Reduction zero-knowledge*

ZHAO Yunlei[1,2**], DENG Xiaotie[2], LEE C. H.[2] and ZHU Hong[1]

(1. Department of Computer Science, Fudan University, Shanghai 200433, China; 2. Department of Computer Science, City University of Hong Kong, Hong Kong, China)

Received June 2, 2003; revised August 6, 2003

Abstract The nature of zero-knowledge is re-examined and the evidence for the following belief is shown: the classic simulation based definitions of zero-knowledge (simulation zero-knowledge) may be somewhat too strong to include some "nice" protocols in which the malicious verifier seems to learn nothing but we do not know how to construct a zero-knowledge simulator for it. To overcome this problem a new relaxation of zero-knowledge, reduction zero-knowledge, is introduced. It is shown that reduction zero-knowledge just lies between simulation zero-knowledge and witness indistinguishability. Under the assumption of existence of one-way permutations a 4-round public-coin reduction zero-knowledge proof system for *NP* is presented and in practice this protocol works in 3 rounds since the first verifier's message can be fixed once and for all.

Keywords: zero-knowledge, non-interactive zero-knowledge, bit commitment, zap, witness indistinguishability.

The notion of zero-knowledge (ZK) was first put forward by Goldwasser et al. to illustrate situations where a prover reveals nothing other than the verity of a given statement to an even malicious verifier[1]. The definitions of zero-knowledge were extensively investigated by Goldreich et al.[2] and the generality of this notion was demonstrated by Goldreich et al. by showing that any *NP*-statement can be proven in zero-knowledge provided that commitment schemes exist[3]. Subsequently, related notions have been proposed, in particular, zero-knowledge argument, witness indistinguishability, zero-knowledge proof of knowledge, non-interactive zero-knowledge. For a good survey in this field, the readers are referred to Ref. [4]. By now, zero-knowledge has played a central role in the field of cryptography and is the accepted methodology to define and prove security of various cryptographic tasks.

Informally speaking, a protocol $\langle P, V \rangle$ for a language L is zero-knowledge if the view of an even malicious verifier in its interaction with the honest prover on input $x \in L$ can be simulated by a simulator itself on x without any interactions. In the rest of this paper we denote by classic simulation based ZK simulation zero-knowledge. The view of the even malicious verifier in its real interactions with the honest prover is a transcript including all the messages exchanged between the honest prover and the malicious verifier. That is, all the messages sent by the honest prover and all the messages sent by the even malicious verifier. We remark that as a privacy criterion zero-knowledge is a quite strong requirement. For some specific applications we may not need such a strong privacy requirement or we may need a protocol with some extra properties which are not guaranteed to be preserved for zero-knowledge protocols (e.g. we may wish the privacy property of our protocol can be preserved under parallel composition, which is not held for zero-knowledge protocols[5]). For this purpose, a relaxation of zero-knowledge, witness indistinguishability (WI), was put forward by Feige et al.[6]. Loosely speaking, a WI protocol is an interactive system in which the view of any polynomial-size verifier is "computationally independent" of the witness used by the honest prover as its auxiliary private input. WI is a relaxation of ZK. It means all ZK protocols are also WI protocols. We remark that although WI is preserved under parallel composition but in general it is believed that ZK is a significantly stronger security notion than WI. That is, a WI protocol may lose much security in comparison with a ZK protocol.

According to the definition of ZK, to simulate the view of a malicious verifier the simulator needs to generate all the messages sent by the honest prover and all the messages sent by this malicious verifier.

* Supported by the National Natural Science Foundation of China (Grant No. 60273045) and the Ministry of Science and Technology of China (Grant No. 2001CCA03000)

** To whom correspondence should be addressed. E-mail: CSYLZHAO@cityu.edu.hk

To simulate all the messages sent by the even malicious verifier the simulator needs to incorporate this verifier into its coding as a subroutine and normally treats it as an oracle. The difficulty occurs when simulating the messages sent by the honest prover since the simulator does not know the prover's auxiliary private input. The normal method employed by the simulator, especially for constant-round zero-knowledge protocol, is the rewinding technique developed by Goldreich et al.[7]. To facilitate the rewinding technique it is normally required that there is a "determining" verifier step which determines some or all of its subsequent behaviors in an execution of the protocol. This "determining" verifier step is typically implemented by commitment schemes and can be viewed as a "promise" made by the verifier that he will learn nothing in his interactions with the prover. Normally, without this determining step we do not know how to construct a zero-knowledge simulator. But in some cases this determining message seems to only facilitate the simulator and without it the malicious verifier does not seem to get more advantages. This is more obvious in the Feige-Lapidot-Shamir (FLS) paradigm[8] that was introduced in the context of non-interactive zero-knowledge (NIZK), and has been extensively used in recent advances of zero-knowledge. We will give the description of the FLS paradigm and discuss the above phenomenon in detail in Section 3.

In this paper, we overcome the above problems by introducing another relaxation of zero-knowledge, which we call reduction zero-knowledge. Our approach is based on the belief that the messages sent by the honest prover play a much more important role in the malicious verifier's ability of learning "knowledge" from its interactions with the honest prover. Informally speaking, a protocol 1 $\langle P_1, V_1 \rangle$ is reduced to another protocol 2 $\langle P_2, V_2 \rangle$ if all the messages the malicious verifier V_1^* can extract from the honest prover P_1 (that is, all the messages sent by P_1) in protocol 1 can also be extracted by a malicious verifier V_2^* from the honest prover P_2 in protocol 2. We say protocol 1 is reduction zero-knowledge if protocol 1 can be reduced to protocol 2 and protocol 2 is simulation zero-knowledge. Note that we do not require V_2^* to get the whole view of V_1^*. We just focus on the messages from the honest prover and ignore the messages from the malicious verifier. We remark that this treatment is reasonable to compare the "knowledge" extracting abilities for different malicious verifiers in different protocols. On one hand, as discussed above we believe that the honest prover's messages are critical for the malicious verifier to learn "knowledge". On the other hand, zero-knowledge is a property of honest prover and it is hard or infeasible to compare the behaviors of different malicious verifiers in different protocols.

As we will show, reduction zero-knowledge just lies between classic simulation zero-knowledge and witness indistinguishability. There are two major contributions of our reduction zero-knowledge. One is that it introduces reduction between different protocols and extends the approaches to characterize the nature of zero-knowledge. Note that reduction is a widely used paradigm in the field of computer science. Another is that reduction zero-knowledge can allow us to design more efficient protocols, especially to simplify the verifier greatly, for which although we do not know how to construct a zero-knowledge simulator but it is believed that it loses little security in comparison with a corresponding simulation zero-knowledge protocol.

1 Preliminaries

In this section we recall the notions, the definitions and the constructions that we will utilize in this paper.

Definition 1 (negligible function). A function $f: N \to [0, 1]$ is negligible if for all polynomials $p(\cdot)$, and for all sufficiently large k, $f(k) < \frac{1}{p(k)}$.

Definition 2 (probability ensembles). A probability ensemble X is a family $X = \{X_n\}_{n \geqslant 1}$ such that X_n is a probability distribution on some finite domain.

Definition 3 (computational indistinguishability). Two probability ensembles $\{X_n\}_{n \in N}$ and $\{Y_n\}_{n \in N}$ are computational indistinguishable if for every probabilistic polynomial time (PPT) algorithm A and for all sufficiently large n, $|\Pr(A(X_n) = 1) - \Pr(A(Y_n) = 1)|$ is negligible in n.

Definition 4 (statistical indistinguishability). The statistical difference between two distributions, X and Y, is defined by $\Delta(X, Y) = \frac{1}{2} \cdot \sum_{\alpha} |\Pr[X = \alpha] - \Pr[Y = \alpha]|$. Two probability ensembles

$\{X_n\}_{n\in N}$ and $\{Y_n\}_{n\in N}$ are statistical indistinguishable if for sufficiently large n $\Delta(X_n, Y_n)$ is negligible in n.

Note that if $\{X_n\}_{n\in N}$ and $\{Y_n\}_{n\in N}$ are statistical indistinguishable then there are no algorithms even with unbounded computational power to distinguish them with non-negligible probability.

Definition 5 (interactive proof system). A pair of probabilistic machines, $\langle P, V\rangle$, is called an interactive proof system for L if V is polynomial-time and the following conditions hold:

(1) Completeness. For every $x \in L$, $\Pr[\langle P, V\rangle(x) = 1] = 1$.

(2) Soundness. For all sufficiently large n and every $x \notin L$ of length n and every interactive machine B (even with unbounded computational power), $\Pr[\langle B, V\rangle(x) = 1]$ is negligible in n.

An interactive proof system is called a public-coin proof system if at each round the prescribed verifier can only send a random string to the prover.

Definition 6 (one-round perfect-binding bit commitment). A perfect-binding bit commitment is a pair of probabilistic polynomial-time algorithms (S, R), satisfying:

(1) Completeness. $\forall k$, $\forall v$, let $c = C_{s_v}(1^k, v)$ and $d = (v, s_v)$, where C is a PPT commitment algorithm that uses s_v as its randomness and d is the corresponding decommitment to c, it holds that $\Pr[(c, d) \xleftarrow{R} S(1^k, v): R(1^k, c, v, d) = \mathrm{YES}] = 1$, where k is a security parameter.

(2) Computational hiding. For every v, u of equal $p(k)$-length, where p is a positive polynomial and k is the security parameter, the random variables $C_{s_v}(1^k, v)$ and $C_{s_u}(1^k, u)$ are computationally indistinguishable.

(3) Perfect binding. For every v, u of equal $p(k)$-length, the random variables $C_{s_v}(1^k, v)$ and $C_{s_u}(1^k, u)$ have a disjoint support. That is, for every v, u, and m, if $\Pr[S_{s_v}(1^k, v) = m]$ and $\Pr[S_{s_u}(1^k, u) = m]$ are both positive then $u = v$ and $s_v = s_u$.

A one-round commitment scheme as above can be constructed based on any one-way function[4].

Definition 7 (hash-based commitment scheme HC). A hash-based commitment scheme (HC) is a pair of probabilistic polynomial-time algorithms (S, R), satisfying:

(1) Completeness. $\forall k$, $\forall v$, $\Pr[(c, d) \xleftarrow{R} S(1^k, v): R(1^k, c, v, d) = \mathrm{YES}] = 1$, where k is a security parameter, $c = \mathrm{HC}(1^k, v)$ and HC is a PPT hash-based commitment algorithm used by S.

(2) Statistical hiding. For every v, u of equal $p(k)$-length, where p is a positive polynomial, the random variables $\mathrm{HC}(1^k, v)$ and $\mathrm{HC}(1^k, u)$ are statistically indistinguishable.

(3) Computational binding. For all PPT algorithm ADV, and all sufficiently large k, $\Pr[(c, v_1, v_2, d_1, d_2) \xleftarrow{R} \mathrm{ADV}(1^k): v_1 \neq v_2 \wedge R(1^k, c, v_1, d_1) = \mathrm{YES} = R(1^k, c, v_2, d_2)]$ is negligible in k.

The readers are referred to Ref. [9] for the construction of such schemes, which are based on the assumption that collision-free hash functions exist.

Protocol (coin flipping over the telephone)[10]. Coin flipping over the telephone is a direct application of commitment schemes and is the way to generate random string via interactions. Suppose Alice and Bob want a fair, common random-string: r in $\{0, 1\}^k$. Using commitment schemes the approach is as follows: Alice uniformly selects a string v in $\{0, 1\}^k$ and sends $y = C_s(1^k, v)$ to Bob, where C is the one-round perfect binding commitment algorithm while using s as its randomness. Then Bob uniformly selects a string u in $\{0, 1\}^k$ and sends u in the clear to Alice. At last Alice decommits to y (by sending (v, s) to Bob) and the final common random string is $r = v \oplus u$.

We remark that in the above protocol if Bob is honest then r is a truly random string. If Bob is malicious but Alice is honest then r is pseudorandom (that is, any PPT algorithm can not distinguish r from a truly random string in $\{0, 1\}^k$ with non-negligible probability).

Definition 8 (zero-knowledge). Let $\langle P, V\rangle$ be an interactive proof system for a language L. We denote by $\mathrm{view}_V^P(x)$ a random variable describing the transcript of messages exchanged between the honest

prover P and the (malicious) verifier V^* in an execution of the protocol on common input x. 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 interactive machine S such that the following two probability distributions are computationally indistinguishable: $\{\text{view}_{V^*}^{P}(x)\}_{x\in L}$ and $\{S(x)\}_{x\in L}$. Machine S is called a zero-knowledge simulator for $\langle P, V\rangle$. In the rest of this paper we refer to zero-knowledge in this definition as simulation zero-knowledge.

Definition 9 (black-box zero-knowledge). Let $\langle P, V\rangle$ be an interactive proof system for a language L. We say that $\langle P, V\rangle$ is black-box zero-knowledge if there exists a probabilistic (expected) polynomial-time interactive machine S, such that for every probabilistic polynomial time interactive machine V^* and for every $x\in L$, the following two probability distributions are computationally indistinguishable: $\{\text{view}_{V^*}^{P}(x)\}_{x\in L}$ and $\{S^{V^*}(x)\}_{x\in L}$. Machine S is called a black-box zero-knowledge simulator for $\langle P, V\rangle$.

Definition 10 (non-interactive zero-knowledge NIZK). Let NIP and NIV be two interactive machines and NIV is also probabilistic polynomial-time, and let $NI\sigma\text{Len}$ be a positive polynomial. We say that $\langle\text{NIP}, \text{NIV}\rangle$ is an NIZK proof system for an NP-language L, if the following conditions hold:

(1) Completeness. $\forall x\in L$ of length n, σ of length $NI\sigma\text{Len}(n)$, and NP-witness w for x, $\Pr[\Pi \xleftarrow{R} \text{NIP}(\sigma, x, w): \text{NIV}(\sigma, x, \Pi) = \text{YES}] = 1$.

(2) Soundness. $\forall x\notin L$ of length n, $\Pr[\sigma \xleftarrow{R} \{0,1\}^{NI\sigma\text{Len}(n)}; \exists \Pi s.t. \text{NIV}(\sigma, x, \Pi) = \text{YES}]$ is negligible in n.

(3) Zero-Knowledgeness. $\exists$ a PPT simulator NIS such that, $\forall$ sufficient large n, $\forall x\in L$ of length n and NP-witness w for x, the following two distributions are computationally indistinguishable: $[(\sigma', \Pi') \xleftarrow{R} \text{NIS}(x): (\sigma', \Pi')]$ and $\Pr[\sigma \xleftarrow{R} \{0,1\}^{NI\sigma\text{Len}(n)}; \Pi \xleftarrow{R} \text{NIP}(\sigma, x, w): (\sigma, \Pi)]$.

The non-interactive zero-knowledge proof system for NP can be constructed based on any one-way permutation[8].

Definition 11 (witness indistinguishability WI). Let $\langle P, V\rangle$ be an interactive proof system for a language $L\in NP$, and let R_L be a fixed NP witness relation for the language L. That is $x\in L$ if there exists a w such that $(x, w)\in R_L$. We denote by $\text{view}_{V^*(z)}^{P(w)}(x)$ a random variable describing the transcript of all messages exchanged between V^* and P in an execution of the protocol on common input x, when P has auxiliary input w and V^* has auxiliary input z. We say that $\langle P, V\rangle$ is witness indistinguishable for R_L if for every PPT interactive machine V^*, and every two sequences $W^1 = \{w_x^1\}_{x\in L}$ and $W^2 = \{w_x^2\}_{x\in L}$, so that $(x, w_x^1)\in R_L$ and $(x, w_x^2)\in R_L$, the following two probability distributions are computational indistinguishable:

$$\{x, \text{view}_{V^*(z)}^{P(w_x^1)}(x)\}_{x\in L, z\in\{0,1\}^*} \text{ and}$$

$$\{x, \text{view}_{V^*(z)}^{P(w_x^2)}(x)\}_{x\in L, z\in\{0,1\}^*}.$$

WI is a relaxation of ZK. All ZK protocols are also WI protocols but the opposite direction is not guaranteed to be true. Actually, it is assumed that ZK is a significantly stronger notion of security than WI[11]. That is a WI protocol may lose much security in comparison with a ZK protocol.

Protocol (Dwork-Naor 2-round WI proof for NP[12]). In Ref. [12] such a protocol is named a zap.

Let L be an NP-Complete language and R_L be its corresponding NP relation. On a security parameter k, let p be a positive polynomial and $x\in L$ be the common input and w be the corresponding NP-witness.

Step 1. The verifier V uniformly selects (and fixes once and for all) $p(k)$ random strings $R_V = r_1, \cdots, r_{p(k)}$ of length $NI\sigma\text{Len}(k)$ each and sends them to prover P.

Step 2. The prover P uniformly selects a random string r of length $NI\sigma\text{Len}(k)$. Then for each i, $1\leqslant i\leqslant p(k)$, P computes an NIZK proof with respect to the common random string $r\oplus r_i$. At last, P sends r and all these $p(k)$ NIZK proofs $\{\text{NIZK}(x, r\oplus r_i)\}_{i=1}^{p(k)}$ to the verifier.

A very interesting property of the Dwork-Naor 2-round WI is that the R_V selected by the verifier can be fixed once and for all. That is, once R_V is selected

then it is also fixed for all subsequent executions of the protocol. It means that R_V can be viewed as a public-key of the verifier. Zap is also used in Refs. [13, 14].

2 Definition of reduction zero-knowledge

In this section we introduce another relaxation of simulation zero-knowledge, which we name reduction zero-knowledge, and discuss the relationship among simulation ZK, WI and our new notion.

Roughly speaking, a protocol 1 $\langle P_1, V_1 \rangle$ is reduced to another protocol 2 $\langle P_2, V_2 \rangle$ if the malicious verifier V_2^* in protocol 2 has the "same" "knowledge"-extracting ability as that of V_1^* in protocol 1. To introduce reduction between different protocols, the difficulty is how to compare the knowledge-extracting ability of different malicious verifiers in different protocols. Note that the view of a (malicious) verifier V^* in an execution of the protocol $\langle P, V \rangle$ is a transcript including two parts. One part is all the messages sent by the honest prover P and the other part is all the messages sent by itself in response to messages received from the honest prover. Our approach is to ignore the influence of the messages sent by the malicious verifier on its "knowledge"-extracting ability from the honest prover. It is justified on the following grounds. On one hand, intuitively, the messages sent by the honest prover should play a more critical role for the malicious verifier to extract "knowledge" from the honest prover. On the other hand, zero-knowledge is a property of the honest prover and it is hard to compare the behaviors of different malicious verifiers in different protocols. The formal definition is given below.

Definition 12 (knowledge-extraction reduction). Let protocol 1 $\langle P_1, V_1 \rangle$ and protocol 2 $\langle P_2, V_2 \rangle$ be two protocols for the same language $L \in NP$. We say protocol 1 is knowledge-extraction reduced to protocol 2 if for every malicious verifier V_1^* there exists a malicious verifier V_2^* such that for each $x \in L$ and its witness w, all the messages received by the malicious verifier V_2^* from the honest prover P_2 (which uses w as its auxiliary private input) in an execution of protocol 2 on common input x and all the messages received by the malicious verifier V_1^* from the honest prover P_1 (which uses w as its auxiliary private input) in an execution of protocol 1 on common input x are computationally indistinguishable.

Definition 13 (reduction zero-knowledge). We say protocol 1 $\langle P_1, V_1 \rangle$ is reduction zero-knowledge if it can be knowledge-extraction reduced to a protocol 2 $\langle P_2, V_2 \rangle$ and protocol 2 is simulation zero-knowledge.

Theorem 1. Any simulation zero-knowledge protocol is also reduction zero-knowledge and reduction zero-knowledge implies witness indistinguishability. That is, as a security criterion, reduction zero-knowledge is weaker than simulation zero-knowledge and stronger than witness indistinguishability.

Proof. First, it can be easily verified that any simulation zero-knowledge protocol is also reduction zero-knowledge since any simulation zero-knowledge protocol is trivially knowledge-extraction reduced to itself.

Next, we show that reduction zero-knowledge implies witness indistinguishability.

By the definition of witness indistinguishability, a protocol for an NP-language L is witness indistinguishable if for any malicious verifier V^* the views of V^* in two independent executions of the protocol on the same common input $x \in L$ while the honest prover using different auxiliary witnesses are computationally indistinguishable. We remark that to prove a protocol is witness indistinguishable the messages sent by the malicious verifier itself in its view can be ignored. That is, if the messages received by the malicious verifier V^* from the honest prover in two independent executions of the protocol on the same common input $x \in L$ (while the honest prover uses different auxiliary witnesses) are computationally indistinguishable then this protocol is witness indistinguishable for L. The reason is that we can view V^* as a PPT next message function NM_{V^*} that computes its next message on common input, its auxiliary input and the messages it has received from the honest prover. Then according to the definition of computational indistinguishability if the messages received by V^* from the honest prover in two independent execution of the protocol on the same common input $x \in L$ (while the honest prover using different auxiliary witnesses) are computationally indistinguishable then the messages sent by the malicious verifier are also computationally indistinguishable.

Now we show that for a reduction zero-knowledge protocol (for an NP-language L) the messages received by the malicious verifier from the honest prover in two independent execution of the protocol on the same common input $x \in L$ while the honest prover using different auxiliary witnesses are computationally indistinguishable. Note that this will be enough to establish the theorem according to above discussions. The reason is that for a reduction zero-knowledge protocol there exists a simulation zero-knowledge protocol for the same NP-language L to which the reduction zero-knowledge protocol can be knowledge-extraction reduced. However, for any malicious verifier of the simulation zero-knowledge protocol, the messages received by this malicious verifier from the honest prover of the simulation zero-knowledge protocol in two independent executions of the simulation zero-knowledge protocol on the same common input $x \in L$ (while the honest prover using different auxiliary witnesses) are computationally indistinguishable since simulation zero-knowledge is also witness indistinguishability. Then according to the definition of knowledge-extraction reduction, we conclude that the messages received by a malicious verifier of the reduction zero-knowledge protocol from honest prover of the reduction zero-knowledge in two independent executions of the reduction zero-knowledge protocol on the same common input $x \in L$ while the honest prover using different auxiliary witness are also computationally indistinguishable.

We remark that although reduction zero-knowledge is a relaxation notion of the normal simulation zero-knowledge, but in contrast to witness indistinguishability it is assumed that a reduction zero-knowledge protocol loses little security in comparison with a corresponding simulation zero-knowledge protocol. This can be seen more clearly in the Feige-Lapidot-Shamir (FLS) paradigm which is to be addressed in the next section.

3 Reduction zero-knowledge proof system for *NP*

In this section, we first present the Feige-Lapidot-Shamir (FLS) paradigm and then give a 4-round public-coin reduction zero-knowledge for NP using the FLS paradigm under the assumption of existence of one-way permutations. Furthermore, in practice our protocol works in 3-round since the first round messages sent by the verifier can be fixed once and for all.

3.1 Feige-Lapidot-Shamir (FLS) paradigm

Let L be an NP language and denote by R_L the corresponding NP relation for L. Suppose the common statement is $x \in L$ and the prescribed prover has an auxiliary private input w such that $(x, w) \in R_L$. The FLS paradigm consists of two phases and converts a witness indistinguishability protocol into a zero-knowledge protocol.

In the first phase, called the generation phase, the prover and the verifier generate a string, denoted τ, which can be viewed as the transcript generated in the first phase, and fix a relation R' for it. In this phase, both the prover and the verifier ignore the inputs (the common input and their auxiliary inputs) and so this phase can be performed even before knowing what is the theorem that will be proven. It means that even a malicious verifier cannot learn "knowledge" from the honest prover in this phase since the prover is oblivious of his auxiliary private input up to now. It is also required that even a malicious prover cannot get a witness w' for τ from its interactions with the honest verifier in this phase, such that $(\tau, w') \in R'$.

In the second phase, called the WI proof phase, the prover proves (using a WI protocol) that either there exists a w such that $(x, w) \in R_L$ or there exists a w' such that $(\tau, w') \in R'$. In practice, in the second phase the honest prover just uses its private input w as the witness to prove that $(x, w) \in R_L$. However, using the malicious verifier as a subroutine a simulator can get a witness w' for τ such that $(\tau, w') \in R'$ and then proves (using the WI protocol) that it knows a witness w' such that $(\tau, w') \in R'$. The witness indistinguishability property is used to ensure that the malicious verifier cannot tell the difference between the real interaction and the simulated one.

3.2 Reduction zero-knowledge for *NP* using FLS paradigm

To provide a reduction zero-knowledge proof system for NP, and according to the definition of reduction zero-knowledge we need to construct two protocols: one is a simulation zero-knowledge proof system for NP, and to which the other protocol can be knowledge-extraction reduced.

For an NP-Complete language L, suppose R_L is the corresponding NP relation for L and the common

input is $x \in L$ of length n. The prover has an auxiliary private input w such that $(x, w) \in R_L$. Under the security parameter n, the prover P uses a one-round perfect binding bit commitment C and the verifier V uses a one-round hash-based bit commitment HC. Using the FLS paradigm, the two protocols are presented below. For both protocols, the first three rounds constitute the generation phase of the FLS paradigm and the last round corresponds to the WI proof phase.

Protocol 1 $\langle P_1, V_1 \rangle$.

Round 1. V_1 first uniformly selects a string v in $\{0,1\}^n$ and computes $hc_v = \mathrm{HC}(1^n, v)$. Then V_1 uniformly selects (fixes once and for all) $p(n)$ random strings $R_{V_1} = r_1, \cdots, r_{p(k)}$ of length $NI\sigma\mathrm{Len}(n)$ each just as does in the first round of the Dwork-Naor two-round WI, where p is a positive polynomial. At last, V_1 sends (hc_v, R_{V_1}) to the prover P_1.

Round 2. P_1 uniformly selects two strings in $\{0,1\}^n$, t and o, and two corresponding random strings s_t and s_o. Then P_1 computes $c_t = C_{s_t}(1^n, t)$ and $c_o = C_{s_o}(1^n, o)$. P_1 then uniformly selects a random string r of length $NI\sigma\mathrm{Len}(n)$ just as does in the second round of the Dwork-Naor 2-round WI. Finally, P_1 sends (c_t, c_o, r) to V_1.

Round 3. V_1 decommits to hc_v. That is V_1 reveals v to P_1.

Round 4. P_1 decommits to c_o by sending (o, s_o) to V_1. Then, with respect to $(x, c_t, c_o, s_o, o \oplus v)$, using the Dwork-Naor 2-round WI P_1 proves the following statement: y = "there exists a w such that $(x, w) \in R_L$ or there exists a string s such that $c_t = C_s(1^n, v \oplus o)$". That is, for each i, $1 \leqslant i \leqslant p(n)$, P_1 computes an NIZK proof for y with respect to common random string $r \oplus r_i$. At last, P_1 sends $(o, s_o, \{\mathrm{NIZK}(y, r \oplus r_i)\}_{i=1}^{p(n)})$ to V_1.

Verifier's decision: If all the $p(n)$ NIZK proofs are acceptable then V_1 accepts $x \in L$, otherwise, rejects.

Protocol 2 $\langle P_2, V_2 \rangle$.

Round 1. V_2 uniformly selects (fixes once and for all) $p(n)$ random strings $R_{V_2} = r_1, \cdots, r_{p(k)}$ of length $NI\sigma\mathrm{Len}(n)$ each just as does in the first round of the Dwork-Naor two-round WI, where p is a positive polynomial. Then V_2 sends R_{V_2} to the prover P_2.

Round 2. P_2 uniformly selects two strings in $\{0,1\}^n$, t and o, and two corresponding random strings s_t and s_o. Then P_2 computes $c_t = C_{s_t}(1^n, t)$ and $c_o = C_{s_o}(1^n, o)$. P_2 then uniformly selects a random string r of length $NI\sigma\mathrm{Len}(n)$ just as does in the second round of the Dwork-Naor 2-round WI. Finally, P_2 sends (c_t, c_o, r) to V_1.

Round 3. V_2 uniformly selects a string v in $\{0,1\}^n$ and sends v to P_2.

Round 4. P_2 decommits to c_o by sending (o, s_o) to V_2. Then, with respect to $(x, c_t, c_o, s_o, o \oplus v)$, using the Dwork-Naor 2-round WI P_2 proves the following statement: y = "there exists a w such that $(x, w) \in R_L$ or there exists a string s such that $c_t = C_s(1^n, v \oplus o)$". That is, for each i, $1 \leqslant i \leqslant p(n)$, P_2 computes an NIZK proof for y with respect to common random string $r \oplus r_i$. At last, P_2 sends $(o, s_o, \{\mathrm{NIZK}(y, r \oplus r_i)\}_{i=1}^{p(n)})$ to V_2.

Verifier's decision: If all the $p(n)$ NIZK proofs are acceptable then V_2 accepts $x \in L$, otherwise, rejects.

For the two protocols presented above, we have the following theorems.

Theorem 2. Assuming collision-resistant hash functions and one-way permutations exist, Protocol 1 is a black-box zero-knowledge proof system for NP.

Proof. The completeness can be easily verified. We focus on soundness and zero-knowledgeness.

Soundness. For any common input $x \notin L$, since the hash-based commitment used by the verifier V is statistically-hiding then a malicious prover P^* (even with unbounded computational power) still cannot learn v before it commits to the value t and o in Round 2 (except for a negligible probability). Then according to the perfect-binding property of the commitment scheme used by the prover, P^* also cannot decommit to c_t as $o \oplus v$ in Round 4 (except for a negligible probability). Also note that since $x \notin L$ there also exists no witness for $x \in L$. It means P^* cannot get a witness for y in Round 4 (except for a negligible probability). Then the soundness of Protocol 1 is fol-

lowed from the soundness of the Dwork-Naor two-round WI proof system.

Zero-knowledgeness. We remark that the same techniques introduced in Ref. [7], i.e. the standard "rewinding" and approximation estimation of the probability on verifier's correct decommitments, can be applied here to show Protocol 1 is really black-box zero-knowledge. Roughly, while oracle accessing V^*, the black-box simulator S^{V^*} works as follows. S^{V^*} first runs V^* to get R_V and hc_v sent by V^* in Round 1. Then for two arbitrary "garbage" values, t' and o' in $\{0,1\}^n$ and two corresponding random strings $s_{t'}$ and $s_{o'}$, S^{V^*} sends $c_{t'} = C_{s_{t'}}(1^n, t')$ and $c_{o'} = C_{s_{o'}}(1^n, o')$ to V^*. Then V^* decommits hc_v and reveals v to S^{V^*}. After getting v, S^{V^*} rewinds to Round 2 and randomly selects a string o and sets $t = o \oplus v$. Then for corresponding random strings s_t and s_o, S^{V^*} sends $c_t = C_{s_t}(1^n, t)$ and $c_o = C_{s_o}(1^n, o)$ to V^*. If V^* decommits hc_v correctly, that is V^* reveals v correctly again, then S^{V^*} will go to Round 4 with s_t as its auxiliary witness and complete its simulation. Here we remark that to make the simulator work in expected polynomial-time, one needs to approximately estimate the probability for that V^* decommits correctly in Round 3. Readers are referred to Ref. [7] for more details. The zero-knowledge property is ensured by the computational-hiding of the commitments used by the prover P and by the witness indistinguishability of the Dwork-Naor 2-round WI protocol.

Now, we consider Protocol 2 in comparison with Protocol 1. We remark that we do not know how to construct a zero-knowledge simulator for Protocol 2 and it is certainly impossible to construct a black-box zero-knowledge simulator for Protocol 2 (assuming NP is not included in BPP) since Protocol 2 is public-coin and Goldreich et al. have shown that only languages in BPP have a constant-round black-box zero-knowledge public-coin proof system[5]. The major difference between Protocol 1 and Protocol 2 is that in Protocol 1 the value v in Round 3 is determined by hc_v (the "determining" message) in Round 1 but in Protocol 2 there is no such "determining" message and so the value v in Round 3 of Protocol 2 may maliciously be a function of the messages sent by the honest prover in Round 2. This is just the reason that the standard rewinding technique fails in showing Protocol 2 is also zero-knowledge.

But, we argue that it is somewhat unfair for Protocol 2 to say that the malicious verifier V_2^* gains "knowledge" from P_2 (since we cannot prove Protocol 2 is zero-knowledge) while the malicious verifier V_1^* learns nothing from P_1 on the following grounds: on one hand, although the value v sent by V_2^* in Round 3 is not determined and may be maliciously dependent on the messages sent by the honest P_2 in Round 2, but up to Round 3 V_2^* has not learnt any knowledge by sending a malicious value v in Round 3 since the first three rounds constitute the first phase of FLS paradigm and we have argued that a malicious verifier cannot learn "knowledge" in the first phase of the FLS paradigm. On the other hand, based on the above discussions one may further argue that although V_2^* cannot learn "knowledge" in the first three rounds but the malicious value v may affect the final distribution of $(x, c_t, c_o, s_o, o \oplus v)$ on which the Dwork-Naor 2-round WI is finally applied in Round 4 and so V_2^* may maliciously extract "knowledge" from the messages sent by P_2 in Round 4. Here, we argue that the Blum's coin flipping over telephone protocol is applied to efface the effect caused by such a malicious v. Formally, we show in below that Protocol 2 is really reduction zero-knowledge.

Theorem 3. Assuming one-way permutations exist, Protocol 2 is a reduction zero-knowledge proof system for NP.

Proof. For each malicious verifier V_2^* of Protocol 2 we construct a malicious verifier V_1^* of Protocol 1 that sets R_{V_1} in Round 1 just to be R_{V_2} by running V_2^*. Then, for each common input $x \in L$ and its corresponding witness w, we say that the messages received by V_2^* from P_2 (which uses w as its auxiliary private input) and the messages received by V_1^* from P_1 (which uses the same w as its auxiliary private input) are computationally indistinguishable. First, in both protocols the messages sent by honest provers in Round 2 are computationally indistinguishable due to the computational-hiding of the commitment scheme used by the honest prover. Second, for the messages sent by the honest provers P_1 and P_2 in Round 4 of both protocols the difference lies in the $p(n)$ NIZK proofs. Note that the distributions of the common random strings used for these NIZK

proofs in both protocols are identical since R_{V_1} is set to be R_{V_2}. Then the difference between the $p(n)$ NIZK proofs sent by P_2 and the $p(n)$ NIZK proofs sent by P_1 is just the distributions of $o \oplus v$. However, according to the property of Blum's coin flipping over the telephone protocol, the distributions of $o \oplus v$ in both protocols are all pseudorandom. It means that the distributions of $o \oplus v$ in the two protocols are just computationally indistinguishable.

References

1 Goldwasser, S. et al. The knowledge complexity of interactive proof system. SIAM Journal on Computing, 1989, 18(1): 186.

2 Goldreich, O. et al. Definitions and properties of zero-knowledge proof systems. Journal of Cryptology, 1994, 7(1): 1.

3 Goldreich, O. et al. Proofs that yield nothing but their validity or all language in *NP* have zero-knowledge proof systems. Journal of the ACM, 1991, 38(1): 691.

4 Goldreich, O. Foundation of Cryptography-Basic Tools. London: Cambridge Press, 2001.

5 Goldreich, O. et al. On the composition of zero-knowledge proof systems. SIAM Journal on Computing, 1994, 25(1): 1.

6 Feige, U. et al. Witness indistinguishability and witness hiding protocols. In: Proceedings of the Annual ACM Symposium on Theory of Computing, New York: ACM Press, 1990, 77.

7 Goldreich, O. et al. How to construct constant-round zero-knowledge proof systems for *NP*. Journal of Cryptology, 1996, 9(2): 167.

8 Feige, U. et al. Multiple non-interactive zero-knowledge proofs under general assumptions. SIAM Journal on Computing, 1999, 29(3): 42.

9 Damgard, I. et al. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. Journal of Cryptology, 1997, 10(3): 163.

10 Blum, M. Coin flipping by telephone. In: Proceedings of IEEE Spring COMPCOM, Las Vegas: IEEE Press, 1982, 133.

11 Barak, B. et al. Resettably-sound zero-knowledge and its applications. In: Proceedings of IEEE Symposium on Foundations of Computer Science, Las Vegas: IEEE Press, 2001, 116.

12 Dwork, C. et al. Zaps and their applications. In: Proceedings of IEEE Symposium on Foundations of Computer Science, Las Vegas: IEEE Press, 2000, 43.

13 Zhao, Y. L. et al. Resettable zero-knowledge in the weak public-key model. In: Proceedings of Advances in Cryptology: EUROCRYPT 2003, Berlin: Springer-Verlag, 2003, 145.

14 Zhao, Y. L. et al. Efficient 4-round zero-knowledge proof system for *NP*. Progress in Natural Science, 2002, 12(12): 948.