no Hamiltonian cycle exists) or with the knowledge extractor (who always stops by brute-force checking the validity of $G_{P^*}$).*

### 4.2 Generic yet practical transformation

We first recall some key tools used in the generic practical transformation: We assume the OWF $f_V$ used in key-generation admits $\Sigma$-protocols. Note that the set of OWFs admitting $\Sigma$-protocols is large, which in particular includes the popular DLP and RSA functions [20, 14]. The PRF used by the prover is the Naor-Reingold PRFs that can be based on the factoring (Blum integers) or the decisional Diffie-Hellman hardness assumptions [18]. 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.

**Verifiable and $\Sigma$-provable trapdoor commitments (VPTC).** For our purpose, we need TC schemes satisfying the following additional requirements:

- 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. *Actually, in the generic practical transformation the public-key verifiability property just serves the role of preimage-verifiable OWF in the above preimage-verifiable OWF-based simplified implementation.*
- Public-key $\Sigma$-provability. On common input $TCPK$ and private input $TCSK$, one can prove, by $\Sigma$-protocols, the knowledge of $TCSK$.

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 [4] and the RSA-based [19] *perfectly-hiding* trapdoor commitment schemes are VPTC.

**The generic practical transformation from any $\Sigma$-protocol.** We highlight the modifications, in comparison with the preimage-verifiable OWF-based simplified implementation. The generic practical implementation is for any language $L$ that admits $\Sigma$-protocols. The RRF is replaced by Naor-Reingold PRF; The OWF $f_V$ used in key-generation stage is replaced by any OWF admitting $\Sigma$-protocols; The trapdoor commitment scheme is replaced by the VPTC scheme, and the sending of the $y_P$ using the preimage-verifiable OWF $f_P$ is just replaced by the sending of $VPTCPK$ on the top (note that we no longer need to reduce $VPTCPK$ to a Hamiltonian Graph by $\mathcal{NP}$-reductions); All WI protocols are replaced by $\Sigma_{OR}$-protocols (without $\mathcal{NP}$-reductions).

### 4.3 Round-optimal implementation

For the above 5-round preimage-verifiable OWF simplified implementation, to further reduce the round-complexity, we want to fold the prover's initialization message, i.e., $(y_P, G_P, R_P)$, into the third-round of the 5-round protocols (that is from the prover to the verifier). This would render us 4-round (that is optimal) rZK-CS arguments for $\mathcal{NP}$ in the BPK model. To this end, we let the verifier use OWP-based one-round perfectly-binding commitment scheme at Stage-2 of Phase-1 (thus waiving the value $R_P$), and replace the Blum's WIPOK protocol (executed on common input $(y_P, G_P, y_V)$ with the verifier playing the role of knowledge prover) by the Lapidot-Shamir WIPOK protocol (as in this case the verifier sends the first-round message without knowing the statement, i.e, $(y_P, G_P, y_V)$, to be proved). But, the challenge here is that, for our purpose, we need the following cryptographic tool (to replace the two-round FSTC scheme): A *one-round* OWP-based trapdoor commitment scheme based on DHC, in which the committer sends the one-round commitments without knowing the graph $G_P$ (serving as $TCPK$) 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. We develop a trapdoor commitment scheme of this type in this work, 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 $\pi$, 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 $\pi$). 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).

**Acknowledgments.** We thank Di Crescenzo, Persiano and Visconti for helpful discussions.

## 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. Blum. How to Prove a Theorem so No One Else can Claim It. In Proceedings of the International Congress of Mathematicians, pages 1444-1451, 1986.
4. Brassard, D. Chaum and C. Crepeau. Minimum Disclosure Proofs of Knowledge. *Journal of Computer Systems and Science*, 37(2): 156-189, 1988.
5. R. Canetti, O. Goldreich, S. Goldwasser and S. Micali. Resettable Zero-Knowledge. In *ACM Symposium on Theory of Computing*, pages 235-244, 2000.
6. 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.
7. 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, 1994.
8. I. Damgard. Lecture Notes on Cryptographic Protocol Theory, Aarhus University.
9. 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.
10. C. Dwork, M. Naor and A. Sahai. Concurrent Zero-Knowledge. In *ACM Symposium on Theory of Computing*, pages 409-418, 1998.
11. 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.
12. O. Goldreich. *Foundation of Cryptography-Basic Tools*. Cambridge University Press, 2001.
13. 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.
14. 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.
15. 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.
16. 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.
17. M. Naor. Bit Commitment Using Pseudorandomness. *Journal of Cryptology*, 4(2): 151-158, 1991.
18. M. Naor and O. Reingold. Number-Theoretic Constructions of Efficient Pseudo-Random Functions. *Journal of the ACM*, 1(2): 231-262 (2004).
19. 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.
20. C. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3): 24, 1991.
21. A. C. C. Yao, M. Yung and Y. Zhao. Concurrent Knowledge-Extraction in the Public-Key Model. Manuscript, 2007.
22. 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.