Electronic Colloquium on Computational Complexity, Report No. 162 (2005)

Generic yet Practical ZK Arguments from any Public-Coin HVZK

Yunlei Zhao [*] Jesper Buus Nielsen [†] Robert H. Deng [‡] Dengguo Feng [§]

Abstract

In this work, we present a generic yet practical transformation from any public-coin honest-verifier zero-knowledge (HVZK) protocols to normal zero-knowledge (ZK) arguments. By "generic", we mean that the transformation is applicable to any public-coin HVZK protocol under any one-way function (OWF) admitting Σ-protocols. By "practical" we mean that the transformation does not go through general $\mathcal{NP}$-reductions and only incurs additionally one round (for public-coin HVZK protocols of odd number of rounds that is the normal case in practice). In particular, if the starting public-coin HVZK protocols and the underlying Σ-protocols are practical, the transformed ZK arguments are also practical. In addition, our transformation also preserves statistical/perfect zero-knowledge. To this end, we develop generic yet practical 3-round perfectly-hiding *equivocal* (string) commitment scheme under any OWF admitting Σ-protocols, which is possibly of independent value. We also show that three rounds is the lower-bound of round-complexity for equivocal commitment schemes.

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 [25]. Since its introduction ZK has found numerous and extremely useful applications, and by now it has been playing a central role in modern cryptography.

ZK protocols for general languages (i.e., $\mathcal{NP}$) constitute an important plausibility result since many important statements are in $\mathcal{NP}$. But, they are normally hard to be directly employed in practice, particularly due to the underlying general $\mathcal{NP}$-reductions. In addition, ZK has many direct efficient applications (mainly employing number-theoretic statements). In particular, a very large number of protocols, named public-coin honest verifier zero-knowledge HVZK protocols, are developed directly for specific number-theoretic languages, which preserves the ZK property only with respect to *honest* verifiers (i.e., they are not normal ZK) but are highly practical. Thus, it's naturally desirable to develop a *generic yet practical* transformation from *any* public-coin HVZK protocols to (normal) ZK protocols.

1.1 Our contributions

In this work, we present a generic yet practical transformation from any public-coin honest-verifier zero-knowledge (HVZK) protocols to normal zero-knowledge (ZK) arguments. By "generic", we mean that the transformation is applicable to any public-coin HVZK protocol under any one-way function (OWF) admitting Σ-protocols. (*Note that the set of OWFs admitting Σ-protocols is large, which in particular includes both DLP and RSA.*) By "practical" we mean that the transformation does not go through general $\mathcal{NP}$-reductions and only incurs additionally one round (for public-coin HVZK protocols of odd number of rounds) or two rounds (for public-coin HVZK protocols of even number of rounds). Note that public-coin HVZK protocols of odd number of rounds is the normal case in practice. If the

[*]Software School, School of Information Science and Engineering, Fudan University, Shanghai 200433, China. ylzhao@fudan.edu.cn

[†]BRICS, University of Aarhus, Danmark. buus@daimi.au.dk

[‡]School of Information Systems, Singapore Management University, Singapore.

[§]State Key Laboratory of Information Security, Beijing, China.

ISSN 1433-8092

starting public-coin HVZK protocols and the underlying Σ-protocols are practical, the transformed ZK arguments are also practical.

To this end, we develop generic yet practical 3-round perfectly-hiding *equivocal* (string) commitment scheme under any OWF admitting Σ-protocols (by a novel use of Damgård's Σ-protocol-based commitment scheme [11] and Σ_{OR}-protocols introduced by Cramer, Damgård and Schoenmakers in [10]), which is possibly of independent value. The construction is conceptually simple, and its DLP or RSA based instantiations need only about 8 exponentiations by each participant for both commitment and decommitment. (We argue that conceptual simpleness of the construction could be a major advantage (rather than disadvantage) of our contributions.) We also show in this work that three rounds is the lower-bound of round-complexity for equivocal commitment schemes.

1.2 Related works and comparisons

Converting HVZK protocols into normal ZK protocols has attracted a series of extensive research efforts (and the idea of reducing the problem of security for arbitrary parties to the case of honest parties can be traced back to the works in secure distributed computing [38, 24]). Converting any (whether public-coin or not) HVZK protocols into normal ZK protocols was first studied by Bellare, Micali and Ostrovsky [2] under the DLP assumption, and the intractability assumption was further reduced to any one-way permutation by Ostrovsky, Venkatesan and Yung [36]. The BMO and OVY transformations preserve perfect/statistical ZK. The works of Damgård, Goldreich, Okamoto and Wigderson [13, 35] further achieved that, under any OWF, any (whether public-coin or not) HVZK protocols can be transformed into *public-coin* normal ZK protocols. The transformation of [13, 35] preserves statistical ZK, but not perfect ZK. Damgård first presented a transformation from any *constant-round* public-coin HVZK protocols to normal ZK protocols *without intractability assumptions* [12], by using the interactive hashing technique employed in [36]. The transformation of [12] preserves perfect/statistical ZK, and also proof/argument of knowledge. This (unconditional) approach was carried to its climax in [26] by showing that any public-coin HVZK *proofs* can be transformed into normal *public-coin* ZK proofs *without intractability assumptions*, by a deep investigation of the Random Selection technique employed in [13]. Combined with the result of [35] (i.e., any language that admits honest-verifier statistical ZK proofs also admits public-coin honest-verifier statistical ZK proofs), it established that the set of languages admitting honest-verifier statistical ZK *proofs* coincides with the set of languages admitting statistical ZK proofs in general. The transformation of [26] preserves perfect/statistical, but it is not provably applicable to HVZK arguments. These (earlier) works show important *general* methodology of achieving ZK protocols in general from HVZK protocols, and demonstrate elegant intersections between cryptography and complexity theory. But these transformations may still be not efficient enough for practice, partially due to the high (additional) round-complexity incurred. In more details, the BMO transformation [2] incurs $c \cdot m(n)$ additional rounds, where c is a constant and $m(n)$ denotes the round-complexity of the starting HVZK protocol on common inputs of length n (which implies the number of incurred rounds is non-constant for non-constant-round starting HVZK protocols); All other transformations presented in [36, 12, 13, 35, 26] incur at least *non-constant* additional rounds in getting normal ZK protocols of negligible soundness error (actually, they incur polynomially many additional rounds for achieving normal ZK protocols of exponentially vanishing soundness error). Other inefficiencies are due to the underlying general $\mathcal{NP}$-reductions (e.g., in the transformations of [2, 36]) and/or the (inefficient) general cryptographic primitives (e.g., OWF-based commitments, et al). In comparison, our transformation is a *practically implementable* one (under any OWF admitting Σ-protocols), transforming *any public-coin* HVZK protocols into normal ZK *arguments* with minimal additional rounds incurred and without going through general $\mathcal{NP}$-reductions. Our transformation preserves perfect/statistical ZK. In addition, our transformation also preserves argument of knowledge for Σ-protocols (that are a special form of public-coin HVZK). Note that for HVZK protocols used in practice, the most often case is the *public-coin* ones without $\mathcal{NP}$-reductions (mainly for number-theoretic languages).

Recently, Micciancio and Petrank presented a *practically implementable* transformation from any public-coin HVZK protocols to normal ZK protocols without general $\mathcal{NP}$-reductions *under the decisional*

Diffie-Hellman (DDH) assumption [31]. The Micciancio-Petrank transformation does *not* preserve perfect/statistical ZK. In comparison, our transformation converts any public-coin HVZK protocols into normal ZK *arguments under any OWF admitting* Σ*-protocol* (that includes, in particular, both DLP and RSA). Our transformation also preserves perfect/statistical ZK. For the additional round-complexity incurred, the Micciancio-Petrank transformation incurs three rounds (for public-coin HVZK protocols of odd number of rounds) or four rounds (for public-coin HVZK protocols of even number of rounds)[*]. In comparison, our transformation incurs only one round for public-coin HVZK protocols of odd number of rounds or two rounds for public-coin HVZK protocols of even number of rounds. For additional computational complexity incurred, the Micciancio-Petrank transformation (that is based on DDH) incurs additional 2 modular exponentiation operations for each round of the original public-coin HVZK protocol. The DLP (that is weaker than DDH) or RSA based implementations of our transformation incur the same additional computational complexity (i.e., 2 exponentiation operation for each round of the original protocol). As a critical tool, [31] achieved a DDH-based perfectly-binding simulatable commitment scheme. In this work, we achieve perfectly-hiding 3-round (that is optimal) equivocal commitment schemes under any OWF admitting Σ-protocols. We also note that the proof of soundness of the transformed ZK protocols by the Micciancio-Petrank transformation is trivial, guaranteed by the nicely explored DDH-based *perfectly-binding* (simulatable) commitment scheme (used by the prover for setting the random challenges of the starting public-coin HVZK protocols by a coin-tossing mechanism). But, the soundness proof of the ZK protocols by our generic yet practical transformation turns out to be complicated and subtle.

Cramer, Damgård and MacKenzie nicely showed how to achieve 4-round perfect ZK proofs of knowledge without general $\mathcal{NP}$-reductions and *without intractability assumptions* for any language L such that L admits Σ-protocols *and its associated commitment scheme* (via the Damgård paradigm for achieving commitment schemes from Σ-protocols [11]) *also admits* Σ*-protocols* [9]. 4-round perfect ZK arguments of knowledge without general $\mathcal{NP}$-reductions for any language admitting Σ-protocols also can be achieved by a Σ_{OR}-based implementation of the Feige-Shamir constant-round ZK arguments [21] (actually, the version appears in [20]). But, the soundness proofs in [9, 21] critically rely on the *special* soundness property of Σ-protocols, and thus the techniques of [9, 21] are probably not applicable for transforming any public-coin HVZK protocols to normal ZK protocols. Recall that any Σ-protocol is itself a 3-round public-coin *special* honest-verifier zero-knowledge SHVZK protocol with *special* soundness. Although Σ-protocols are useful tools, there are also many public-coin HVZK protocols (mainly for number-theoretic languages) that are not Σ-protocols. We also note that if the starting public-coin HVZK protocols are just Σ-protocols, the transformed protocols by our generic yet practical transformation are also 4-round perfect ZK arguments of knowledge (without general $\mathcal{NP}$-reductions) for any languages admitting Σ-protocols.

2 Preliminaries

We use standard notations and conventions below for writing probabilistic algorithms, experiments and interactive protocols. 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 S is a finite set we also denote by $|S|$ the size of S.) If α is neither an algorithm nor a set then $x \leftarrow \alpha$ is a simple assignment statement. By $[R_1; \cdots; R_n : v]$ we denote the set of values of v that a random variable can assume, due to the distribution determined by the sequence of random processes $R_1, R_2, \cdots, R_n$. By $\Pr[R_1; \cdots; R_n : E]$ we denote the probability of event E, after the ordered execution of random processes $R_1, \cdots, R_n$.

Let $\langle P, V \rangle$ be a probabilistic interactive protocol, then the notation $(y_1, y_2) \leftarrow \langle P(x_1), V(x_2)\rangle(x)$ denotes the random process of running interactive protocol $\langle P, V \rangle$ on common input x, where P has private input x_1, V has private input x_2, y_1 is P's output and y_2 is V's output. We assume wlog that the

[*]In [31], the authors only analyzed the additional round complexity incurred for HVZK of odd number of rounds.

output of both parties P and V at the end of an execution of the protocol $\langle P, V\rangle$ contains a transcript of the communication exchanged between P and V during such execution.

Definition 2.1 (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 system is called a public-coin system if at each round the prescribed verifier only tosses a predetermined number of coins and send the outcome (random challenge) to the prover.

Definition 2.2 (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 $view_{V^*(z)}^{P(w)}(x)$ a random variable describing the contents of the random tape of V^* and the messages V^* receives from P during 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 $W^1 = \{w_x^1\}_{x\in L}$ and $W^2 = \{w_x^2\}_{x\in L}$ for sufficiently long x, so that $(x,\, w_x^1) \in R_L$ and $(x,\, w_x^2) \in R_L$, the following two ensembles are computationally indistinguishable by any non-uniform PPT algorithm: $\{x,\, view_{V^*(z)}^{P(w_x^1)}(x)\}_{x\in L,\, z\in\{0,1\}^*}$ and $\{x,\, view_{V^*(z)}^{P(w_x^2)}(x)\}_{x\in L,\, z\in\{0,1\}^*}$. 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*

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

Definition 2.3 (zero-knowledge ZK) *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 $view_{V^*(z)}^{P(w)}(x)$ a random variable describing the contents of the random tape of V^* and the messages V^* receives from P during an execution of the protocol on common input x, when P has auxiliary input w and V^* has auxiliary input z. Then we say that $\langle P,\, V\rangle$ is zero-knowledge if for every probabilistic polynomial-time interactive machine V^* there exists a probabilistic (expected) polynomial-time oracle machine S, such that for all sufficiently long $x \in L$ the ensembles $\{view_{V^*}^{P(w)}(x)\}_{x\in L}$ and $\{S^{V^*}(x)\}_{x\in L}$ are computationally indistinguishable. Machine S is called a ZK simulator for $\langle P,\, V\rangle$. The protocol is called statistical ZK if the above two ensembles are statistically close (i.e., the variation distance is eventually smaller than $\frac{1}{p(|x|)}$ for any positive polynomial p). The protocol is called perfect ZK if the above two ensembles are actually identical (i.e., except for exponentially negligible probabilities, the two ensembles are equal).*

Definition 2.4 (public-coin HVZK) *Let $\langle P,\, V\rangle$ be a public-coin interactive protocol for a language L in which the prescribed honest verifier V is supposed to send $t-1$, $t \geqslant 2$, random challenges. Denote by α_i, $1 \le i \le t$, the i-th message of honest prover and by c_i, $1 \le i \le t-1$, the i-th random challenge of the honest verifier. The honest prover P is a PPT interactive machine that on the common input x, an auxiliary input w and a partial transcript T where T is either an empty string or a sequence of messages $(\alpha_1, c_1, \cdots, c_i)$, $1 \le i \le t-1$, outputs the next message α_{i+1}. We denote by $view_V^{P(w)}(x)$ a random variable describing the contents of the random tape of (the honest) V and the messages V receives from P during an execution of the protocol on common input x when P has auxiliary input w. Such a public-coin protocol is called honest verifier zero-knowledge (HVZK) if there exists a probabilistic polynomial time simulator S such that for all sufficiently long $x \in L$ the following ensembles are computationally indistinguishable: $\{S(x)\}_{x\in L}$ and $\{view_V^{P(w)}(x)\}_{x\in L}$.*

In above definition, we have assumed that the prover initiates the protocol. It also can be easily extended to deal with the case that the verifier initiates the protocol.

Definition 2.5 (Σ-protocol [8]) *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 k (that is polynomially related to n) selected uniformly at random in $\{0,1\}^k$.*
- *Perfect SHVZK (Special honest verifier zero-knowledge). There exists a probabilistic polynomial-time (PPT) simulator S, which on input x (where there exists a w such that $(x, w) \in R$) and a random challenge string $\hat{e}$, outputs an accepting conversation of the form $(\hat{a}, \hat{e}, \hat{z})$, with the same probability distribution as the real conversation (a, e, z) between the honest $P(w)$, V on input x.*

Σ-Protocol for DLP [37]. The following is a Σ-protocol $\langle P, V\rangle$ proposed by Schnorr [37] 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 $\mathbf{Z}_{2^k}$ and sends it to P. Here, k is fixed such that $2^k < q$.
- P sends $z = r + ew \ mod\ q$ to V, who checks that $g^z = ah^e \ mod\ p$, that p, q are prime and that g, h have order q, and accepts iff this is the case.

Σ-Protocol for RSA [28]. 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 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^k} and sends it to P. Here, k is fixed such that $2^k < q$.
- 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 [10]. 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 k, 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 k-bit string s and sends it to P.
- P sets $e_b = s \oplus e_{1-b}$ and computes the answer z_b to challenge e_b using (x_b, a_b, e_b, w) as input. He sends (e_0, z_0, e_1, z_1) to V.

- V checks that $s = e_0 \oplus e_1$ 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 2.1 [10] *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.*

The perfect SHVZK simulator of Σ_{OR} [10]. For a Σ_{OR}-protocol of above form, denote by S_{OR} the perfect SHVZK simulator of it and denote by S_b the perfect SHVZK simulator of the protocol $\langle P_b, V_b \rangle$ for $b \in \{0, 1\}$. Then on common input (x_0, x_1) and a random string $\hat{e}$ of length k, $S_{OR}((x_0, x_1), \hat{e})$ works as follows: It firstly chooses a random k-bit string $\hat{e}_0$, computes $\hat{e}_1 = \hat{e} \oplus \hat{e}_0$, then S_{OR} runs $S_b(x_b, \hat{e}_b)$ to get a simulated transcript $(\hat{a}_b, \hat{e}_b, \hat{z}_b)$ for $b \in \{0, 1\}$, finally S_{OR} outputs $((\hat{a}_0, \hat{a}_1), \hat{e}, (\hat{e}_0, \hat{z}_0, \hat{e}_1, \hat{z}_1))$.

Definition 2.6 (trapdoor (string) 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. $\forall n$, $\forall v$ of length k (where $k = k(n)$ for some polynomial $k(\cdot)$),*
  $\Pr[(TCPK, TCSK) \xleftarrow{R} \text{TCGen}(1^n); (c, d) \xleftarrow{R} \text{TCCom}(1^n, 1^k, TCPK, v) :$
  $\text{TCKeyVer}(1^n, TCPK) = \text{TCVer}(1^n, 1^k, TCPK, c, v, d) = 1] = 1$.
- *Computational Binding. For all sufficiently large n's and for any PPT adversary A, the following probability is negligible in n (where $k = k(n)$ for some polynomial $k(\cdot)$):*
  $\Pr[(TCPK, TCSK) \xleftarrow{R} \text{TCGen}(1^n); (c, v_1, v_2, d_1, d_2) \xleftarrow{R} \text{A}(1^n, 1^k, TCPK) :$
  $\text{TCVer}(1^n, 1^k, TCPK, c, v_1, d_1) = \text{TCVer}(1^n, 1^k, TCPK, c, v_2, d_2) = 1 \bigwedge |v_1| = |v_2| = k \bigwedge v_1 \neq v_2]$.
- *Perfect (or Computational) Hiding. $\forall$ $TCPK$ such that* $\text{TCKeyVer}(TCPK, 1^n) = 1$ *and $\forall$ v_1, v_2 of equal length k, the following two probability distributions are identical (or computationally indistinguishable):*
  $[(c_1, d_1) \xleftarrow{R} \text{TCCom}(1^n, 1^k, TCPK, v_1) : c_1]$ *and* $[(c_2, d_2) \xleftarrow{R} \text{TCCom}(1^n, 1^k, TCPK, v_2) : c_2]$.
- *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 k, the following two probability distributions are identical (or computationally indistinguishable):*
  $[(c_1, d_1) \xleftarrow{R} \text{TCCom}(1^n, 1^k, TCPK, v_1); d_2' \xleftarrow{R} \text{TCFake}(1^n, 1^k, TCPK, TCSK, c_1, v_1, d_1, v_2) : (c_1, d_2')]$
  and $[(c_2, d_2) \xleftarrow{R} \text{TCCom}(1^n, 1^k, TCPK, v_2) : (c_2, d_2)]$.

Damgård's paradigm for achieving perfectly-hiding trapdoor commitment schemes from Σ-protocols [11]. Recall that by the *perfect* SHVZK property, the distribution of $(\hat{a}, \hat{e}, \hat{z})$ outputted by the SHVZK simulator $S(x, \hat{e})$ on a random string $\hat{e}$ of length k is *identical* to that of real interaction transcript (a, e, z) between the honest P, V on x. But, in the real conversation (a, e, z), a is independent of e and thus reveals no information of e. This means that in the simulated transcript $(\hat{a}, \hat{e}, \hat{z})$, $\hat{a}$ reveals also no information of $\hat{e}$ (perfect hiding). Furthermore, given $\hat{a}$, the simulator S also cannot answer a different challenge $\hat{e}' \neq \hat{e}$ because of the special soundness of Σ-protocols (this is just the binding property of commitments). The important observation here is that even if $\hat{e}$ is an *arbitrary* string of length k (rather than a random string), $\hat{a}$ still perfectly hides $\hat{e}$. This is so because of the *perfect* SHVZK property of Σ-protocols. This directly brings us the following perfectly-hiding trapdoor commitment scheme $\langle P, V \rangle$.

Round-1. On a security parameter n, let f be a OWF that admits Σ-protocols (with random challenges of length k that is polynomially related to n). Then, the commitment receiver V randomly selects an element x of length n in the domain of f, computes $f(x)$, and sends $f(x)$ to the commitment sender P.

Round-2. To commit a message m of length k, P runs the SHVZK simulator $S(f(x), m)$ to get a simulated transcript, denoted $(\hat{a}, m, \hat{z})$, pretending that it "knows" the preimage of $f(x)$. P sends $\hat{a}$ to V as the commitment.

Decommitment Stage. P reveals $(m, \hat{z})$ and V accepts if $(\hat{a}, m, \hat{z})$ is an accepting conversation on $f(x)$.

The perfectly-hiding and computationally-binding properties of the above scheme can be easily checked according to the above arguments. The trapdoorness property is from the observation that if one knows the preimage of the (*well-formed*) $f(x)$ then it can equivocate $\hat{a}$ at its wish.

3 Generic yet Practical Equivocal Commitments with Σ_{OR}-Protocols

In this section, we present *generic yet practical* 3-round perfectly-hiding equivocal (string) commitment schemes under any OWF admitting Σ-protocols. By "generic" we mean our equivocal commitments can be implemented under any one-way function (OWF) that admits Σ-protocols (*note that the set of OWFs admitting Σ-protocols is large, which in particular includes both DLP and RSA*). By "practical" we mean our equivocal commitment schemes do not go through general $\mathcal{NP}$-reductions and if the underlying Σ-protocols are practical then the transformed equivocal commitment protocols are also practical. With DLP or RSA as examples, the DLP-based or RSA-based instantiations need only 8 exponentiations by each participant for both commitment and decommitment. We also clarify in this section the relationship between trapdoor commitments and equivocal commitments, and show that three rounds is the lower-bound of round-complexity for equivocal commitment schemes.

Informally speaking, a commitment scheme is equivocal if it satisfies the following additional requirement. There exists an efficient algorithm, called the simulator, which outputs a transcript leading to a "fake" commitment such that: (1) the "fake" commitment can be decommitted to both 0 and 1; and (2) the distribution of the simulated transcript is indistinguishable from that of the real view of an even malicious commitment receiver in a real execution of the protocol. Equivocal commitments could be viewed as a stronger notion than trapdoor commitments and are widely used as a key ingredient for achieving various advanced cryptographic protocols, e.g., zero-knowledge [33, 18, 19], secure multi-party computation [7, 29], and more advanced commitment schemes (like non-malleable commitments [16, 22, 17, 14], simulation-sound commitments [30] and universally composable commitments [6, 15, 7]), et al.

Now, we present the formal definition of (black-box) equivocal (string) commitments that is the black-box and full version of the definition presented in [18].

Definition 3.1 *Let $\langle P, V\rangle$ be an interactive protocol. We say that $\langle P, V\rangle$ is a (black-box)* **equivocal** *commitment scheme if it satisfies the following:*

Perfect (or computational) hiding. *For all sufficiently large n's, any PPT adversary V^* and any s, s' of equal length k (where $k = k(n)$ for some polynomial $k(\cdot)$), the following two probability distributions are identical (or computationally indistinguishable):* $[(\alpha, \beta) \leftarrow \langle P(s), V^*\rangle(1^n, 1^k) : \beta]$ *and* $[(\alpha', \beta') \leftarrow \langle P(s'), V^*\rangle(1^n, 1^k) : \beta']$.

Computational Binding. *For all sufficiently large n's, and any PPT adversary P^*, the following probability is negligible in n:* $\Pr[(\alpha, \beta) \leftarrow \langle P^*, V\rangle(1^n, 1^k); (t, (t, v)) \leftarrow \langle P^*(\alpha), V(\beta)\rangle(1^n, 1^k);$ $(t', (t', v')) \leftarrow \langle P^*(\alpha), V(\beta)\rangle(1^n, 1^k) : |v| = |v'| = k \bigwedge v \neq v']$.

That is, no PPT adversary P^* can decommit the same transcript of the commitment stage to two different values with non-negligible probabilities.

Simulation indistinguishability and equivocability. *There exists a probabilistic polynomial-time S (who works in two stages: the commitment stage S^c and the decommitment stage S^d) such that for any probabilistic polynomial-time algorithm V^*, it holds:*

1. *Indistinguishability. For any string $s \in \{0,1\}^k$, the distributions $T(S)$ and $T(P)$ are computationally indistinguishable, where*

$$T(S) = [(\alpha, \beta) \leftarrow S^c_{V^*}(1^n, 1^k); (t, (t, s)) \leftarrow S^d_{V^*(\beta)}(\alpha, s, 1^n) : (\beta, (t, s))]$$

$$T(P) = [(\alpha, \beta) \leftarrow \langle P(s), V^* \rangle (1^n, 1^k); (t, (t, s)) \leftarrow \langle P(\alpha, s), V^*(\beta) \rangle (1^n, 1^k) : (\beta, (t, s))]$$

.

   That is, for any string s of length k, S outputs a simulated transcript (of both the commitment stage and the decommitment stage) that is indistinguishable from the view of V^* in real interactions with P when P commits s in the commitment stage and then reveals s in the decommitment stage, where the simulation of the commitment stage (i.e, S^c) is independent of the string s that is only given to S^d after the simulation of S^c is finished.

2. *Equivocability. For all constant c, all sufficiently large n, any string $s \in \{0,1\}^k$, where $k = k(n)$ for some polynomial $k(\cdot)$, it holds that $|p_0 - p_1| \leq n^{-c}$, where p_0, p_1 are, respectively,*

$$\Pr[(\alpha, \beta) \leftarrow S^c_{V^*}(1^n, 1^k); (t, (t, v)) \leftarrow S^d_{V^*(\beta)}(1^n, 1^k, \alpha, s) : v = s],$$

$$\Pr[(\alpha, \beta) \leftarrow \langle P(s), V^* \rangle (1^n, 1^k); (t, (t, v)) \leftarrow \langle P(\alpha, s), V^*(\beta) \rangle (1^n, 1^k) : v = s].$$

   That is, S can decommit the simulated transcript of the commitment stage to any *value (of length k) correctly.*

Trapdoor commitments versus equivocal commitments. We first remark that the notions of trapdoor commitments and equivocal commitments are significantly different in nature. In particular, the trapdoorness property of a trapdoor commitment scheme is defined with respect to *honest* commitment receiver. In other words, the trapdoorness property of a trapdoor commitment scheme is defined with respect to *well-formed $TCPK$*. For the Damgård's Σ-protocol based trapdoor commitment scheme (described in Section 2), the $TCPK$ is $y = f(x)$. Then, the trapdoorness property says that for well-formed $y = f(x)$ (which guarantees the existence of the preimage x), if one knows the preimage x then it can equivocate commitments at its wish. But, it does not guarantee that the trapdoorness property still holds if y is maliciously formed. For example, a malicious commitment receiver may send a maliciously formed y' such that there exists no preimages of y', while it is hard for the honest commitment sender to verify whether or not the preimages of y' exist. For example, consider the SQUARE one-way permutation (over the quadratic residues): $f(x) = x^2 \mod N$, where $N = p \cdot q$ and $p = q = 3 \mod 4$. Then the malicious commitment receiver may form y' to be a non-square such that there exists no x satisfying $y' = x^2 \mod N$. Note that the honest commitment sender (that is a PPT algorithm) cannot efficiently verify whether y' is a square or not, and thus the trapdoorness property becomes meaningless in this case. But, for equivocal commitments, the equvocability is defined with respect to *any malicious commitment receiver*, whether honest or dishonest. This is a difference in nature between trapdoor commitments and equivocal commitments.

Now, we present the generic yet practical 3-round equivocal (string) commitment scheme under any OWF admitting Σ-protocols, which is depicted in Figure 1 (page 9).

The commitment stage of the protocol depicted in Figure 1 runs in 4 rounds, but it can be reduced into 3 rounds by merging Phase-2 into the second round of Phase-1. As we shall see, 3 rounds is the optimal round complexity for (black-box) equivocal commitments in the standard model.

Theorem 3.1 *Suppose f_V be* any *one-way function that admits Σ-protocols, the protocol depicted in Figure 1 is a 3-round perfectly-hiding equivocal (string) commitment scheme in the standard model.*

The Σ_{OR}-based equivocal commitment scheme $\langle P, V \rangle$

Commitment Stage. The commitment stage consists of the following two phases:

Phase-1. On a security parameter n, the commitment receiver V selects a OWF f_V that admits Σ-protocols, randomly selects two elements in the domain of f_V, x_V^0 and x_V^1 of length n, and computes $y_V^0 = f_V(x_V^0)$ and $y_V^1 = f_V(x_V^1)$. Finally, V sends (y_V^0, y_V^1) to the commitment sender P, and proves to P that it knows the preimage of either y_V^0 or y_V^1 by executing the Σ_{OR}-protocol on (y_V^0, y_V^1) and playing the role of the knowledge prover. The witness used by V during the execution of the Σ_{OR}-protocol is x_V^b for a randomly chosen bit b in $\{0,1\}$.

Phase-2. P firstly checks the validity of the Σ_{OR}-protocol and aborts if it is not valid. Otherwise, suppose $m \in \{0,1\}^k$ be the message to be committed, P runs the perfect SHVZK simulator $S_{OR}((y_V^0, y_V^1), m)$ (as described in Section 2) to get a simulated transcript, denoted $((\hat{a}_0, \hat{a}_1), m, (\hat{e}_0, \hat{z}_0, \hat{e}_1, \hat{z}_1))$, pretending that it "knows" either the preimage of y_V^0 or the preimage of y_V^1. Finally, P sends $(\hat{a}_0, \hat{a}_1)$ to V while keeping $(m, (\hat{e}_0, \hat{z}_0, \hat{e}_1, \hat{z}_1))$ in secret as the decommitment information.

Decommitment Stage. P reveals $(m, (\hat{e}_0, \hat{z}_0, \hat{e}_1, \hat{z}_1))$. V checks that $m = \hat{e}_0 \oplus \hat{e}_1$ and that for both $b = 0$ and $b = 1$ $(\hat{a}_b, \hat{e}_b, \hat{z}_b)$ is an accepting conversation with respect to y_V^b. V accepts if all the above are valid, otherwise it rejects.

Figure 1. Generic yet practical round-optimal equivocal string commitments under any OWF admitting Σ-protocols

Proof (of Theorem 3.1).

Perfect hiding.

The perfectly-hiding property of the protocol can be easily checked by noting that Phase-1 is independent of the message m to be committed and Phase-2 also perfectly hides m due to the *perfect* SHVZK property of Σ-protocols as discussed in Section 2.

Computational binding.

Note that the binding property of the protocol relies on the secrecy of the preimages of (y_V^0, y_V^1) and V does prove to P such knowledge in Phase-1. But the Σ_{OR}-protocol used in Phase-1 only guarantees WI property which is a much weaker security notion than zero-knowledge. And thus, one may argue that by interacting with V, an adversary P^* could potentially gain some knowledge about the preimages and the gained knowledge could help it violate the binding property. What save us here are the key pair trick (which is originally introduced by Naor and Yung in the PKE setting [34] and is also employed in the ZK setting [20]) and the (perfect) WI property of Σ_{OR}-protocols.

Specifically, suppose the protocol does not satisfy the binding property, then there exists a PPT adversary P^* such that with non-negligible probability $q(n)$ P^* can decommit the transcript of the commitment stage to two different values. Then we can construct a PPT algorithm E that breaks the one-wayness of f_V with non-negligible probability. On an input y in the range of f_V, E works as follows: It randomly selects an element x' of length n in the domain of f_V, computes $y' = f_V(x')$, randomly selects a bit b in $\{0,1\}$, sets $y^b = y$ and $y^{1-b} = y'$. Finally, E sends (y^0, y^1) to P^*, and proves to P^* that it knows either the preimage of y^0 or the preimage of y^1 by executing the Σ_{OR}-protocol with x' as its witness. Now, suppose P^* can decommit the transcript of the commitment stage to two different values with non-negligible probability $q(n)$, then according to the special soundness of Σ-protocols it means that P^* can output a preimage of y^b or y^{1-b} with probability $q(n)$. But because the perfect WI property of Σ_{OR}-protocols, we know with probability $\frac{1}{2}q(n)$ E will get the preimage of $y^b = y$, which violates the one-wayness of f_V.

Equivocal commitments.

We directly deal with the more complicated 3-round case when Phase-2 is combined into the second round of Phase-1. For any PPT adversary V^*, the PPT simulator S is depicted in Figure 2 (page 10).

The equivocability simulator $S = (S^c, S^d)$

The simulation of S^c: S^c runs V^* as a subroutine and goes through the following (at most) two stages:

Stage-1. Denote by a, e, z the first, the second and the third message of the Σ_{OR}-protocol in Phase-1 of the commitment protocol (depicted in Figure 1) with respect to a pair $(y^0_{V^*}, y^1_{V^*})$. S^c runs V^* as a subroutine and in the second round (of the *combined* 3-round protocol) it does the following: S^c runs $S_{OR}((y^0_{V^*}, y^1_{V^*}), 0^k)$ to get a simulated transcript, denoted $((\hat{a}_0, \hat{a}_1), 0^k, (\hat{e}_0, \hat{z}_0, \hat{e}_1, \hat{z}_1))$ (that is, S^c tries to commit 0^k), randomly selects a random string e in $\{0,1\}^k$ and send $(e, (\hat{a}_0, \hat{a}_1))$ to V^*. Whenever V^* aborts with an incomplete execution of Phase-1 or (a, e, z) is not an accepting conversation with respect to $(y^0_{V^*}, y^1_{V^*})$, then S^c stops and outputs the transcript up to now. Otherwise, it goes to Stage-2.

Stage-2. Whenever S^c receives z from V^* in Stage-1 and (a, e, z) is an accepting conversation with respect to $(y^0_{V^*}, y^1_{V^*})$, S^c does the following:

Repeat Loop: Repeat up to $k \cdot 2^k$ times

- S^c rewinds V^* to the point that V^* just sent a, runs $S_{OR}((y^0_{V^*}, y^1_{V^*}), 0^k)$ (with fresh randomness) to get a simulated transcript, denoted $((\hat{a}'_0, \hat{a}'_1), 0^k, (\hat{e}'_0, \hat{z}'_0, \hat{e}'_1, \hat{z}'_1))$, randomly selects e' from $\{0,1\}^k/e$ and returns back $(e', (\hat{a}'_0, \hat{a}'_1)))$ to V^*.
- S^c runs V^* further and if it receives back a z' such that (a, e', z') is an accepting conversation with respect to $(y^0_{V^*}, y^1_{V^*})$ (this means E gets the preimage of either $y^0_{V^*}$ or $y^1_{V^*}$) then S^c outputs $((y^0_{V^*}, y^1_{V^*}), a, e', z', (\hat{a}'_0, \hat{a}'_1))$ and stop. Otherwise, goto **Repeat Loop**.

Failure with Error Message. In case all the $k \cdot 2^k$ attempts have failed, S^c aborts with an error message.

The simulation of S^d: Denote by α the output of S^c. In case S^c did not stop in Stage-1 and did not abort with an error message in Stage-2, then for any string s of length k S^d does the following: it generates the decommitment information, denoted by $(\hat{e}''_0, \hat{z}''_0, \hat{e}''_1, \hat{z}''_1)$, by using the extracted preimage of either $y^0_{V^*}$ or $y^1_{V^*}$ as its trapdoor such that $((\hat{a}'_0, \hat{a}'_1), s, (\hat{e}''_0, \hat{z}''_0, \hat{e}''_1, \hat{z}''_1))$ constitute a successful conversation for showing the knowledge of the preimage of either $y^0_{V^*}$ or $y^1_{V^*}$.

Figure 2. The equivocability simulator

Lemma 3.1 *S^c works in expected polynomial-time and the probability that S^c aborts with an error message is negligible.*

Proof (of Lemma 3.1). We charge each execution of Phase-1 of the protocol (depicted in Figure 1) as unit cost and our aim is to show that the expected charge accumulated in the **Repeat Loop** of Stage-2 is $poly(k)$.

Denote by p the probability that V^* successfully finishes Stage-1 in the simulation. As S^c goes into Stage-2 only if $p > 0$, below we distinguish two cases: $0 < p < 2^{-(k-1)}$ and $p \geq 2^{-(k-1)}$. When $0 < p < 2^{-(k-1)}$, the repeat loop is iterated at most $k \cdot 2^k < \frac{2k}{p}$ times. When $p \geq 2^{-(k-1)}$, each iteration of the repeat loop extracts a preimage of $(y^0_{V^*}, y^1_{V^*})$ with probability at least $p - 2^{-k}$. Thus, the expected number of iterations of the repeat loop in this case is less than $(p - 2^{-k})^{-1} \leq \frac{2}{p}$. Furthermore, as we shall see, with probability at least $1 - 2^{-k}$, the repeat loop is not repeated more than $\frac{2k}{p}$ times.

S^c may abort with an error messages only in two cases. The first case is when $p \geq 2^{-(k-1)}$, but in this case abort happens with probability at most $(1 - \frac{p}{2})^{k \cdot 2^k} < 2^{-k}$, since $k \cdot 2^k \geq \frac{2k}{p}$. The second case is when $0 < p < 2^{-(k-1)}$, but in this case S^c goes into Stage-2 only with probability $p < 2^{-(k-1)}$. We conclude that S^c aborts with an error message with probability at most $2^{-(k-1)}$. □

There are two differences between the simulated transcript outputted by S and the real interaction transcript between V^* and the honest P who is committing to a message s of length k. One is that in real interactions, P commits s and then reveals s accordingly, but in the simulation S always commits 0^k in the commitment stage and then reveals s in the decommitment stage; The second is that in the simulation S may abort with an error message. But, the first difference makes no distinction due to the perfectly-hiding property of the protocol (i.e., the perfect SHVZK property of Σ-protocols), and the second difference occurs only with negligible probability and thus also cannot make distinguishable distinction.

For the equivocability property, according to Definition 3.1, $p_1 = p$, and $0 \leq p_0 \leq p$ when $p < 2^{-(k-1)}$ or $(1 - 2^{-k}) \cdot p \leq p_0 \leq p$ when $p > 2^{-(k-1)}$. We conclude $|p_0 - p_1|$ is negligible. □

Finally, we show that the round-complexity (i.e., three rounds) of our generic yet practical equivocal commitment scheme is optimal. Specifically, we show that assuming $\mathcal{NP} \not\subseteq \mathcal{BPP}$, there is no 2-round (black-box) equivocal commitment scheme *of non-interactive decommitment stage* (that is the normal case of commitment schemes).

Theorem 3.2 *Assuming $\mathcal{NP} \not\subseteq \mathcal{BPP}$, there is no 2-round (black-box) equivocal commitment scheme (of non-interactive decommitment stage).*

Proof (of Theorem 3.2). We prove this theorem separately according to whether the prover or the verifier sends the first-round message. Specifically, we prove the following two propositions:

Proposition 3.1 *There is no 2-round (black-box) equivocal commitment scheme in which the verifier sends the first-round message.* Note that this proposition holds unconditionally.

Proof (of Proposition 3.1). For any 2-round commitment scheme, denote by α (sent by the verifier) the first-round message and by β (sent by the prover) the second-round message. By the binding property, there is no PPT algorithm that *given* (a correctly generated) α can generate a valid β such that it can decommit (α, β) to two different values. This means that even if the commitment receiver always uses the same α in all its interactions (i.e., the commitment receiver correctly generates α and then fixes it once and for all) and even just publishes α as its public-key, still no PPT algorithm can violate the binding property with respect to the same α. Now, suppose the 2-round scheme is a equivocal commitment scheme, then according to the definition of equivocal commitments there is a PPT simulator that given the *fixed* α can generate a valid β such that it can decommit (α, β) to two different values. This violates the binding property of commitments. □

Proposition 3.2 *Assuming $\mathcal{NP} \not\subseteq \mathcal{BPP}$, there is no 2-round (black-box) equivocal commitment scheme in which the prover sends the first-round message.*

Proof (of Proposition 3.2). Suppose there exists a 2-round black-box equivocal commitment scheme in which the prover sends the first-round message, then we will construct a 3-round black-box ZK argument for $\mathcal{NP}$, which contradicts the lower-bound of [23] (i.e., only languages in $\mathcal{BPP}$ have 3-round black-box ZK arguments).

Specifically, consider the Blum's protocol for the $\mathcal{NP}$-Complete language Directed Hamiltonian Cycle DHC (that is depicted in Appendix A). We replace the underlying perfectly-binding commitment scheme used in the first-round of Blum's protocol by the assumed 2-round prover-initiated black-box equivocal commitment scheme. The resultant protocol still runs in three rounds. The black-box ZK property of the resultant protocol is direct from the simulation indistinguishability and equivocability property of the assumed 2-round black-box equivocal commitment scheme. The computational soundness of the resultant protocol is from the computational binding property of the assumed 2-round equivocal commitment scheme. Specifically, the ability to answer two different challenges (sent in the second-round) with respect to the same equivocal commitments (executed in the first-round and a part of the second-round) implies breaking the computational binding property of the assumed 2-round equivocal commitment scheme. □ □

Comments and Comparisons. By a novel use of Damgård's Σ-protocol-based commitment scheme [11] and Σ_{OR}-protocols introduced by Cramer, Damgård and Schoenmakers in [10], our construction for achieving generic yet practical round-optimal equivocal commitments are actually of conceptual simpleness. *We argue that conceptual simpleness of the construction could be one major advantage (rather than disadvantage) of our contributions.*

The notion of equivocal commitments is proposed in [1], where the construction of such a scheme was left over as an open problem. Equivocal commitments are firstly achieved in [16] by a modification of Naor's scheme [32] in the common reference string model. Equivocal commitments in the standard model without setup assumptions are firstly achieved in [18]. The approach of [18] is to nicely modify the schemes of [32, 4] with a coin-tossing on the top, and thus the resultant equivocal commitment schemes run in at least four rounds. Very recently, Katz and Ostrovsky nicely present a zero-knowledge based solution for equivocal commitments [29], which can be easily modified to run in three rounds. The Katz-Ostrovsky scheme is a bit commitment scheme and to commit a string the scheme will be repeated in parallel, and thus may not be efficient for committing strings. Note that our scheme is equivocal string commitment scheme, and can be practically instantiated with a very small constant number of exponentiations. Also, although 3-round equivocal commitment scheme is (implicitly) known in [29], a proof of the lower-bound of round-complexity (as demonstrated in Theorem 3.2) is unknown previously, to the best of our knowledge.

4 Generic yet Practical Transformation from any Public-Coin HVZK Protocol to ZK Argument

The high-level overview of the transformation. Given a public-coin HVZK protocol $\langle P_L, V_L \rangle$ for a language L. Let $2t-1$ ($t \geq 2$) be the number of rounds of the protocol. Without loss of generality we assume all random challenges of the honest verifier V_L are of the same length. More precisely, the prover P_L is a PPT interactive machine that on input a string $x \in L$ of length n, an auxiliary input w, and a partial transcript T where T is either an empty string or a sequence of messages $(\alpha_1, c_1, \cdots, c_i)$, $1 \leq i \leq t-1$, outputs the next message α_{i+1}. The verifier algorithm V_L answers each prover message α_i with a challenge c_i of length k (that is polynomially related to n) taken uniformly at random from $\{0,1\}^k$, and at the end of interactions applies a verification procedure $V_L(x, \alpha_1, c_1, \cdots, \alpha_{t-1}, c_{t-1}, \alpha_t)$ to determine whether to accept or reject x. We also denote by S_L the HVZK simulator of $\langle P_L, V_L \rangle$.

To transform $\langle P_L, V_L \rangle$ into a normal ZK protocol $\langle P, V \rangle$ for the same language L, we design a coin-tossing mechanism to set the random challenges of $\langle P_L, V_L \rangle$ jointly by the prover and the (possibly malicious) verifier, by employing the generic yet practical equivocal commitment scheme developed in Section 3. The observation here is that all these $t-1$ equivocal commitments can share the same Phase-1 of the commitment stage (as described in Figure-1), and thus the incurred additional round-complexity could be minimal. The transformed protocol $\langle P, V \rangle$ is depicted in Figure 3 (page 13).

Comments on the incurred round-complexity: It's easy to check that, in comparison with the starting public-coin protocol $\langle P_L, V_L \rangle$, the number of the incurred additional rounds is that of Phase-1. Furthermore, the second-round and the third-round of Phase-1 can be merged into the first two rounds of Phase-2. Thus, the transformation only incurs additionally one round. (For public-coin HVZK protocols of even number of rounds, the transformation incurs additionally two rounds.)

Theorem 4.1 *Assuming f_V be any OWF admitting Σ-protocols, the transformed protocol $\langle P, V \rangle$ (depicted in Figure 3) is a ZK argument for (the same) language L.*

Proof (of Theorem 4.1).

Zero-knowledge.

The strategy of the ZK simulator S is as follows: it first runs the HVZK simulator $S_L(x)$ (of the starting public-coin HVZK protocol $\langle P_L, V_L \rangle$) to get a simulated transcript $(\alpha_1, c_1, \cdots, \alpha_{t-1}, c_{t-1}, \alpha_t)$; Then, by running the malicious V^* as a subroutine, S tries to mimic the simulated transcript in the Phase-2 interactions (in case V^* successfully finishes Phase-1), by setting the outcome of the underlying coin-tossing to be c_i for each i, $1 \leq i \leq t-1$. Then, the ZK property can be easily derived from the simulation indistinguishability and equivocability of the underlying (generic yet practical) equivocal

The transformed ZK argument $\langle P, V\rangle$

Common input. An element $x \in L \cap \{0,1\}^n$.

P private input. An $\mathcal{NP}$-witness w for $x \in L$.

Phase-1. On a security parameter n, V selects a OWF f_V that admits Σ-protocols, randomly selects two elements in the domain of f_V, x_V^0 and x_V^1 of length n, and computes $y_V^0 = f_V(x_V^0)$ and $y_V^1 = f_V(x_V^1)$. Finally, V sends (y_V^0, y_V^1) to P, and proves to P that it knows the preimage of either y_V^0 or y_V^1 by executing the Σ_{OR}-protocol on (y_V^0, y_V^1) and playing the role of the knowledge prover. We denote by S_{OR} the perfect SHVZK simulator of the underlying Σ_{OR}-protocol. The witness used by V during the execution of the Σ_{OR}-protocol is x_V^b for a randomly chosen bit b in $\{0,1\}$. If V successfully finishes the Σ_{OR}-protocol, P moves into Phase-2; otherwise P aborts.

Phase-2. For $i := 1$ to t do: (Step-3 and Step-4 below can be merged with Step-1 and Step-2 of the following iteration.)

Step-1. If $i = t$ then P runs P_L to compute $\alpha_t = P_L(x, w, c_1, \cdots, c_{t-1})$, sends α_t to V, and the protocol $\langle P, V\rangle$ goes to "**Verifier's decision**". If $i \leq t-1$, the prover P randomly selects $c_P^{(i)}$ of length k, and runs the perfect SHVZK simulator $S_{OR}((y_V^0, y_V^1), c_P^{(i)})$ (as described in Section 2) to get a simulated transcript, denoted $((\hat{a}_0^{(i)}, \hat{a}_1^{(i)}), c_P^{(i)}, (\hat{e}_0^{(i)}, \hat{z}_0^{(i)}, \hat{e}_1^{(i)}, \hat{z}_1^{(i)}))$, pretending that it "knows" the preimage of either y_V^0 or y_V^1. Then, P runs P_L to compute $\alpha_i = P_L(x, w, c_1, \cdots, c_{i-1})$, where c_0 is set to be an empty string. Finally, P sends $((\hat{a}_0^{(i)}, \hat{a}_1^{(i)}), \alpha_i)$ to V.

Step-2. The verifier V sends back P a random challenge $c_V^{(i)}$ of length k.

Step-3. The prover P computes $c_i = c_P^{(i)} \oplus c_V^{(i)}$ and sends $(c_P^{(i)}, (\hat{e}_0^{(i)}, \hat{z}_0^{(i)}, \hat{e}_1^{(i)}, \hat{z}_1^{(i)}))$ to the verifier V.

Step-4. The verifier V checks that whether or not $((\hat{a}_0^{(i)}, \hat{a}_1^{(i)}), c_P^{(i)}, (\hat{e}_0^{(i)}, \hat{z}_0^{(i)}, \hat{e}_1^{(i)}, \hat{z}_1^{(i)}))$ is an accepting conversation for showing the knowledge of the preimage of either y_V^0 or y_V^1. If it is accepted then V computes $c_i = c_V^{(i)} \oplus c_P^{(i)}$, otherwise V aborts.

Verifier's decision The verifier accepts x if and only if $V_L(x, \alpha_1, c_1, \cdots, \alpha_{t-1}, c_{t-1}, \alpha_t)$ outputs 1.

Figure 3. The generic yet practical transformation from public-coin HVZK to (normal) ZK argument.

commitment scheme. The fact that our transformation preserves statistical/perfect ZK is from the observation that, according to the analyses of the simulation procedure of the equivocability simulator $S = (S^c, S^d)$ presented in the proof of Theorem 3.1, except for exponentially negligible probabilities the ZK simulator generates exactly the correct conversation.

Computational soundness.

Note that if the underlying equivocal commitment scheme (for coin-tossing) was perfectly-binding, the soundness of the transformed protocol $\langle P, V\rangle$ would trivially follows from the soundness of the starting public-coin HVZK protocol $\langle P_L, V_L\rangle$. This is the case in [31], where the nicely explored DDH-based (simulatable) commitment scheme is perfectly-binding. But, for our case, the underlying equivocal commitment scheme is actually *computational binding* (actually, it seems that any equivocal commitment scheme intrinsically could not be perfectly-binding). Also, for any starting public-coin HVZK protocol, we cannot count on the special soundness as guaranteed by Σ-protocols. Thus the soundness proof here is (a bit) more complicated and subtle.

The key observation here is that, conditioned on a malicious polynomial-time (wlog, deterministic) P^* can successfully convince the honest verifier V of a false statement "$x \in L$" while $x \notin L$, then there must exist an i, $1 \leq i \leq t-1$, a positive polynomial $q(\cdot)$ and two disjoint sets, denoted $\mathcal{S}$ and $\mathcal{S}'$, where $\mathcal{S} \subseteq \{0,1\}^k$ and $\mathcal{S}' \subseteq \{0,1\}^k$ and $|\mathcal{S}| \geq \frac{1}{q(n)} \cdot 2^k$ and $|\mathcal{S}'| \geq \frac{1}{q(n)} \cdot 2^k$, such that P^* can decommit $(\hat{a}_0^{(i)}, \hat{a}_0^{(i)})$ to a value, denoted $c_{P^*}^{(i)} (\in \{0,1\}^k)$, for all random challenges $c_V^{(i)}$'s fallen into the set $\mathcal{S}$, and decommit $(\hat{a}_0^{(i)}, \hat{a}_0^{(i)})$ to another different value, denoted $c_{P^*}^{(i)\prime} (\in \{0,1\}^k / c_{P^*}^{(i)})$, for all random challenges $c_V^{(i)}$'s fallen

The algorithm $E(1^n, y)$

$x' \xleftarrow{R} \{0,1\}^n$; $y' \longleftarrow f_V(x')$.
$b \xleftarrow{R} \{0,1\}$.
Set y_V^b be y', x_V^b be x' and y_V^{1-b} be y.
$i \xleftarrow{R} \{1, 2, \cdots, t-1\}$.
S runs P^* as a subroutine and acts accordingly by playing the role of the honest verifier, until E receives from P^* the message $(c_{P^*}^{(i)}, (\hat{e}_0^{(i)}, \hat{z}_0^{(i)}, \hat{e}_1^{(i)}, \hat{z}_1^{(i)}))$. Note that in the interactions of Phase-1, E uses $x' = x_V^b$ as its witness.
If $(c_{P^*}^{(i)}, (\hat{e}_0^{(i)}, \hat{z}_0^{(i)}, \hat{e}_1^{(i)}, \hat{z}_1^{(i)}))$ is valid (i.e., P^* correctly decommits $(\hat{a}_0^{(i)}, \hat{a}_1^{(i)})$ with respect to the random challenge $c_V^{(i)}$ sent by E), then E rewinds P^* to the state that P^* just sent $(\hat{a}_0^{(i)}, \hat{a}_1^{(i)})$.
$c_V^{(i)\prime} \xleftarrow{R} \{0,1\}^k / c_V^{(i)}$.
Returns back $c_V^{(i)\prime}$ to P^* and runs P^* further, looking forward to receiving again a valid decommitment message of the same $(\hat{a}_0^{(i)}, \hat{a}_1^{(i)})$ but to a different value $c_{P^*}^{(i)\prime} \neq c_{P^*}^{(i)}$ from which E can extract the preimage of either y_V^0 or y_V^1.

Figure 4. The algorithm $E(1^n, y)$

into the set $\mathcal{S}'$. The underlying reason is that, if the above does not hold then the computational soundness of the protocol $\langle P, V\rangle$ just follows from the soundness of the starting public-coin HVZK protocol $\langle P_L, V_L\rangle$.

Now, suppose the transformed protocol $\langle P, V\rangle$ does not satisfy computational soundness, then we will use the above observation to reach a contradiction to the one-wayness of f_V. Specifically, we will construct a PPT algorithm E that on common input y outputs a preimage of y under f_V with non-negligible probabilities in polynomial-time. The algorithm E is depicted in Figure 4 (page 14).

Suppose P^* can convince the honest verifier of a false statement "$x \in L$" while $x \notin L$ with non-negligible probability $\frac{1}{p(n)}$ for some positive polynomial $p(\cdot)$, then it is easy to check that E will output a preimage of either y_V^0 or y_V^1 with probability at least $\frac{1}{t \cdot q(n) \cdot p(n)}$ in polynomial-time. Due to the (perfect) WI property of the Σ_{OR}-protocol executed in Phase-1, we conclude with probability at least $\frac{1}{2 \cdot t \cdot q(n) \cdot p(n)}$ E will output the preimage of $y = y_V^{1-b}$ in polynomial-time, which violates the one-wayness of f_V. $\square$

Acknowledgments. We are grateful to the anonymous referee of ECCC for kindly referring us the detailed list of earlier works on converting HVZK to normal ZK [2, 36, 12, 13, 35, 26].

References

[1] D. Beaver. Adaptive Zero-Knowledge and Computational Equivocation. In *ACM Symposium on Theory of Computing*, pages 629-638, 1996.

[2] M. Bellare, S. Micali and R. Ostrovsky. The (True) Complexity of Statistical Zero-Knowledge. In *ACM Symposium on Theory of Computing*, pages 494-502, 1990.

[3] 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.

[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. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *IEEE Symposium on Foundations of Computer Science*, pages 136-145, 2001.

[6] R. Canetti and M. Fischlin. Universal Composable Commitments. In *J. Kilian (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2001, LNCS 2139*, pages 19-40. Springer-Verlag, 2001.

[7] R. Canetti, Y. Lindell, R. Ostrovsky and A. Sahai. Universally Composable Two-Party and Multi-Party Secure Computation. In *ACM Symposium on Theory of Computing*, pages 494-503, 2002.

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

[9] R. Cramer, I. Damgård and P. MacKenzie. Efficient Zero-Knowledge Proof of Knowledge Without Intractability Assumptions. In *(Ed.): Third International Workshop on Practice and Theory in Public-Key Cryptosystems, PKC 2000, LNCS ?*, pages ?, Springer-Verlag, 2000.

[10] R. Cramer, I. Damgård 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.

[11] I. Damgård. On the Existence of Bit Commitment Schemes and Zero-Knowledge Proofs. In *G. Brassard (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1989, LNCS 435*, pages 17-27. Springer-Verlag, 1989.

[12] I. Damgård. Interactive Hashing can Simplify Zero-Knowledge Protocol Design Without Computational Assumptions. In *D. R. Stinson (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1993, LNCS 773*, pages 100-109. Springer-Verlag, 1993.

[13] I. Damgård, O. Goldreich, T. Okamoto and A. Wigderson. Honest Verifier vs. Dishonest Verifier in Public-Coin Zero-Knowledge Proofs. In *D. Coppersmith (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1995, LNCS 403*, pages 325-338. Springer-Verlag, 1995.

[14] I. Damgård and J. Groth. Non-interactive and reusable non-malleable commitment schemes. In *ACM Symposium on Theory of Computing*, pages 426-437, 2003.

[15] I. Damgård and J. B. Nielsen. Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor. In *M. Yung (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2002, LNCS 2442*, pages 581-596. Springer-Verlag, 2002.

[16] G. Di Crescenzo, Y. Ishai and R. Ostrovsky. Non-Interactive and Non-Malleable Commitment In *ACM Symposium on Theory of Computing*, pages 141-150, 1998.

[17] G. Di Crescenzo, J. Katz, R. Ostrovsky and A. Smith. Efficient and Non-Interactive Non-Malleable Commitments. In *B. Pfitzmann (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 2001, LNCS 2045*, pages 40-59. Springer-Verlag, 2001.

[18] G. Di Crescenzo and R. Ostrovsky. On Concurrent Zero-Knowledge with Pre-Processing. In *M. J. Wiener (Ed.): Advances in Cryptology-Proceedings of CRYPTO 1999, LNCS 1666*, pages 485-502. Springer-Verlag, 1999.

[19] C. Dwork, M. Naor and A. Sahai. Concurrent Zero-Knowledge. *Journal of the ACM*, 51(6): 851-899, 2004.

[20] U. Feige. Alternative Models for Zero-Knowledge Interactive Proofs. Ph.D. Thesis, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel, 1990. Available from: `http://www.wisdom.weizmann.ac.il/∼feige`.

[21] 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.

[22] M. Fischlin and R. Fischlin. Efficient Non-Malleable Commitment Schemes. In *M. Bellare (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2000, LNCS 1880*, pages 413-431. Springer-Verlag, 2000.

[23] O. Goldreich and H. Krawczky. On the Composition of Zero-Knowledge Proof Systems. *SIAM Journal on Computing*, 25(1): 169-192, 1996.

[24] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game-A Completeness Theorem for Protocols with Honest Majority. In *ACM Symposium on Theory of Computing*, pages 218-229, 1987.

[25] 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.

[26] O. Goldreich, A. Sahai and S. Vadhan. Honest-Verifier Statistical Zero-Knowledge Equals General Statistical Zero-Knowledge. In *ACM Symposium on Theory of Computing*, pages 399-408, 1998.

[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. Journal version appears in *SIAM Journal on Computing*, 18(1): 186-208, 1989.

[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] P. MacKenzie and K. Yang. On simulation-Sound Trapdoor Commitments. In *C. Cachin and J. Camenisch (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 2004, LNCS 3027* , pages 382-400. Springer-Verlag, 2003.

[31] D. Micciancio and E. Petrank. Simulatable Commitments and Efficient Concurrent Zero-Knowledge. In *E. Biham (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 2003, LNCS 2656* , pages 140-159. Springer-Verlag, 2003.

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

[33] M. Naor, R. Ostrovsky, R. Venkatesan and M. Yung. Perfect Zero-Knowledge Arguments for NP Using Any One-Way Permutation. *Journal of Cryptology*, 11(2): 87-108, 1998.

[34] M. Naor and M. Yung. Public-Key Cryptosystems Provably Secure Against Chosen Ciphertext Attacks. In *ACM Symposium on Theory of Computing*, pages 427-437, 1990.

[35] T. Okamoto. On Relationships between Statistical Zero-Knowledge Proofs. In *ACM Symposium on Theory of Computing*, pages 649-658, 1996.

[36] R. Ostrovsky, R. Venkatesan and M. Yung. Interactive Hashing Simplifies Zero-Knowledge Protocol Design. In *T. Helleseth (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 1993, LNCS 765* , pages 267-273. Springer-Verlag, 1993.

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

[38] A. C. Yao. How to Generate and Exchange Secrets. In *IEEE Symposium on Foundations of Computer Science*, pages 162-167, 1986.

A Blum's protocol for DHC [3]

Blum's protocol for the $\mathcal{NP}$-Complete language DHC (Directed Hamiltonian Cycle) is n-parallel repetitions of Blum's basic protocol for proving the knowledge of Hamiltonian cycle on a given directed graph G [3]. The following is the description of Blum's *basic* protocol for DHC:

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).