Zero Knowledge Proofs of Knowledge in Two Rounds

U. Feige, A. Shamir
Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel

Abstract

We construct constant round ZKIPs for any NP language, under the sole assumption that oneway functions exist. Under the stronger Certified Discrete Log assumption, our construction yields perfect zero knowledge protocols. Our protocols rely on two novel ideas: One for constructing commitment schemes, the other for constructing subprotocols which are not known to be zero knowledge, yet can be proven not to reveal useful information.

1 Introduction

The concept of zero knowledge interactive proofs (ZKIPs) was introduced by Goldwasser, Micali and Rackoff (see [18] for definitions). Goldreich, Micali and Wigderson [17] show that under the assumption that secure bit commitment schemes exist, any NP language has a ZKIP system. In a remark they hint how to modify their protocols and obtain a 4-move ZKIP (a move is a message from verifier V to prover P, or from P to V, and a round is two consecutive moves). Alas, proving the correctness of the construction implied by this hint met unexpected technical difficulties, and was recently characterized by Goldreich (in a private communication) as an open problem.

In this paper we study the model where both P and V are polynomial time with auxiliary input (as studied in [5], [12] and others). We present a construction of constant round ZKIPs for any NP language, which relies on the polynomiality of P, and differs from the suggested construction in [17] (such ZKIPs are termed "computationally sound" by Goldreich, and "arguments" by Brassard and Crépeau). The protocol is a proof of knowledge, and not just a proof of assertion. Its success is evidence that P "knows" a witness to the given NP statement. (See [12] or [23] for definitions and discussion of the importance of proofs of knowledge). Our protocol offers different levels of security, depending on the strength of the cryptographic assumption we make:

G. Brassard (Ed.): Advances in Cryptology - CRYPTO '89, LNCS 435, pp. 526-544, 1990.

1. By relying only on the assumption that oneway functions exist, our protocol takes 5 moves and is computational zero knowledge. This assumption is used in two ways: As a special case of the more general *Invulnerable Generator Assumption* (IGA), which in essence means that it is possible to generate hard, certified elements of some NP sets, and in order to construct secure commitment schemes. The construction of commitment schemes from any oneway function follows from recent results in [20] and [21], and requires a preliminary move).

2. By relying on the assumption that one-to-one oneway functions exist, our protocol takes only 4 moves, and is still computational zero knowledge. The fact that the oneway functions are one-to-one enables the construction of commitment schemes without the preliminary move, thus saving one move in the protocol.

3. By relying on the *certified discrete log assumption*, our protocol takes 4 moves, and is *perfect* zero knowledge.

The 4 moves protocols are optimal among all the zero knowledge protocols in which the zero knowledge property is proved by resettable blackbox simulation. This follows from [15], where it is proved that only languages in BPP have 3 move interactive proofs which can be proven zero knowledge by such a simulation.

Other general schemes for constant round ZKIPs (though not 2-rounds) have been independently discovered. Brassard, Crépeau and Yung [6] construct a 6-move protocol which is perfect zero knowledge (in the model where the prover is polynomial time). Their protocol relies on the Certified Discrete Log Assumption (or alternatively, on a generalization of this assumption). Our construction achieves 4-move perfect zero knowledge protocols, in the same model and under the same assumption, and reduces the number of bits communicated by a factor of $O(n)$. Goldreich and Kahn [14] announced a 5-move bounded round ZKIP based on claw-free pairs of functions. The cryptographic assumptions made in order to prove the correctness of their protocols are stronger than the assumption we make.

The reader should not confuse the issue of constant round ZKIPs with that of noninteractive (one move) zero knowledge protocols [4], as noninteractive zero knowledge assumes the existence of a random string agreed upon by P and V. Our protocols start from scratch. Likewise, the reader should not confuse our perfect zero knowledge protocols with Fortnow's [13] "impossibility" result, as Fortnow proves his result in a model where the provers are not limited to polynomial time.

We want to highlight two aspects of our protocol:

1. One of the subprotocols we use is not known to be zero knowledge, yet provably does not reveal any useful information. This is an application of a new concept of *witness hiding* [11], which can replace the standard concept of zero knowledge in many cryptographic applications.

2. We show a general technique for constructing commitment schemes out of ZKIPs. This is a dual to the well known technique of constructing ZKIPs out of commitment schemes. (This result was discovered independently by Damgard [9]).

2 Notation and Definitions

For a discussion on the following definitions, see [18] (interactive proofs and zero knowledge),[12] and [23] (proofs of knowledge).

Our model of computation is the probabilistic polynomial time interactive Turing machine (both for provers P and for verifiers V). The common input is denoted by x, and its length is denoted by $|x| = n$. Each machine has an auxiliary input tape. P's auxiliary input is denoted by w. V's auxiliary input is denoted by y, and for truthful V y is empty. $\nu(n)$ denotes a function vanishing faster than the inverse of any polynomial. Formally:

$$\forall k \; \exists N \; s.t. \; \forall n > N \;\; \nu(n) < \frac{1}{n^k}$$

Negligible probability is probability behaving as $\nu(n)$. *Overwhelming* probability is probability behaving as $1 - \nu(n)$.

$A(x)$ denotes the output of algorithm A on input x. This may be a random variable, if A is allowed to toss coins. $V_P(x)$ denotes V's output on input x, after participating in an interactive proof (P, V). $M(x, A)$ (where A may be either P or V) denotes algorithm M's output on input x, where M may use algorithm A as a subroutine (blackbox). Each call M makes to A counts as one computation step for M.

Definition 2.1: Let L be an NP language accepted by the polynomial time nondeterministic Turing machine M_L. A *computation path* is a sequence of nondeterministic choices that M_L makes. The set of M_L's accepting computation paths on input $x \in L$ is called the *witness set* of x, and is denoted by $w(x)$. $\diamond$

Definition 2.2: An *interactive proof of knowledge* system for NP language L is a pair of algorithms (P, V) satisfying:

1. Completeness: For any $x \in L$, for any $w \in w(x)$, $V_{P(x,w)}(x)$ accepts with overwhelming probability. Formally:

$$\forall x \in L \; \forall w \in w(x) \; Prob(V_{P(x,w)}(x) accepts) > 1 - \nu(n)$$

   The probability is taken over the coin tosses of P and V.

2. Soundness: For any x, for any P', P' can convince V to accept only if he actually "knows" a witness for $x \in L$. Expected polynomial time knowledge extractor M is used in order to demonstrate P's ability to compute a witness. Formally:

$$\exists M \; \forall P' \; \forall x \; \forall w'$$

$$(Prob(V_{P'(x,w')}(x) accepts) - Prob(M(x, P'(x,w')) \in w(x))) < \nu(n)$$

The probability is taken over the coin tosses of V and M. P' is assumed not to toss coins, since his favorable coin tosses can be incorporated into the auxiliary input w'. The knowledge extractor M is allowed to use P' as a subroutine.

◇

Remark: For $x \notin L$, the probability that V accepts is negligible, since the witness set of such inputs is empty. ◇

We recall the definition for indistinguishability of ensembles which is needed for the subsequent definition of zero knowledge (and later will be used in the definition of witness indistinguishability).

Definition 2.3: Let I be an infinite set of strings, and let E_1 and E_2 be two probability ensembles. (For any $x \in I$, $E_1(x)$ and $E_2(x)$ are random variables). For any algorithm D denote by $P_1^D(x)$ ($P_2^D(x)$ respectively) the probability that D outputs 1 on input x and an element chosen according to probability distribution $E_1(x)$ ($E_2(x)$). The ensembles E_1 and E_2 are *polynomially indistinguishable* if for any nonuniform polynomial time distinguisher D,

$$|P_1^D(x) - P_2^D(x)| < \nu(n)$$

◇

Instead of using the term "polynomial indistinguishability" we shall just use "indistinguishability". Two variants on the definition are the following: If D is not restricted to polynomial time, the ensembles are termed *statistically* indistinguishable. If furthermore, the condition required is $P_1^D(x) = P_2^D(x)$, the ensembles are termed *perfectly* indistinguishable.

Definition 2.4: Proof system (P, V) is *zero knowledge* over L if there exists a simulator M which runs in expected polynomial time, such that for any probabilistic polynomial time V', for any input $x \in L$, associated witness w and auxiliary input to V' y, the two ensembles $V'_{P(x,w)}(x, y)$ and $M(x, V'(x, y))$ are polynomially indistinguishable. M is allowed to use V' as a subroutine. ◇

The concept defined above is also termed *computational* zero knowledge. *Statistical* (*perfect* respectively) zero knowledge is defined with *polynomial* indistinguishability replaced by *statistical* (*perfect*) indistinguishability.

Definition 2.5: A *move* of an interactive proof is a messages sent by one of the participants. Two moves (a message sent by V followed by a message sent by P) are called a *round*. ◇

In our protocols we make use of two cryptographic assumptions. Before stating the assumptions, we borrow (and modify) terminology from [1], which discusses the generation of hard, certified elements of NP sets.

Definition 2.6: Let G be a random polynomial time generating algorithm producing on input 1^n instances $(x, w) \in S$ of length n, where $S = \{(x, w)\}$ is a set recognizable in polynomial time. G is an *almost everywhere $\nu(n)$-invulnerable generator* (or just *invulnerable generator*) if for any polynomial time nonuniform adversary algorithm A, $Prob((x, A(x)) \in S) < \nu(n)$. The probability is taken over the coin tosses of G (A is assumed not to toss coins, as the most advantageous coin tosses can be incorporated into his nonuniform description). $\diamond$.

We stress that G must be invulnerable to *nonuniform* adversaries, unlike the case in [1]. Our first assumption, which is used for constructing computational zero knowledge protocols, relates to the existence of invulnerable generators.

Invulnerable Generator Assumption (IGA): There exists an invulnerable generator.

IGA is a very weak assumption, and follows from the more widely used assumption: The existence of oneway functions secure w.r.t. nonuniform algorithms.

Definition 2.7: Let G be a random polynomial time generating algorithm producing on input 1^n instances x of length n, and let f be a length preserving function whose domain is the range of G. f is *oneway* if it can be computed in polynomial time, but there is no nonuniform polynomial time algorithm which inverts f with nonnegligible probability, where the probability is taken over the distribution generated by G. $\diamond$

The next assumption is used only for our perfect zero knowledge protocols. The same assumption is used in [6] in the construction of their perfect zero knowledge protocols.

Certified Discrete Log Assumption (CDLA): There exists an invulnerable generator for the set $\{((p, g, c, x), y)\}$, where p is a prime, g is a generator for Z_p^*, c a certificate for the first two facts (e.g. a recursively certified complete factorization of $p - 1$ [22]), and $g^y = x \pmod p$. $\diamond$

We assume that the invulnerable generator for CDL chooses y (and x) with uniform probability from $[1, p-1]$ (any other invulnerable generator can be modified to achieve this using the self randomizing properties of the discrete log problem [2]).

3 Overview of the protocol

In suffices to present a constant round zero knowledge proof of knowledge for one NP complete language. Any other NP statement can be proved in zero knowledge by first reducing the statement to an instance of the NP complete language. This reduction must satisfy three properties:

1. It must be computable in polynomial time.

2. It must be *witness preserving*. This property is necessary for the completeness property of the protocol. It enables P, which has a witness to the original NP statement, to construct a witness to the generated instance of the NP complete language.

3. Witnesses must be *efficiently invertible.* This property is necessary for the knowledge soundness property of the protocol. It allows V to conclude that P knows a witness to the original NP statement, even though P only demonstrates knowledge of a witness to the syntactically different NP complete statement.

The particular NP complete language we choose is DHC (directed Hamiltonian cycle). We specify a polynomial reduction procedure from any NP language L to DHC, to be used whenever such a reduction is called for. This ensures that the reduction process itself conveys no information (and thus the zero knowledge property is preserved). We assume L is initially given as a nondeterministic Turing machine which accepts L. The reduction proceeds in two stages: Reducing the computation of this nondeterministic Turing machine to an instance of SAT (see [8]), and reducing SAT to DHC (see [19]). The reader may check for himself that this chain of reductions is witness preserving and efficiently invertible.

Our starting point is the basic step of Blum's [3] ZKIP for the Directed Hamiltonian cycle (DHC) problem. In the rest of this presentation, we assume that all the graphs are directed, and that all the cycles are directed Hamiltonian cycles.

Protocol 1: Common input: A Hamiltonian graph G with n nodes. P has H, a cycle in G, on his knowledge tape. The basic step of Blum's protocol is composed of three moves:

1. P secretly chooses a random permutation π and permutes the nodes of G. P secretly constructs the adjacency matrix of $\pi(G)$ and commits himself to each entry ('1' if an edge exists, '0' otherwise) independently.

2. V randomly selects a bit '0' or '1' as a challenge and sends it to P.

3. If P receives '0', P reveals π and all the committed bits. V can check that the edges revealed indeed correspond to the graph $\pi(G)$. If P receives '1', P reveals only n edges, comprising a cycle in G. V can easily check that this is the case from the structure of the adjacency matrix.

Blum's full protocol is constructed by sequentially iterating this basic step n independent times (sequential composition). It is complete (truthful P and V can successfully execute the protocol), sound (assuming the commitment scheme is sound, P's ability to complete the protocol implies his knowledge of a cycle in G), and zero knowledge (proven by resettable simulation, assuming the commitment scheme is secure). If the basic step is carried out n times in parallel (parallel composition), the protocol remains complete and sound, but it is conjectured not to be zero knowledge (unless DHC $\in$ BPP [15]).

We did not specify the exact commitment scheme to be used in Protocol 1. GMW [17] suggested using probabilistic encryption functions , where P commits to a bit by encrypting it. A different approach, which relies on P being polynomial

time, was suggested in [5]. P commits to a bit by his internal state of knowledge. For $b \in \{0,1\}$ and commitment c, if P knows a string $w_b(c)$, he can open c as b. P cannot know both $w_0(c)$ and $w_1(c)$, as this implies the knowledge of a certain value which cannot be computed in polynomial time. If furthermore, V has secret "trapdoor" information which allows V (unlike P) to compute matching pairs $(w_0(c), w_1(c))$, the scheme is termed *trapdoor commitment scheme* (or *chameleon* in [5]). Our intention is to use a trapdoor commitment scheme in Protocol 1. In this case, the protocol remains zero knowledge even under parallel composition (which needs only 3 moves). The simulator M, simulating the protocol, would be allowed to open each bit either as 1 or 0 (because V can), and M could imitate P's role without ever resetting P.

The approach described in the previous paragraph is not new, and it serves as the starting point of [6]'s protocol as well. This approach has two problematic aspects:

1. The trapdoor commitment schemes described in [5], rely on very specific cryptographic assumptions (typically number theoretic). We want our protocols to rely on assumptions which are as weak as possible.

2. How do we know that the commitment scheme P uses is really trapdoor? In other words: How do we know that V can open committed bits both as 0 and as 1? V must somehow prove his knowledge of trapdoor information, without revealing the trapdoor information itself. The obvious way of doing this is by V giving a zero knowledge proof that he knows the trapdoor. As our goal is to construct 4-move ZKIPs, it seems unattainable: V's proof must take less than 4 moves, and thus, as noted earlier for 3-move protocols, it cannot be zero knowledge!

In [6] the authors do not set it as a goal to solve the first of the two problems, and the second problem is ingeniously avoided (at the expense of increasing the number of moves to 6). Our new construction solves both problems.

4 A Novel Trapdoor Commitment Scheme

Definition 4.1: A *trapdoor bit commitment scheme* consists of a *commit* stage and a *reveal* stage. The scheme must satisfy the following properties:

- Completeness: Party A can commit to any bit b (either 0 or 1).
- Soundness: A has negligible probability of constructing a commitment which he can later reveal in two possible ways: both as 0 and as 1.
- Security: Party B has negligible probability of predicting the value of a committed bit.

- Trapdoor: B (through some trapdoor information) can construct commitments, indistinguishable from A's commitments, which he can later reveal in two possible ways: both as 0 and as 1.

Our construction of trapdoor commitment schemes is based on the following observation: The basic step of zero knowledge proofs of knowledge can be used as a commitment scheme, provided the prover does not have the knowledge he "claims" to have. Recall the 3-move basic step protocol we presented earlier (Protocol 1). If P could complete the third move in a satisfactory way no matter what V sends in the second move, this would imply P's knowledge of a cycle in G. Conversely, if P does not know a cycle in G, P cannot answer both a '0' challenge and a '1' challenge of V. This leads to the following commitment scheme:

- Preliminary phase: V sends P a Hamiltonian graph G, in which the polynomial time P presumably cannot find a cycle.
- P commits to 0 by choosing a random permutation π, permuting the nodes of G, and committing to the entries of the resulting adjacency matrix (but this time using a commitment scheme which need not be trapdoor!). P may reveal the committed bit '0' by revealing π and the entries of the matrix.
- P commits to 1 by choosing the n node clique and committing to its adjacency matrix (which is all 1). P may reveal the committed bit '1' by opening a random cycle in this matrix.

The above trapdoor commitment scheme has all the desired properties:

- Completeness: P can commit to any bit.
- Soundness: Even a cheating P cannot open the same committed value both as '0' and as '1', for this would imply his knowledge of a cycle in G, and this we assume he does not know.
- Security: V's ability to predict P's bit from the commitment phase implies V breaking the nontrapdoor commitment scheme.
- Trapdoor: If V knows a cycle in G, he can open bits he originally committed to as '0', both as '0' and as '1'.

It remains to fill in a few technical details.

- Our trapdoor commitment scheme is based on a nontrapdoor commitment scheme. Nontrapdoor commitment schemes can be constructed from any one-to-one oneway function (e.g. see [16]). At the price of one preliminary move, they can be constructed from any oneway function (see [20] and [21]).

- To make the trapdoor commitment scheme sound, we have to assume that P does not know a cycle in G, and thus V must choose a difficult instance of DHC. This can be done easily under IGA (see definition 2.6). Using the invulnerable generator, V chooses a random instance $(x, w) \in S$, where S is the corresponding invulnerable set, and reduces the problem "there exists w such that $(x, w) \in S$" to an instance G of the NP complete problem DHC. Since the reductions are efficiently invertible, finding a cycle in G implies associating a witness to the original x, and this problem is assumed to be hard.

The commitment scheme we constructed is trapdoor. This is necessary for the zero knowledge property of our 2-round protocol. But zero knowledge is a property which protects honest provers P against dishonest verifiers V. If V is dishonest, how can P trust him to send a graph which indeed contains a cycle? And even if the graph does contain a cycle, how do we know that V can find such a cycle? The answer is that a preliminary step is missing: V must prove beforehand that he knows a cycle in G. This must be done under the following constraints:

- V's proof must convince P.
- P must not be able to use this proof to learn a cycle in G.
- A simulator M (which is allowed to use V as a subroutine) should be able to extract the cycle from V.

This can be done easily. The commitment scheme is the basic step of some zero knowledge proof of knowledge. Thus V can just execute the complete zero knowledge proof of knowledge to show that he himself knows a cycle, without revealing any useful information about this cycle to P.

There is one problem left. V's proof takes too many moves, as in order to guarantee the zero knowledge property, V must employ sequential composition of the basic step. However, the zero knowledge property is not really necessary in V's proof. In order to guarantee the soundness property of the trapdoor commitment scheme it suffices that P does not learn any cycle in G from V's proof, and we do not care if other "irrelevant" information is revealed to P. This extra flexibility in the security demands of V's protocol can be exploited by employing parallel composition instead of sequential composition in the construction of V's proof of knowledge from Protocol 1. In the next section we show that under suitable conditions (which are easy to meet) the parallel composition, though not zero knowledge, does not reveal any cycle in G.

5 Introduction to Witness Hiding Protocols

The concept of witness hiding protocols is a development of the concept of "transferable information", used in proving that the parallel version of the Fiat-Shamir

protocol is secure [12]. For a full treatment of witness hiding protocols see [11]. This section only offers an introduction sufficient for the purposes of this paper.

The concept of Witness Hiding (WH - to be defined shortly) is a possible alternative to zero knowledge. It is a weaker requirement than zero knowledge, but in many cases, it still satisfies the security demands of cryptographic protocols. It comes together with a technical tool (witness indistinguishability), which replaces the technique of resettable simulation. The advantage WH has over zero knowledge is that (under well defined conditions) it is preserved under general composition of protocols. As a special case of general composition, it is preserved under parallel composition.

Informally, a protocol (P,V) is WH if participating in the protocol does not help the verifier (which can be either the original V or a cheating V') to compute appropriate witnesses to the input. This is a natural security requirement of many cryptographic protocols. In order to prove the WH property, one must show that if V' can compute a witness to the input after participating in the interactive proof, then he had this capability in him even before the protocol began. To this end we introduce the witness extractor M. We give the technical definition of this concept, and refer the reader to [11] for a discussion on this definition and a comparison between WH and zero knowledge.

Definition 5.1: Let (P,V) be a proof of knowledge system for language L. Let $\pi = \pi(n)$ be a probability distribution on the inputs $x \in L$ of size n, and on their corresponding witnesses $w(x)$. (P,V) is *witness hiding* (WH) on (L,π) if there exists a witness extractor M which runs in expected polynomial time, such that for any nonuniform polynomial time V'

$$Prob(V'_{P(x,w)}(x) \in w(x)) - Prob(M(x,V') \in w(x)) < \nu(n)$$

The probability is taken over the distribution of the inputs and witnesses, as well as the random tosses of P and M. The witness extractor is allowed to use V' (but not P) as a subroutine. $\diamond$

In order to make use of the notion of WH, we need a technical tool for proving that protocols are WH. For this end we define *witness indistinguishability* (WI).

Definition 5.2: Proof system (P,V) is *witness indistinguishable* (WI) if for any V', for any large enough input x, for any $w_1 \in w(x)$ and $w_2 \in w(x)$, and for any auxiliary input y for V', the ensembles, $V'_{P(x,w_1)}(x,y)$ and $V'_{P(x,w_2)}(x,y)$, generated as V's view of the protocol are indistinguishable.$\diamond$

WI involves no simulator M, and is suggested as an alternative to the resettable simulation technique. The next Theorem shows that WI is implied by zero knowledge.

Theorem 5.1: Let (P,V) be any zero knowledge protocol. Then the protocol is WI.

Proof (sketch): The proof follows from the transitivity of the indistinguishability relation. For input x, assume distinguisher D has probability p of outputting 1 on V's view of P's proof, when P is using w_1. By the zero knowledge property,

D has the same probability p (up to negligible additive terms) of outputting 1 on the simulated view created by M. But the view M creates is independent of the witness P is using. Thus D has probability p of outputting 1 on V's view even if P is using w_2. $\diamond$

Theorem 5.2: WI is preserved under parallel composition of protocols.

Proof (sketch): Consider polynomially many parallel executions of a WI protocol (P,V). Assume that there exists a verifier V' for which this parallel composition is not WI. That is, there exist infinitely many n, inputs $x(n)$, auxiliary inputs $y(n)$ to V', and witnesses $w_1(n)$ and $w_2(n)$, such that the two ensembles $V'_{P(x,w_1)}(x,y)$ and $V'_{P(x,w_2)}(x,y)$ are polynomially distinguishable. Then somewhere there must be a "polynomial jump": For any n, there exists $k(n)$, such that if P uses witness $w_1(n)$ for executions of index less than $k(n)$, and P uses witness $w_2(n)$ for executions of index greater than $k(n)$, the ensembles (which differ only in the witness used in iteration $k(n)$) are distinguishable. We construct a modified cheating V^* for the original protocol (P,V). V^* has as auxiliary input $y^*(n) = (y(n), k(n), w_1(n), w_2(n))$. This random polynomial time V^*, when interacting with truthful P, simulates by himself all the other parallel iterations which are taking place with V'. We use here the fact that both V' and P are polynomial time, and so V^* can simulate both these protocols. Now V^* can distinguish between truthful P using $w_1(n)$ and $w_2(n)$, which is a contradiction to our assumption that the original protocol was WI. $\diamond$

The above two Theorems establish a methodology for constructing WI protocols. Take the basic step of a ZKIP. By Theorem 5.1 it is also WI. Iterate the basic step n times in parallel. This is not zero knowledge, but by Theorem 5.2, it is WI. What we need now is to establish a connection between WI and WH. We cannot prove that any WI protocol is also WH, but we can specify conditions under which WI implies WH. These conditions involve the particular method by which input instances are generated. A protocol may be trivially WI because every input has only one possible witness. In this case WI cannot imply anything. But if any input has at least two "independent" witnesses, than the WI property is nontrivial, and it may be possible to infer WH from WI. We demonstrate this point by the following protocol, which will subsequently be used in our *perfect* zero knowledge 2-rounds proofs of knowledge.

Protocol 2 is based of the discrete log problem. We modify this problem so that each instance of the new modified problem has two independent witnesses.

Protocol 2: The common input is generated by a slight twist to the invulnerable generator of the certified discrete log assumption, which forces the input to have two witnesses. The input is (p, g, c, x_1, x_2), where p, g, and c are as described in CDLA, and x_1 and x_2 are integers in Z_p^* chosen randomly and independently. This instance is defined to have two possible witnesses: w_1 satisfying $g^{w_1} = x_1 \bmod p$, and w_2 satisfying $g^{w_2} = x_2 \bmod p$. (w_i is the *discrete logarithm* of x_i). P receives as witness $w \in \{w_1, w_2\}$. P proves that he knows the discrete log of either x_1 or x_2. The basic step of the protocol is:

1. P chooses secretly, randomly and independently r_1, r_2, and computes $y_1 = x_1 g^{r_1} \bmod p$, $y_2 = x_2 g^{r_2} \bmod p$. P sends these two values in a random order to V.

2. V replies by a random challenge: 0 or 1.

3. If P receives 0, P reveals r_1 and r_2, and V checks that y_1 and y_2 were constructed correctly (satisfy $y = xg^r \bmod p$). If P receives 1, P reveals the discrete log of *only one of* the y's (which equals $w + r \pmod{p-1}$).

This basic step is executed $\log p$ independent times *in parallel.*

Theorem 5.3: The above protocol is a complete and sound witness hiding proof that P knows the discrete log of one of two inputs, under the distribution of inputs specified in CDLA (see section 2).

Proof (sketch): The proof of the completeness and soundness properties of the protocol is standard. We prove only the WH property.

The basic step of the protocol is zero knowledge (and even perfect zero knowledge). The parallel composition of the basic step gives a protocol which is presumably not zero knowledge, but still (by Theorems 5.1 and 5.2) this protocol is WI. Assume that for some V' Protocol 2 is not WH. Then V' learns with nonnegligible probability one of the discrete logs (w.l.o.g., assume it is w_1). But because of WI, V' has the same probability of learning w_1 whether P is really using w_1 or the other witness w_2. This gives a random polynomial time algorithm for computing the discrete log, contradicting CDLA: On input (p, g, c, x), M has to compute w satisfying $x = g^w \bmod p$. M chooses randomly and uniformly w_2 and creates $x_2 = g^{w_2} \bmod p$. M sets $x_1 = x$ and uses (p, g, c, x_1, x_2) as input to Protocol 2. In this protocol M simulates P and uses his control over V' to obtain V's replies. By our assumption, V' has nonnegligible probability of extracting w_1 from this protocol. Thus M has nonnegligible probability of computing the discrete log of x, violating CDLA. $\diamond$

The basic step of protocol 2 can be used as a trapdoor commitment scheme, provided P knows neither w_1 nor w_2. P commits to 0 by constructing y_1 and y_2 as in move 1 of protocol 2. P commits to 1 by choosing random r, and constructing one of the y's as g^r.

In order to construct the 3-move WH protocol (protocol 2), we composed two random instances of the discrete log problem. This procedure of composing two random instances can be followed with any NP language. Protocol 2 uses self randomizing [2] properties of the discrete log. For other NP languages, we may not have direct protocols for proving knowledge of one of two witnesses, and in this case, P and V may reduce the composed input instance to an instance of the NP-complete set DHC, and use protocol 1. If the instances of the original NP language are generated by an invulnerable generator, then the proof that the resulting protocol is WH is similar to the proof of Theorem 5.3. In [11] we prove a stronger and more general Theorem, stated here without proof.

Theorem 5.4: Let G be any generator for a set $S = \{(x, w)\}$ recognizable in BPP (which need not be invulnerable). Let π be the distribution obtained by two

independent applications of G, taking (x_1, x_2) as common input, and one of w_1 or w_2 at random as a corresponding witness. Let (P, V) be any WI system for proving knowledge of a witness w s.t. $((x_1, w) \in S \text{ or } (x_2, w) \in S)$. Then (P, V) is WH. $\diamond$

6 The Full Protocols

We first present a 3-round *perfect* zero knowledge proof of knowledge for DHC, based on CDLA. Then we show how this protocol can be modified to two rounds. Finally we show how the cryptographic assumption can be weakened, from CDLA to the existence of any one-to-one oneway function, at the price of obtaining a protocol which is only computationally zero knowledge. The construction of 5-move protocols from any oneway function (using Naor's [21] bit commitment scheme) is an easy consequence of our techniques.

Protocol 3a: Perfect zero knowledge proof of knowledge for DHC. Cryptographic assumption: CDLA. Protocol 3a is a sequential composition of Protocol 2 and Protocol 1, described in sections 5 and 3 respectively.

Common input: G, a Hamiltonian graph. P's witness is H, a cycle in G.

Move 1: V uses the modified invulnerable generator based on CDLA to generate an input $I = (p, g, c, x_1, x_2)$ for protocol 2. V is to act as prover in protocol 2, and so he discards randomly either w_1 or w_2, and keeps the other (denoted as w) as his auxiliary input. V sends I to P.

Null Move: P uses the certificate c to check that p is prime and g a generator. If this check fails, P stops.

Moves 1-3: V and P now perform Protocol 2 with I as input. Note that in this subprotocol, the roles are reversed: V is the prover (with w as auxiliary input) and P is the verifier.

Null Move: If Protocol 2 is completed successfully, than P is convinced that V knows a witness to I. Otherwise P stops.

Moves 4-6: P and V execute the parallel composition of Protocol 1, with common input G, and P's auxiliary input is H. As a bit commitment scheme, P uses the basic step of Protocol 2 with I as input. This commitment scheme is trapdoor, because V already proved (in the previous subprotocol) that he knows a witness for I.

Null Move: If Protocol 1 is completed successfully, V accepts.

$\diamond$

Protocol 3a consists of 6 moves. In order to reduce the number of moves to 4 (i.e. the number of rounds to 2), we modify this protocol:

Protocol 3: 2-Round perfect zero knowledge proof of knowledge for DHC. The protocol proceeds exactly as protocol 3a, but executes subprotocols 2 and 1 almost in parallel by regrouping the six moves into four super-moves: (1); (2,4); (3,5); (6). (The reader should work out this regrouping by himself, from the explicit description of protocols 1 and 2). $\diamond$

Protocol 3 assumes CDLA, which is a strong assumption. The modified discrete log problem was used as a trapdoor commitment scheme. In order to use a weaker cryptographic assumption, we base the trapdoor commitment scheme on DHC (as explained in section 4). The protocol we obtain this way is only computational zero knowledge.

Protocol 4a: Computational zero knowledge proof of knowledge for DHC. Cryptographic assumptions: IGA and the existence of bit commitment schemes. Both assumptions follow from the single assumption that one-to-one oneway functions exist.

Common input: G, a Hamiltonian graph. P's witness is H, a cycle in G.

Move 1: V uses the invulnerable generator to create two hard certified instances $(x_1, w_1) \in S$ and $(x_2, w_2) \in S$. V reduces the NP statement "there exists w such that either $(x_1, w) \in S$ or $(x_2, w) \in S$" to an instance I of the NP complete problem DHC. The witnesses w_1 and w_2 are transformed into two cycles in I, of which V randomly discards one and keeps the other (denoted as w). V sends (I, x_1, x_2) to P.

Null Move: P checks that I is indeed obtained from (x_1, x_2) by the publicly known polynomial reduction. If this check fails, P stops.

Moves 1-3: V and P now perform the parallel composition of Protocol 1 on input I. Note that in this subprotocol, the roles are reversed: V is the prover (with auxiliary input w) and P is the verifier. In order to execute the prover's part in Protocol 1, V must randomly choose a bit commitment scheme. We denote this commitment scheme by C_V.

Null Move: If Protocol 1 is completed successfully, then P is convinced that V knows a witness to I. Otherwise P stops.

Moves 4-6: P and V again execute the parallel composition of Protocol 1, this time with P as the prover, V as the verifier, G as the common input, and H as P's auxiliary input. As a trapdoor bit commitment scheme, P uses the basic step of the same Protocol 1, but with I as input. For this basic step, P needs to use another (nontrapdoor) bit commitment scheme (see Protocol 1), and we denote this commitment scheme by C_P.

Null Move: If Protocol 1 is completed successfully, V accepts.

◇

Protocol 4a takes 6 moves. As before, we can reduce the number of moves to 4 by parallelizing some of the steps:

Protocol 4: 2-Round computational zero knowledge proof of knowledge for DHC. Transform protocol 4a into protocol 4 in exactly the same way as protocol 3a is transformed to protocol 3. ◇

7 Correctness

Theorem 7.1: Under the certified discrete log assumption, Protocol 3 is a complete, sound and perfect zero knowledge proof of knowledge of a Hamiltonian cycle

in a directed graph.

Proof (sketch): We sketch the proof for Protocol 3a. The proof for Protocol 3 is similar. We need to prove three properties: Completeness, soundness, zero knowledge.

Completeness: Trivial.

Soundness: We describe a knowledge extractor M, which stops in expected polynomial time, and the probability it outputs a cycle in G is the same (up to negligible additive terms) as the probability that P' convinces a truthful V.

Given (a possibly cheating) P', M first executes the whole protocol (P', V), by faithfully simulating V's part. If V rejects, M stops and outputs nothing. Otherwise, M repeatedly resets P' to step 5 of the protocol, chooses new random challenges (in step 2 of Protocol 1), until P' again meets these challenges successfully. When this happens, or if P' failed to answer 2^n successive challenges, M stops. Now there are three possible events:

1. P' failed to answer 2^n successive challenges.
2. The second set of challenges was met by P' by opening some committed bit differently than in his original success.
3. The soundness of the trapdoor commitment scheme was not violated, and from the two successful executions, M can derive a cycle in G. (This is because P' opened the committed matrix and showed its isomorphism to G, and showed a cycle in the same matrix).

We have to show two things: That M's expected running time is polynomial, and that the first two events have negligible probability. In order to analyze M's running time, we specify by p P's conditional probability of completing the protocol, given that move 4 was completed. Then M's expected running time is proportional to $(1-p)\cdot 1 + p\cdot\frac{1}{p} < 2$ times the running time of V (which is polynomial). This also shows that the probability of the first event is negligible $(O(2^{-n}))$. As to the probability of the second event: Consider its a-priori probability at the beginning of the protocol. If it is negligible, we ignore it. If it is not negligible, this violates the CDLA, as described in the proof of Theorem 5.3 (the WH property of Protocol 2).

Zero-knowledge: We describe a simulator M, which for any (possibly cheating) V', creates in expected polynomial time a view of the protocol indistinguishable from the view of V'. The simulator first performs P's part in moves 1-3. If V' does not complete this subprotocol successfully, M stops. Otherwise, M repeats move 2, each time with different randomly chosen challenges, until V' again successfully meets M's challenges. From the two successful executions M can find a discrete log w. To guard against an infinite execution in case there is only one set of challenges that V answers correctly, M tries in parallel to find w_1 by himself (using exhaustive search).

Once M finds a w, he can create instances of the trapdoor commitment scheme which he can open both as 0 and as 1. This allows him to carry out P's part in moves 4-6, without knowing a Hamiltonian cycle and without using resettable simulation.

The analysis of M's running time is similar to the analysis of knowledge soundness. The view created is perfectly indistinguishable from V's view of the protocol when executed with a real P. $\diamond$

Theorem 7.2: Under the invulnerable generator assumption, and under the assumption that secure (nontrapdoor) bit commitment schemes exist, Protocol 4 is a complete, sound and computational zero knowledge proof of knowledge of a Hamiltonian cycle in a directed graph.

Proof (sketch): We sketch the proof for Protocol 4a (the proof for Protocol 4 is similar). The main new complication is the use of the two nontrapdoor commitment schemes: C_V and C_P. We concentrate only on aspects that do not appear in the proof of Theorem 7.1.

Completeness: Proving this property involves only one subtle point: The derivation of I, which serves as the basis of the trapdoor commitment scheme, is done by a polynomial reduction to DHC which preserves witnesses. This ensures that V can complete his part of the protocol in moves 1-3, where he has to know a cycle in I.

Soundness: The knowledge extractor M works in a way analogous to the proof of Theorem 7.1. The proof that its expected running time is polynomial is similar. Complications arise from two sources:

1. In addition to the three events we had earlier, we now have a fourth possible event: P' met two challenges, by violating the soundness property of his nontrapdoor commitment scheme C_P.

2. P' may learn a cycle in I in moves 1-3, by violating the security of V's nontrapdoor commitment scheme C_V.

Thus in proving the knowledge soundness property, we must exclude the above two events as having negligible probability. This follows from the assumption that there exist sound and secure bit commitment schemes.

Zero knowledge: Again, the nontrapdoor commitment schemes cause complications.

1. The simulator M, in trying to extract a witness for I from moves 1-3, may not succeed, and instead discover a violation of the soundness of C_V (a bit commitment V once opened as 0 and once as 1).

2. The protocol is not perfect zero knowledge. The view generated by M differs from the view generated in real executions of the protocol in the value of the unopened commitments to bits by C_P.

In proving that the protocol is zero knowledge, we must prove that the first event has negligible probability. This follows from the assumption that C_V is a sound commitment scheme. We must also prove that no nonuniform polynomial time distinguisher can take advantage of the differences between the two ensembles. This follows from the assumption that C_P is a secure commitment scheme. $\diamond$

8 Concluding Remarks

Witness hiding is an attractive alternative to zero knowledge not only in interactive proofs. Its use offers advantages also in noninteractive proofs. Noninteractive zero knowledge proofs, as introduced in [4] and [10], postulate the existence of a publicly known random string (such as tables of random numbers prepared by the RAND corporation). [4] and [10] show how a prover may use this random string to write down a noninteractive zero knowledge proof of any NP statement. However, the prover may not use the same common random string in order to prove many (more than $\log n$) statements, since the zero knowledge property breaks down. In [11] we show that WH does not suffer from the same drawback, and noninteractive witness hiding protocols may use the same common random string repeatedly, without jeopardizing the WH property. This property is valuable in using noninteractive WH protocols as a basic primitive in the construction of more complicated cryptographic primitives (such as signature schemes [7]). More details appear in [11].

Acknowledgements

We thank Gilles Brassard, Joan Feigenbaum, Oded Goldreich, Shafi Goldwasser, Joe Kilian and Mike Luby for helpful discussions.

References

[1] M. Abadi, E. Allender, A. Broder, J. Feigenbaum, L. Hemachandra, *On Generating Solved Instances of Computational Problems* Proc. of CRYPTO88.

[2] D. Angluin, D. Lichtenstein, *Provable Security of Cryptosystems: a Survey* TR-288, Yale University, 1983.

[3] M. Blum, *How to Prove a Theorem So No One Else Can Claim It* Proc. of the International Congress of Mathematicians, Berkeley, California, USA, 1986, pp. 1444-1451.

[4] M. Blum, P. Feldman, S. Micali, *Non-Interactive Zero-Knowledge and its Applications* Proc. of 20th STOC 1988, pp. 103-112.

[5] G. Brassard, D. Chaum, C. Crépeau, *Minimum Disclosure Proofs of Knowledge* JCSS, Vol. 37, 1988, pp. 156-189.

[6] G. Brassard, C. Crépeau, M. Yung, *Everything in NP can be argued in perfect zero-knowledge in a bounded number of rounds* Proc. of 16th ICALP, Stresa, Italy, 1989.

[7] M. Bellare, S.Goldwasser, *New Paradigms for Digital Signatures and Message Authentication Based on Non-Interactive Zero Knowledge Proofs* these proceedings.

[8] S. Cook, *The Complexity of Theorem Proving Procedures* Proc. of 3rd STOC 1971, pp. 151-158.

[9] I. Damgård, *On the Existence of Bit Commitment Schemes and Zero Knowledge Proofs* these proceedings.

[10] A. De Santis, S. Micali, G. Persiano, *Non-Interactive Zero-Knowledge Proof Systems* Proc of CRYPTO-87, pp. 52-72.

[11] U. Feige, A. Shamir, *Witness Hiding Protocols and Their Applications* In preparation.

[12] U. Feige, A. Fiat, A. Shamir, *Zero Knowledge Proofs of Identity* Journal of Cryptology, Vol 1, 1988, pp. 77-94. (Preliminary version in Proc. of 19th STOC 1987, pp. 210-217.)

[13] L. Fortnow, *The Complexity of Perfect Zero-Knowledge* Proc. of 19th STOC, 1987, pp. 204-209.

[14] O. Goldreich, private communication.

[15] O. Goldreich, H. Krawczyk, *On the Composition of Zero-Knowledge Proof Systems* manuscript, May 1989.

[16] O. Goldreich, L. Levin, *A Hard-Core Predicate for all Oneway Functions* Proc. 21st STOC 1989, pp. 25-32.

[17] O. Goldreich, S. Micali, A. Wigderson, *Proofs that Yield Nothing But Their Validity and a Methodology of Cryptographic Protocol Design* Proc. 27th FOCS, 1986, pp. 174-187.

[18] S. Goldwasser, S. Micali, C. Rackoff, *The Knowledge Complexity of Interactive Proof Systems* SIAM J. Comput. Vol. 18, No. 1, pp. 186-208, February 1989.

[19] E. Horowitz, S. Sahni, *Fundamentals of Computer Algorithms* Computer Science Press, Computer Software Engineering Series, (pp. 526-529).

[20] R. Impagliazzo, L. Levin, M. Luby, *Pseudo-Random Generation From Oneway Functions* Proc. 21st STOC, 1989, pp. 12-24.

[21] M. Naor, *Bit Commitment using Pseudo-Randomness* these proceedings.

[22] V. Pratt, *Every Prime Has a Succinct Certificate* SIAM J. Computing 4 (1975), pp. 214-220.

[23] M. Tompa, H. Woll, *Random Self-Reducibility and Zero Knowledge Interactive Proofs of Possession of Information* Proc. 28th FOCS, 1987, pp. 472-482.