Increasing the Power of the Dealer in Non-interactive Zero-Knowledge Proof Systems

Danny Gutfreund and Michael Ben-Or

Computer Science Department, Hebrew University, Jerusalem, Israel

Abstract. We introduce weaker models for non-interactive zero knowledge, in which the dealer is not restricted to deal a truly random string and may also have access to the input to the protocol (i.e. the statement to prove). We show in these models a non-interactive statistical zero-knowledge proof for every language that has (interactive) statistical zero-knowledge proof, and a computational zero-knowledge proof for every language in NP. We also show how to change the latter proof system to fit the model of non-interactive computational zero-knowledge with preprocessing to improve existing results in term of the number of bit commitments that are required for the protocol to work.

1 Introduction

When zero-knowledge proofs were first introduced by Goldwasser, Micali and Rackoff [10] it seemed that interaction played a crucial role in those proof systems. Indeed zero-knowledge was shown to exist only for languages in BPP in the most straightforward non-interactive model ([12]). Blum, Feldman and Micali [1] showed however that if we change the model slightly then zero-knowledge can be achieved for languages not known to be trivial. In their model they assumed that both prover and verifier are dealt with a truly random string called the reference string. The proof consists of one message sent from the prover to the verifier and then the verifier decides whether to accept or reject according to this message, the input and the reference string.

Non-interactive zero-knowledge proofs are not only communication efficient, they also have several applications not offered by interactive zero-knowledge proofs. They have been used in applications like digital signature schemes secure against adaptive chosen message attack ([2]), public key cryptosystems secure against chosen cipher text attack ([5], [18]), and memoryless key distributions ([2]).

Two notions of non-interactive zero-knowledge proofs have been studied: statistical zero-knowledge where the distribution over the real protocol is statistically close to the distribution induced by the simulator, and computational zero-knowledge where these two distributions are computationally indistinguishable.

Statistical zero-knowledge. The study of non-interactive statistical zero-knowledge has been recently initiated by [6]. They showed a complete promise problem for the class of languages that have non-interactive statistical zero-knowledge proofs (denoted $NISZK$). They were followed by [14] who studied

T. Okamoto (Ed.): ASIACRYPT 2000, LNCS 1976, pp. 429–442, 2000.

the relationships between the class of languages that have interactive statistical zero-knowledge proofs (denoted SZK) and $NISZK$. They showed conditions under which these two classes are equal. In particular that if $NISZK$ is closed under complementation then $NISZK = SZK$.

Computational zero-knowledge. Blum et. al ([1]) showed that every language in NP has a non-interactive computational zero-knowledge proof based on a number theoretic assumption. Since then various researchers improved this result by both relaxing the assumptions needed and making the proofs more efficient in term of the number of committed bits ([3], [4], [8], [17]). The most recent results are based on the assumption that one-way permutations exists. In [7] a weaker model was introduced called non-interactive zero-knowledge with preprocessing. They showed that every language in NP has a proof system in this model based on the assumption that one-way functions exists.

Our work. In this paper we investigate relaxed versions of the non-interactive zero-knowledge with random reference string model. Specifically, we consider models in which the dealer (we refer to the *dealer* as the entity that provides the reference string to the protocol) is not restricted to deal a string of independent unbiased coin flips to the prover and the verifier (a private coins dealer rather than a public coins one that can only publish his coins flips). Two models are considered, in the first, the reference string is a sample from a distribution that can be sampled efficiently. In the second model, this distribution can also depend on the input to the protocol.

For statistical zero-knowledge we show that the class of languages that have non-interactive statistical zero-knowledge proof system with a (polynomial-time) dealer that has access to the input equals to the class SZK. This result not only gives a new characterization of the class SZK but it also shows that if the dealer is given sufficient abilities (i.e. access to the input and the ability to compute) then every language in SZK has a communication efficient statistical zero-knowledge proof (with a reference string). In the traditional model of statistical zero-knowledge, the known generic protocols for SZK require polynomial number of communication rounds ([13]).

For computational zero-knowledge we show for every language in NP a non-interactive zero-knowledge proof system with a dealer that can make polynomial-time computations but does not have access to the input. The proof has perfect completeness, perfect soundness, and it is based on the weakest cryptographic assumption of the existence of one-way functions. We then show how this proof can be changed to fit the non-interactive zero-knowledge with preprocessing model, to improve (in term of the number of bit commitments) a protocol by [7]. We also overview some known applications of non-interactive zero-knowledge and check what additional assumptions or changes should be done (if at all) in order to replace the random reference string model with the relaxed models in these applications. In particular we argue that the digital signature scheme of [2] which is secure against adaptive chosen message attack can be done with our model.

2 Definitions

Let us first recall the definition of non-interactive zero-knowledge with a random reference string ([1]).

Definition 1. A non-interactive computational (resp. statistical) zero-knowledge proof system with a random reference string for a language L is defined by a computationally unbounded TM P (the prover), a probabilistic polynomial-time TM V (the verifier), a probabilistic polynomial-time TM S (the simulator) and a polynomial q. On an input x both P and V have access to a shared random reference string σ, where $\sigma \in_R \{0,1\}^{q(|x|)}$. The proof consists of one message sent from P to V, and then V based on x, σ and this message either accepts or rejects. The following should hold:

1. (completeness) if $x \in L$ then $Pr(V(x, \sigma, P(x, \sigma)) = \mathit{accept}) > 2/3$
2. (soundness) if $x \notin L$ then for every prover's strategy P^*, $Pr(V(x, \sigma, P^*(x, \sigma)) = \mathit{accept}) < 1/3$
3. (zero-knowledge) if $x \in L$ then the following two distributions are computationally indistinguishable (resp. have statistical difference bounded by a negligible function):
   (a) $(\sigma, P(x, \sigma))$
   (b) $S(x)$

We define now relaxed versions of the shared random reference string model. The first relaxation we introduce is that the shared reference string need not be truly random, we only require that it can be sampled in polynomial-time.

Definition 2. A non-interactive computational (resp. statistical) zero-knowledge proof system with a protocol-dependent reference string for a language L is defined by P, V, S, and q as above, and the reference string σ is $f(r)$, where f is a polynomial-time computable function, and $r \in_R \{0,1\}^{q(|x|)}$.

The second relaxation is that the shared reference string can not only be non-uniformly distributed, but it can also depend on the input.

Definition 3. A non-interactive computational (resp. statistical) zero-knowledge proof system with an input-dependent reference string for a language L is defined by P, V, S, and q as above, and the reference string σ is $f(x, r)$, where f is a polynomial-time computable function, x is the input to the protocol, and $r \in_R \{0,1\}^{q(|x|)}$.

Note that non-interactive zero-knowledge under all the definitions is closed under parallel repetitions, therefore the error bound can be brought down to be exponentially small in the length of the input.
We denote by $NICZK$ (resp. $NISZK$), $Protocol-Dependent\ NICZK$ (resp. $Protocol{-}Dependent\ NISZK$), and $Input{-}Dependent\ NICZK$ (resp. $Input{-}Dependent\ NISZK$) the class of languages possessing a non-interactive computational (resp. statistical) zero-knowledge proof system with a shared random, protocol-dependent and input-dependent string respectively. SZK is the class of all the languages that have statistical-zero knowledge proof system as defined by [10].

3 Statistical Zero-Knowledge

In this section we show that if we relax the model of non-interactive proof systems then every language in SZK has a proof system in this relaxed model:

Theorem 4. $SZK = Input - Dependent\ NISZK$.

3.1 Motivation

It is an interesting question to understand how much more can be proven in non-interactive statistical zero-knowledge as we gradually increase the power of the dealer. By referring to the power of the dealer we mean the type of computations he can do, and does he have access to the input (the statement to prove). Clearly, if the dealer is computationally unbounded and has access to the input then everything computable can be proven non-interactively with perfect zero-knowledge, by using the dealer as an unbounded trusted prover that tells the verifier whether the statement is correct or not. What happens if we do not give the dealer so much power? We can divide the languages into classes according to the power of the dealer in the non-interactive statistical-zero knowledge proof systems for them. We have the following hierarchy of classes, each one containing the class above it:

- No dealer: this class equals to BPP ([12]).
- The dealer can just toss coins (a public coins dealer): this is the class $NISZK$, languages not known to be in BPP were shown to be in this class ([6], [14]).
- The dealer can toss coins and make polynomial-time computations (a private coins polynomial-time dealer): this is the class $protocol - dependent\ NISZK$.
- A private coins polynomial-time dealer with access to the input: this is the class $input - dependent\ NISZK$ which equals to SZK (Theorem 4).
- A private coins unbounded dealer with access to the input: everything computable is in this class.

By showing the exact location of SZK in this hierarchy we get a strong connection between the question of how much interaction is needed for statistical zero-knowledge and how much power the dealer must have in non-interactive statistical zero-knowledge. Better understanding of this hierarchy can shed light on the SZK vs. $NISZK$ question ([14]).

3.2 Separating Distributions

An important notion in the proof of theorem 4 will be the statistical difference between two distributions. Let us first define the statistical difference and some notations concerning it.

Definition 5. Let X and Y be two distributions (or random variables) over a discrete space D. The statistical difference between X and Y, denoted as $||X - Y||$ is:
$||X - Y|| = MAX_{S \subseteq D} |Pr(X \in S) - Pr(Y \in S)|$

Through out this paper we consider distributions with "succinct" description, i.e. distributions produced by circuits (with multiple output gates) when feeding them a uniformly chosen input. We write C when we refer both to the circuit itself and to the distribution it induces. We will use the notation $x \leftarrow C$ to denote that x is a sample taken from the distribution C, i.e. the output of the circuit C when feeding it a uniformly chosen input.

In our proof of theorem 4 we will use the langauge STATISTICAL-DIFFERENCE shown to be complete for the class SZK ([21]), and a "separated" version of this language.

Definition 6. STATISTICAL-DIFFERENCE (SD) is the following promise problem:
$SD_Y = \{(D_0, D_1) : ||D_0 - D_1|| > 2/3\}$
$SD_N = \{(D_0, D_1) : ||D_0 - D_1|| < 1/3\}$
Where (D_0, D_1) is a pair of distributions with "succinct" description.

Definition 7. "Separated" STATISTICAL-DIFFERENCE (SD') is the following promise problem:
$SD'_Y = \{(D_0, D_1) : \text{for } x \leftarrow D_0,\ Pr(x \in \mathit{Range}(D_1)) < f(n)\}$
$SD'_N = \{(D_0, D_1) : ||D_0 - D_1|| < f(n)\}$
Where $n = |D_0, D_1|$ and f is a negligible function.
In other words, a pair of circuits is in SD'_Y if the probability that a sample taken from the first circuit is in the range of the second is negligible, and a pair of circuits is in SD'_N if the distributions induced by the circuits have a negligible statistical difference. Note that $SD'_Y \subseteq SD_Y$ and $SD'_N \subseteq SD_N$.

Our main tool will be the following lemma:

Lemma 8. *SD reduces to $\overline{SD'}$.*
Specifically, given a pair of circuits (D_0, D_1) there is a polynomial-time computable function that maps them to a new pair of circuits (C_0, C_1) s.t:
$||D_0 - D_1|| > 2/3 \longrightarrow ||C_0 - C_1|| < f(n)$
$||D_0 - D_1|| < 1/3 \longrightarrow \text{for } x \leftarrow C_0,\ Pr(x \in \mathit{Range}(C_1)) < f(n)$
Where f is a negligible function.

Sahai and Vadhan showed a reduction from SD to $\overline{SD}$ ([22]). The same reduction accomplishes Lemma 8, although it is implicit in their proof. We do not give the proof here and refer the reader to [22].

3.3 Proof of the Main Theorem

Claim 9. *$SZK \subseteq Input - Dependent\ NISZK$*

Proof. SD is a complete promise problem for the class SZK ([21]) and SD reduces to $\overline{SD'}$ (Lemma 8), therefore it is enough to show an input-dependent non-interactive statistical zero-knowledge proof system for the language $\overline{SD'}$.

Let l be the number of input gates of D_0 and D_1 (w.l.o.g they have the same number of input gates).
The proof system:
Common input: (D_0, D_1)
Shared reference string: $\sigma = D_0(r)$, where $r \in_R \{0,1\}^l$
The protocol:

1. P sends $r' \in \{0,1\}^l$.
2. V accepts if and only if $D_1(r') = \sigma$

Completeness: Recall that $(D_0, D_1) \in \overline{SD'}_Y$ means that $||D_0 - D_1|| < f(n)$ (f is a negligible function). Let k be the number of output gates of D_0 and D_1 (again w.l.o.g this number is the same for D_0 and D_1). Then by the definition of statistical difference, for $x \leftarrow D_0$, $y \leftarrow D_1$, and for every $S \subseteq \{0,1\}^k$:
$|Pr(x \in S) - Pr(y \in S)| < f(n)$
Let $T = Range(D_0) \setminus Range(D_1)$, $x \leftarrow D_0$ and $y \leftarrow D_1$. Then the following holds:
$f(n) > |Pr(x \in T) - Pr(y \in T)| = Pr(x \in T)$
In other words the probability that a sample taken from D_0 is not in the range of D_1 is at most $f(n)$. Therefore the probability that σ is in the range of D_1 is at least $1 - f(n)$ and with this probability (over r) the (computationally unbounded) prover will be able to find r' s.t $D_1(r') = \sigma$.
Soundness: $(D_0, D_1) \in \overline{SD'}_N$ means that the probability that a sample taken from D_0 is in the range of D_1 is at most $f(n)$, so the probability that σ is not in the range of D_1 is at least $1 - f(n)$. In this case there is no r' s.t $D_1(r') = \sigma$ and the prover will fail.
Simulation:
S: Choose $r \in_R \{0,1\}^l$, output $(D_1(r), r)$.
If $(D_0, D_1) \in \overline{SD'}_Y$ then the distributions D_0, D_1 has statistical difference at most $f(n)$ therefore σ and the first message of the simulator has statistical difference $f(n)$ at the most. The second message both in the protocol and the simulator is determined by the first message (the shared reference string) to be a random input (to C_1) in the preimage of the first message. So for a given first message, the second message has the same distribution in the protocol and the simulator. □

Claim 10. *$SZK \supseteq Input - Dependent\ NISZK$*

Proof. : We will show a reduction from any language in $Input - Dependent\ NISZK$ to the language $\overline{SD}$. As SD is complete for SZK, and SZK is closed under complementation [19] this will suffice to prove the claim. Let (P, V) be an input-dependent $NISZK$ protocol for a language L with exponentially small error bound. Let S be the simulator for the protocol, f the polynomial-time computable function that produces the reference string and q the polynomial (in the input length) that defines the length of the input to f. Define μ to be the negligible function bounding the statistical difference between the outputs of (P, V) and S. For an input x, define the following pair of distributions:

D_0: choose $r \in_R \{0,1\}^{q(|x|)}$, output $f(x,r)$.
D_1: run the simulator S on x to obtain (σ, p), if $V(x,\sigma,p) = 'accept'$ output σ, otherwise output $0^{q(|x|)}$ (here $0^{q(|x|)}$ is canonic for a string outside the range of f).

First we show that if $x \in L$ then $||D_0 - D_1|| < 1/3$. Let $n = |x|$, the statistical difference between the first message of the simulator and the real reference string is bounded by $\mu(n)$. Also, $Pr((P,V)(x) = 'accept') > 1 - 2^{-n}$, therefore with probability $1 - 2^{-n} - \mu(n)$ D_1 will output the first message of the simulator and thus $||D_0 - D_1|| < 2\mu(n) + 2^{-n}$.
Next we show that if $x \notin L$ then $||D_0 - D_1|| > 2/3$. Define:
$T = \{\sigma : \exists r \in \{0,1\}^{q(n)}$ and $\exists p$ s.t $f(x,r) = \sigma$ and $V(x,\sigma,p) = 'accept'\}$
That is, T is the set of all reference strings for which there exist a proof that will convince V. Since the soundness error is bounded by 2^{-n}, $Pr_{r \in \{0,1\}^{q(n)}}(f(x,r) \in T) < 2^{-n}$. Since D_1 only outputs $\sigma \in T$ or $0^{q(n)}$, and D_0 outputs a real reference string, $||D_0 - D_1|| > 1 - 2^{-n}$. □

4 Computational Zero-Knowledge

4.1 A Protocol-Dependent *NICZK* Proof for *NP*

In this section we show a protocol-dependent non-interactive CZK proof system for every language in NP with perfect completeness and perfect soundness. It is based on the assumption that one-way functions exists. The proof system is for the NP-complete language $3 - COL$ of all the 3-colorable graphs.
In our protocol we will make use of *characters*. *Characters* were used by [17] in their non-interactive zero-knowledge proof for NP in the random reference string model. A *character* is an object that can have one of four possible values: 1, 2, 3 or WC (wild card). The value of a *character* is unknown to the verifier unless the prover reveals it for him. It can be revealed according to the following rules: if the value is 1, 2, or 3 then it can only be revealed to this value. If the value is WC then it can either be revealed to 1, 2 or 3 (what ever the prover chooses).
Characters can be implemented in the following way: a *character* will be the commitment on a triplet of bits that can have one of the following values: 001 (the *character* 1), 010 (the *character* 2), 100 (the *character* 3), or 111 (WC). The security of the bit-commitment scheme ensures that the value of a *character* is unknown to the polynomial-time verifier, unless the (computationally unbounded) prover reveals it for him. To reveal the value of a *character*, the prover opens one bit from the triplet and the verifier checks that the value of this bit is 1. The location of the revealed bit determines the value of the *character*. Clearly, $\{1, 2, 3\}$ *characters* can only be revealed to their real value and WC can be revealed to any value.
Next we define a test to check whether two nodes of an edge in the graph are colored in different colors. The test will be conducted in the following way: with each node we associate a triplet of *characters*. Two of them are WC and one

is a non-WC *character*, thus we have two non-WC *characters* associated with the edge (one in each node). We make sure that these two *characters* have different values (that is, the dealer prepares them in this manner). We define the color of a node to be the position of the non-WC *character* within the triplet of *characters* associated with it. We say that two such triplets of *characters* are consistent if they are representing the same color. By reordering the *characters* in each triplet, the prover can determine the colors of the two nodes. In order to prove that the two nodes were given different colors, the prover will reveal the two triplets of *characters*, and the verifier will accept if and only if the two triplets are the same permutation of $\{1, 2, 3\}$. We call this test the inconsistency test. We prove now some properties of this test.

Claim 11 (completeness). *If the two nodes were given different colors then the prover will always pass the test.*

Proof. The fact that the two nodes have two different colors means that the two non-WC *characters* are in different positions in the triplets associated with the nodes. That is, if we align the two triplets, against each non-WC *character* in one triplet there will be a WC *character* in the other, and in one position there will be a WC against a WC. The prover will reveal each WC *character* which is aligned against a non-WC *character* to the value of this *character*. The two WC *characters* which are aligned against each other will be revealed to the value which is not used in the other positions in the triplets (recall that the two non-WC *characters* have different values, therefore this value is determined). So in each position the same value will be revealed in the two triplets and each value in $\{1, 2, 3\}$ will be revealed exactly once. □

Claim 12 (soundness). *If the two nodes were given the same color then the prover will always fail the test.*

Proof. The fact that the two nodes have the same color means that if we align the two triplets, the two non-WC *characters* will be aligned against each other. Since they have two different values and the prover can not reveal them to any other value, the verifier will always see different values in this position in the two triplets and will reject the test. □

Claim 13 ("zero-knowledge"). *If the values of the non-WC characters are chosen uniformly (with the restriction that they have different values) then if the nodes were given different colors, the triplets will be revealed to a random permutation of* $\{1, 2, 3\}$ *(with probability 1/6 for each permutation).*

Proof. Assume w.l.o.g that the first node receives the color 1 (the non-WC *character* is in the first position), the second receives the color 2, and we first choose the value of the first *character* with probability 1/3 for each possible value and then we choose the value of the second *character* with probability

1/2 for each remaining possible value. Clearly each pair can be chosen with probability 1/6. After the values of the non-WC *characters* are chosen, the permutation is determined. This is because each WC *character* which is aligned against a non-WC *character* must be revealed to its value, and the value of the WC *characters* which are aligned against each other is determined to be the value which was not chosen for the two non-WC *characters*. Furthermore, each choice of values for the non-WC *characters* defines a different permutation. Therefore each permutation of $\{1, 2, 3\}$ can be revealed with probability 1/6. □

We can now present a protocol-dependent non-interactive computational zero-knowledge proof system for the language $3 - COL$:

The proof system:

Input: A graph $G = (V, E)$ (denote $n = |V|$).

Reference string: An independently and randomly chosen inconsistency tests for each one of the edges of the complete graph K_n[1], where for each node all the triplets of *characters* associated with it are consistent. In other words, with each node we associate a $(n-1) \times 3$ matrix of *characters*, where two columns contains only WC *characters* and one contains only non-WC *characters*. For each matrix the position of the non-WC column is chosen randomly and independently.

The proof:

1. P's proof is divided into two stages:
   (a) For each node in V, P swaps two columns of the matrix associated with it to create a new matrix.
   (b) For each edge in E, P performs the inconsistency test associated with it (with the new matrices from stage 1(a)).
2. V accepts if and only if P passes all the inconsistency tests.

Proof of correctness:

Completeness: Let γ be a 3-coloring of G. For each node there is a $(n-1) \times 3$ matrix of *characters* associated with it. With each such matrix we associate a color according to the position of the non-WC column. In stage 1(a), for each node, if the matrix does not represent the color of the node according to γ, P swaps the non-WC column with a WC one so that the new matrix will represent the right color. If the matrix does represent the right color, P swaps the two WC columns. Since γ is a legal 3-coloring of G, after stage 1(a) all the matrices of adjacent nodes are inconsistent and P will always pass all the inconsistency tests (claim 11).

Soundness: The fact that there is no legal 3-coloring of G means that it is impossible to bring the matrices to a state where for every two adjacent nodes the matrices associated with them are inconsistent (otherwise the state of the matrices will define a valid 3-coloring). Therefore for at least one edge in E, P will always fail the inconsistency test (claim 12).

[1] Note that in the protocol-dependent model the dealer does not have access to the input, therefore he must prepare an inconsistency test for every possible edge.

Simulation:

1. (simulation of the reference string) Associate with each node a $(n-1) \times 3$ matrix of WC *characters*. This is done by committing on $9n(n-1)$ bits of value 1.
2. (simulation of stage 1(a)) For each node in V, choose randomly two columns in the matrix associated with it and swap them.
3. (simulation of stage 1(b)) For each $e \in E$, choose a random permutation of $\{1,2,3\}$ and reveal the two triplets of *characters* associated with e (i.e. the simulation of the inconsistency test for e) to be this permutation.

The security property of the bit-commitment scheme ensures that the real reference string and the simulated one are computationally indistinguishable.
Let γ be a 3-coloring of G. Let $c \in \{1,2,3\}$ be the color of $v \in V$ according to γ. Let A_v be the matrix associated with v in the real proof. The position of the non-WC column in A_v is chosen randomly. If A_v represents the color c, and this happens with probability 1/3, then the two WC columns (the columns that do not represent c) will be swapped in stage 1(a). Otherwise A_v represents one of the other two colors, with probability 1/3 for each one, and then the column that represents c will be swapped with the non-WC column. So in stage 1(a) the two columns that will be swapped in A_v are random and since the position of the non-WC column was chosen independently for all the matrices this is also the case for all the matrices together.
After all the matrices were brought in stage 1(a) to a state where every two adjacent nodes have inconsistent matrices, in every inconsistency test in stage 1(b) a random permutation of $\{1,2,3\}$ will be revealed (claim 13). Since the values for the inconsistency tests were chosen independently, all the permutations that will be revealed in stage 1(b) will be random and independent. □

Remark: The proof system presented above requires that the reference string will contain $O(n^2)$ bit commitments. The reason for this is that the structure of the graph is unknown in advance and the dealer must prepare an inconsistency test for each possible edge (i.e. for every edge in the complete graph with n nodes). However, if we use a more "structured" NP-complete problem such as the coloring problem on a wrapped de Bruijn graph ([20]), where only $O(nlogn)$ local tests are needed we can improve the number of bit-commitments to $O(nlogn)$. A proof system for this problem will be presented in the full version of this paper.

4.2 Non-interactive Proofs with Preprocessing

The notion of non-interactive zero-knowledge proofs with preprocessing was first introduced by [7]. In their model, the proof system is divided into two stages: first, before there is a statement to prove, the prover and the verifier execute an interactive protocol which ends with both of them agreeing on a common reference string. Then, when there is a statement to prove, the prover sends a single message to the verifier, and the latter decides whether to accept or reject according to the input, the prover's message, and the reference string. The original proof system required that the reference string will contain $O(n^3)$ bit commitments to prove that a 3-CNF formula of length n is satisfiable, and it was based on the assumption that one-way functions exist. [16] gave a non-interactive zero-knowledge proof system with preprocessing that requires less bit commitments and also has the property that multiple statements can be proven based on a single reference string. However, this came at the expense of a stronger cryptographic assumption, namely oblivious transfer.

The protocol-dependent model seems to be close to the non-interactive with preprocessing model. In both cases the prover's message is based on a non-random reference string. In the protocol-dependent model, a trusted (polynomial-time) dealer provides the reference string and in the preprocessing model the prover and the verifier agree on it in advance. Indeed, the proof system we introduced in the previous section can be easily changed to work in the preprocessing model. To see that, notice that the following language is in NP:

$L = \{(s, 1^n) : s \in \{0,1\}^{poly(n)}$, s is a valid reference string for the protocol-dependent proof system for $3 - COL$ (of a graph with n nodes)$\}$

The proof system (in the preprocessing model) will be:

Stage 1 (interactive): The prover chooses a reference string in the same way the dealer chose it in the protocol-dependent model. Then the prover proves interactively and in zero-knowledge that this string is in L. Since $L \in NP$, such a proof exists under the assumption that one-way functions exists ([11]).

Stage 2 (non-interactive): Continue as in the protocol-dependent model.

This proof system improves [7] as it is based on the same assumption and it requires that the reference string will contain only $O(nlogn)$ bit commitments to prove a statement of size n.

4.3 Efficient Provers and Applications

Note that the protocol in section 4.1 requires that the prover will have computational power sufficient to reverse the bit commitments of the dealer, i.e. to compute the inverse of a one way function. For the protocol to be applicable we would like it to work for an efficient prover, that is, a polynomial-time prover with an auxiliary input containing a witness for the NP statement. For this we will have to change the cryptographic assumption, and instead of the bit commitment to be based on any one way function we would like it to be based on a family of unapproximable trapdoor predicates ([9]).

The proof system requires now an additional preliminary step: the prover sends to the dealer a predicate for which he knows the trapdoor to, and the dealer uses it for his bit commitments. Now the prover can reverse the bit commitments, and the protocol continues as before.

Non-interactive zero-knowledge proofs were shown to be useful in many applications. For applications where the prover is also the dealer (e.g. digital signature schemes secure against adaptive chosen message attack [2] and memoryless key distribution [2]) the protocol-dependent model will still work. This is because the prover will commit on the bits and therefore will be able to open them at a later stage. It is in the prover's best interest that the reference string will be correctly prepared. For public key cryptosystems secure against chosen ciphertext attack ([5], [18]) this is not the situation. The key-generator can not generate the public key (that includes the reference string) without knowing which predicate to use for the bit-commitments (the one that the prover knows the trapdoor for). Therefore the prover must notify it in advance which predicate to use. This is a major drawback because for each prover we will need a different public key.

5 Concluding Remarks

We showed that if we assume that the (polynomial-time) dealer has access to the input to the protocol as well as private coins then every language that has a statistical zero-knowledge proof system also has a non-interactive statistical zero-knowledge proof system. It would be very interesting to understand whether these assumptions are required for SZK to be done non-interactively.

In the Computational zero-knowledge setting, we showed an efficient (in term of the number of bit-commitments) $protocol - dependent\ NICZK$ protocol for every language in NP. The protocol is based on the assumption of the existence of one-way functions (for unbounded provers), or on the assumption of the existence of a family of unapproximable trapdoor predicates (for efficient provers). We also showed how this model can replace the traditional model in some applications such as secure digital signatures. For secure public-key cryptosystems the use of our model raises problems, namely that a public-key is generated for a particular prover (sender) and can not be used by anyone. If we could avoid this problem, we would get a very interesting result, that the use of non-interactive zero-knowledge proofs in order to get a public-key cryptosystem that is secure against chosen cipher text attack does not impose a stronger cryptographic assumption than the public-key encryption itself. This is due to [9] who showed that the existence of a (semantically secure) public-key cryptosystem is equivalent to the existence of a family of unapproximable trapdoor predicates. Current use of non-interactive zero-knowledge in the random reference string model is based on the stronger assumption that there is a family of trapdoor permutations.

6 Acknowledgements

We would like to thank Salil Vadhan and Oded Goldreich for valuable discussions about zero-knowledge. We would also want to thank them as well as Yonatan Bilu and anonymous referees for helpful comments on the paper. The first author also wishes to express his gratitude to Avi Wigderson who directed him to this topic of research.

This research was supported in part by the Leibniz Center, a US-Israel Binational research grant, and an EU Information Technologies grant (IST-FP5).

References

1. Manuel Blum, Paul Feldman, Silvio Micali: Non-interactive zero-knowledge and its applications (extended abstract). In Proceedings of the 20th ACM Symposium on the Theory of Computing, 103-112, Chicago, Illinois, 2-4 May 1988.
2. Mihir Bellare, Shafi Goldwasser: New paradigms for digital signatures and message authentication based on non-interactive zero-knowledge proofs. In Advances in Cryptology CRYPTO '89, LNCS volume 435, pages 194-211. Springer-Verlag, 1990.
3. Manual Blum, Alfredo De Santis, Silvio Micali, Guiseppe Persiano: Non-interactive zero-knowledge. SIAM Journal on Computing, 20(6):1084-1118, December 1991.
4. Ivan Damgard: Non-interactive circuit-based proofs and non-interactive perfect zero knowledge with preprocessing. In Advances in Cryptology EUROCRYPT '92. LNCS volume 658. Springer Verlag. 1992.
5. Danny Dolev, Cynthia Dwork, Moni Naor: Non-malleable cryptography (extended abstract). In Proceedings of the 23rd Annual ACM Symposium on the Theory of Computing, pages 542-552, New Orleans, Louisiana, 6-8 May, 1991.
6. Alfredo De Santis, Giovanni Di Crescenzo, Giuseppe Persiano, Moti Yung. Image density is complete for non-interactive SZK. In Automata, Languages and Programming, 25th International coloquium , LNCS, Springer-Verlag. 1998.
7. Alfredo De Santis, Silvio Micali, Giuseppe Persiano: Non-interactive zero-knowledge proof-systems with preprocessing. In Advances in Cryptology, CRYPTO 88, LNCS volume 403, Springer Verlag.
8. Uriel Feige, Dror Lapidot, Adi Shamir: Multiple non-interactive zero-knowledge proofs based on a single random string. In Proceedings of the 31st Annual Symposium on Foundations of Computer Science, pages 308-317, 1990.
9. Shafi Goldwasser, Silvio Micali: Probabilistic encryption. Journal of Computer and System Sciences, Vol. 28, pages: 270-299, April 1984.
10. Shafi Goldwasser, Silvio Micali, Charles Rackoff: The knowledge complexity of interactive proof systems. SIAM journal of computing, 18(1): 186-208, 1989.
11. Oded Goldreich, Silvio Micali, Avi Wigderson: Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. Journal of the association for computing machinery, 38(1):691-729, 1991.
12. Oded Goldreich, Yair Oren: Definitions and properties of zero-knowledge proof systems. Journal of chryptology, 7(1):1-32, Winter 1994.
13. Oded Goldreich, Amit Sahai, Salil Vadhan: Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In Proceedings of the 30th Annual ACM Symposium on the Theory of Computing, pages 399-408,1998.
14. Oded Goldreich, Amit Sahai, Salil Vadhan: Can statistical zero-knowledge be made non-interactive? or on the relationship of SZK and NISZK. In Michael Wiener, editor, Advances in Cryptology CRYPTO '99, LNCS, Springer-Verlag, 1999.
15. Russell Impaliazzo, Leonid A. Levin, Michael Luby. Pseudo-random generation from one-way functions (extended abstract). In Proceedings of the 21st Annual ACM Symposium on the Theory of Computing, pages 12-24, 1989.
16. Joe Kilian, Silvio Micali, Rafail Ostrovsky: Minimum resource zero-knowledge proofs. In Proceedings of the 30th annual Symposium on Foundations of Computer Science, pages 474-479, 1989.

17. Joe Kilian, Erez Petrank: An efficient non-interactive zero-knowledge proof system for NP with general assumptions. Journal of Cryptology, 11(1):1-27, Winter 1998.
18. Moni Naor, Moti Yung: Public-key cryptosystems provably secure against chosen ciphertext attacks. In Proceedings of the 22nd Annual ACM Symposium on the Theory of Computing, pages 427-437, Baltimore, Maryland, 14-16 May 1990.
19. Tatsuaki Okamoto: On relationships between statistical zero-knowledge proofs. In Proceedings of the 28th Annual Symposium on the Theory of Computing, 1996.
20. Dan Spielman: Computationally efficient error-correcting codes and holographic proofs. PhD thesis, 1995.
21. Amit Sahai and Salil Vadhan: A complete promise problem for statistical zero-knowledge. In Proceedings of the 38th Annual Symposium on Foundations of Computer Science, pages 448-457, 1997.
22. Amit Sahai, Salil Vadhan: Manipulating statistical difference. In Panos Pardalos, Sanguthevar Rajaseekaran, and Jose Rolim, editors, Proceedings of the DIMACS Workshop on Randomization Methods in Algorithm Design, Princeton, NJ, 1998.