Zero Knowledge Proofs of Identity

Uriel Feige, Amos Fiat* and Adi Shamir

Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot 76100, Israel

(extended abstract)

Abstract

In this paper we extend the notion of zero knowledge proofs of membership (which reveal one bit of information) to zero knowledge proofs of knowledge (which reveal no information whatsoever). After formally defining this notion, we show its relevance to identification schemes, in which parties prove their identity by demonstrating their knowledge rather than by proving the validity of assertions. We describe a novel scheme which is provably secure if factoring is difficult and whose practical implementations are about two orders of magnitude faster than RSA-based identification schemes. In the last part of the paper we consider the question of sequential versus parallel executions of zero knowledge protocols, define a new notion of "transferable information", and prove that the parallel version of our identification scheme (which is not known to be zero knowledge) is secure since it reveals no transferable information.

1. Introduction

Zero knowledge proofs (Goldwasser, Micali and Rackoff [1985]) are an elegant technique to limit the amount of information transferred from a prover A to a verifier B in a cryptographic protocol. As defined in the original GMR paper, the proofs refer to language membership problems (is input I a member of language L?), and their applicability to any language L in NP was recently demonstrated in Goldreich, Micali and Wigderson [1986]. Additional properties of zero knowledge proofs were investigated in Goldwasser and Sipser [1986], Brassard and Crepeau [1986], Chaum [1986], and many other papers.

The name "zero knowledge proofs" is slightly misleading, since the prover A reveals one bit of knowledge to the verifier B (namely that I belongs to L). Our first objective in this paper is to show that it is possible to extend this notion to "truly zero knowledge proofs" which do not even reveal this single bit. The basic idea is to replace "knowledge" by "knowledge about knowledge": A's goal is not to prove that I *belongs* to L, but to prove that he *knows* the status of I with respect to L. From B's point of view, he didn't get any information whatsoever about "the real world" (I, L and their relationships) – only about A's state of knowledge concerning the real world.

As a motivating (but technically inaccurate) example, consider a prover A who wants to prove to a skeptical B that he has settled Fermat's last theorem. With the type of proof introduced in this paper, A can convince B that he is a mathematical superstar without telling B anything new about the problem – not even whether he has found a proof or a counterexample!

A related idea was presented by Galil Haber and Yung [1985], who defined the notion of "result indistinguishable protocols". However, the two models differ in their goals: In their model the prover A proves either that I belongs to L or that I does not belong to L, B knows which claim is being proven and gets a convincing proof, while the passive eavesdropper C (who is not allowed to participate or meddle in the protocol) cannot determine from the communication tape which claim is being proven and whether the tape constitutes a convincing proof. The main difficulty in extending such a result to our model is that we want the same party B to be ignorant of what is being proven and yet to be convinced by the proof.

The intuitive observation that the zero knowledge paradigm could be used to prove knowledge rather than existence of witnesses is not new, and appeared (without proper formalization) in several paranthetical remarks in Goldwasser, Micali and Rackoff [1985], Chor, Goldwasser, Micali and Awerbuch [1985], Galil, Haber and Yung [1985], Chaum [1986], and several other papers. As demonstrated in this paper, the formal definitions of "proofs of membership" and "proofs of knowledge" are quite different, and there is a fundamental philosophi-

* Current address: EECS, The University of California at Berkeley

cal difference between them. The notion of "knowledge" is very fuzzy, and a-priori it is not clear what proofs of knowledge actually prove. Several researchers have investigated this notion from a different point of view (see e.g. Halpern and Moses [1986], and Rosenschein [1985]), and we believe that a combined approach to knowledge and proofs of knowledge can have an important impact on areas outside cryptography (in particular logic and distributed computing).

2. Interactive proofs of knowledge

Our model differs from the original GMR model in several important aspects. We do not allow the prover to be infinitely powerful (such a prover knows everything about its inputs by definition), and restrict both the prover and the verifier to be polynomial time probabilistic Turing machines. The machines have a common input tape I, two communication tapes CA and CB, private work tapes WA and WB, and private random tapes RA and RB. In addition, the prover is given access to a restricted oracle in the form of a private tape S. The contents of this tape satisfies a publicly known polynomial time predicate $P(I,S)$ whenever the predicate is satisfiable, and remains blank otherwise. Note that the prover cannot ask the oracle additional questions about the same or other inputs, and has to rely only on the given witness S in his proof of knowledge. This model restricts A's proofs of knowledge to problems in NP, and typical examples of predicates are "S is a valid 3-colouring of graph I" and "S satisfies the CNF formula I".

In the rest of this paper we adopt the following conventions:

Notation:

1. $\overline{A}$ (straight A) represents the real prover who follows its designated protocol by using its explicit access to the oracle tape.
2. $\widetilde{A}$ (crooked A) represents a polynomial time cheater who cannot access the oracle tape but can deviate from the protocol in an arbitrary way.
3. A represents either $\overline{A}$ or $\widetilde{A}$.
4. $\overline{B}$ (straight B) represents the real verifier who follows its designated protocol.
5. $\widetilde{B}$ (crooked B) represents an arbitrary polynomial time program which tries to extract additional information from $\overline{A}$.
6. B represents either $\overline{B}$ or $\widetilde{B}$.
7. (A,B) represents the execution of the two party protocol in which A is the prover and B is the verifier.

To demonstrate the subtlety of interactive proofs of knowledge, consider the following examples in which I is the product of two primes and A claims that it knows its factorization S. We do not insist that the proofs should be zero knowledge, and we use the fact that factoring and square root extraction are computationally equivalent. The question we address in each case is whether A's proof of knowledge should convince $\overline{B}$:

Example 1: A extracts the square roots mod I of several substrings $R_1 \cdots R_k$ from RA, and sends RA and the roots of about one quarter of the R_i to $\overline{B}$. $\overline{B}$ should not be convinced since he cannot verify that A really uses his random tape.

Example 2: $\overline{B}$ sends several substrings $R_1 \cdots R_k$ from RB to A, and receives their square roots mod I in one quarter of the cases. Even though $\overline{B}$ itself could not use the results to compute S, it is clear that a slightly modified $\overline{B}'$ could compute S with high probability by sending A the squares mod I of these substrings.

Example 3: Let f be a strong one way function which is known to both parties. $\overline{B}$ sends several substrings $R_1 \cdots R_k$ from RB to A and receives the square roots of one quarter of the values $f(R_1) \cdots f(R_k)$. The problem is not completely defined, but $\overline{B}$ should be wary of A's proof: Even though arbitrary square roots cannot be extracted without knowledge of the factorization of I, the randomizing function f prevents even a modified $\overline{B}'$ from getting the second square root of the same number it needs to factor I.

Example 4: To show that $\overline{B}$'s skepticism in Example 3 can sometimes be justified, consider the function $f(x) = x^2 (\mathrm{mod}\ I)$. This is probably a strong one way function but it interacts badly with the problem at hand, and enables a cheating $\widetilde{A}$ to extract square roots even when it does not know the factorization of I.

Example 5: Consider now the function $f(x) = ax^2 + b (\mathrm{mod}\ I)$ with non zero a and b. The multiplication by a and addition of b seems to destroy the ability of $\overline{B}'$ to factor I, which suggests that $\overline{B}$ should not be convinced. However, Pollard proved that binary quadratic equations such as $ax^2 + b = y^2 (\mathrm{mod}\ I)$ can be solved in polynomial time, and thus $\overline{B}'$ can prepare an x for which he already knows one square root y of $f(x)$, but A cannot know which. As a result, $\overline{B}$ should be convinced in this case that A knows the factorization of I.

Formalizing the concept of "interactive proofs of knowledge" is not easy. GMR's definition of "interactive proofs of membership" is based on the language recognition paradigm: Some intrinsic property of the inputs I is used to define the subset L of "good" inputs, and then we can check the adequacy of a particular protocol $\overline{A}\ \overline{B}$ by verifying that for all I and $\widetilde{A}$:

1. If I belongs to L, $\overline{B}$ accepts $\overline{A}$'s proof with overwhelming probability.
2. If I does not belong to L, $\overline{B}$ accepts $\widetilde{A}$'s proof with negligible probability.

In other words, $\overline{B}$ accepts exactly L in the sense that the real $\overline{A}$ can convince $\overline{B}$ to accept at least L, but even a faulty $\tilde{A}$ cannot convince $\overline{B}$ to accept more than L.

The existence of some S which satisfies $P(I,S)$ is an intrinsic property of I which can be used to distinguish between "good" and "bad" inputs. However, there are many cases in which proofs of existence of S are meaningless, and what makes the proofs surprising in these cases is only their constructive nature which demonstrates knowledge. Consider, for example, the following predicates $P(I,S)$:

1. S is the complete factorization of I.
2. S is the discrete logarithm of I modulo a prime Q.
3. S is either a short witness for square freeness or a number larger than 1 whose square divides I.

In all these examples the mere existence of S can be proven by $\overline{B}$ in advance (in examples 1 and 3 the predicate is always satisfiable, and in example 2 the exceptions are the set of multiples of Q, which is polynomially recognizable). When S always exists, the original GMR definition degenerates to the uninteresting "$\overline{B}$ should always accept". What we really want is to distinguish between inputs I for which S is "known" and inputs I for which S is "not known". However, the subset of I for which S is "known" is ill-defined: The same I should sometimes be accepted and sometimes be rejected by $\overline{B}$. Since the proper decision depends on A's state of knowledge rather than on I's intrinsic properties, we can try to reformulate the two conditions in terms of A instead of I:

1. If A is $\overline{A}$, $\overline{B}$ should accept its proofs with overwhelming probability for all I for which $P(I,S)$ is satisfiable.
2. If A is $\tilde{A}$, $\overline{B}$ should accepts its proofs with negligible probability for all I.

This definition is motivated by the fact that $\overline{A}$ is always given access to the oracle tape that contains S, whereas $\tilde{A}$ is never given access to this tape. However, even if P is a difficult predicate, there can be infinitely many "easy" instances of I for which $\tilde{A}$ can compute the S by itself and then mimic $\overline{A}$'s proof to $\overline{B}$. Consequently, we have to allow $\overline{B}$ to accept $\tilde{A}$'s proofs occasionally, but only when it really happens to know S. In other words, we require that for all I and A (which can be either $\overline{A}$ or $\tilde{A}$):

1. If A knows S, $\overline{B}$ should accept A's proof for I with overwhelming probability.
2. If A does not know S, $\overline{B}$ should accept A's proof for I with negligible probability.

This makes it necessary to define the set of things that a particular program knows. An informal definition of this concept was given in one of GMR's paranthetical remarks:

> $\tilde{A}$ knows S if there is some polynomial time Turing machine M with complete control over $\tilde{A}$ which prints S as a result of its interaction with $\tilde{A}$.

The notion of knowledge captured by this definition is very broad, and $\tilde{A}$ may not even be "aware" of its knowledge of S. For example, S may be stored in an inaccessible portion of $\tilde{A}$'s code, or may appear temporarily in its work tape, or may be computed only as a complicated function of values derived from polynomially many executions of $\tilde{A}$ during which M repeatedly changes the contents of $\tilde{A}$'s random and work tapes. Unfortunately, this notion of knowledge is incompatible with the definition of interactive proofs of knowledge given above: In the first part of the definition A may "know" S in some bizarre way which cannot be detected by the particular $\overline{B}$, and on the other hand the second part of the definition becomes the tautology "if A cannot convince anyone that it knows S, it should not convince $\overline{B}$ that it knows S".

Having seen the many pitfalls along the path, we finally propose our formal definition of interactive proofs of knowledge. It combines most of of the ideas discussed so far, but blends them in a different way:

Definition: A pair of interacting polynomial time probabilistic Turing Machines $\overline{A}$, $\overline{B}$ is called an interactive proof system of knowledge for the polynomial time predicate $P(I,S)$ if:

1. For all I for which $P(I,S)$ is satisfiable, the execution of $(\overline{A},\overline{B})$ on input I succeeds with overwhelming probability.
2. There exists an M (with complete control over $\tilde{A}$) such that for all $\tilde{A}$, RA and sufficiently large I, if the execution of $(\tilde{A},\overline{B})$ on input I succeeds with non negligible probability, then the output produced by M at the end of the execution of $(\tilde{A},M)$ on input I satisfies the predicate P with overwhelming probability. More formally,

$$\forall a\ \exists M\ \forall \tilde{A}\ \ \forall RA \forall b\ \exists c\ \forall |I| > c$$
$$\text{Prob}\big((\tilde{A},\overline{B}) \text{ accepts } I\big) > 1/|I|^a \Longrightarrow$$
$$\text{Prob}\big(\text{output of } (\tilde{A},M) \text{ on } I \text{ satisfies } P\big) > 1 - 1/|I|^b.$$

Remarks:

1. We want $\overline{B}$'s conviction that $\tilde{A}$ knows S to depend only on the randomness of its own coin flips, and thus we universally quantify over the RA and define the probabilities only over the random choices of RB. To execute $\tilde{A}$ over arbitrarily large inputs, we assume that RA is infinite but only polynomially long prefixes in it are actually scanned.
2. The parameter a specifies the meaning of "non negligible" while the parameter b specifies the meaning of "overwhelming". The choice of M may depend on

the former but not on the latter, since we want the asymptotic failure rate of the chosen M to be smaller than the inverse of any polynomial.

3. The uniformity of the "interrogator" M which extracts S from any $\tilde{A}$ that manages to convince $\overline{B}$ makes it possible to distinguish between "obvious" and "accidental" knowledge. Since the same M should work for all $\tilde{A}$, it makes little sense for M to try to analyze $\tilde{A}$'s program or to meddle with its tapes. In fact, for our purposes it suffices to give M the power to reset and rerun $\tilde{A}$ polynomially many times without inspecting or modifying its tapes.
4. The justification of interactive proofs of knowledge is purely statistical: The success of a particular execution of $(\tilde{A}, \overline{B})$ is a circumstantial evidence that the success rate of $(\tilde{A}, \overline{B})$ on the same input I and random tape RA is non negligible, and thus that the knowledge of S can be effectively demonstrated with overwhelming probability. However, the assumption about the probability distribution is a global property which is not really testable by $\overline{B}$, and thus no firm consequence about $\tilde{A}$'s knowledge should be directly provable from $\overline{B}$'s decision to accept $\tilde{A}$'s proof.

GMR's definition of zero knowledge can be stated in the following informal way:

An interactive proof system of membership is zero knowledge if for all $\tilde{B}$ its view of the communication in $(\overline{A}, \tilde{B})$ can be recreated with an indistinguishable probability distribution *under the sole assumption that S exists* by a polynomial time probabilistic Turing machine M.

As demonstrated in this paper, proofs of knowledge make perfect sense even when the set of I for which S exists is polynomially recognizable (and in particular when S always exists). We exploit this extra degree of freedom by making our definition of zero knowledge slightly stronger than the GMR definition:

Definition: An interactive proof system of knowledge is zero knowledge if for all $\tilde{B}$ its view of the communication in $(\overline{A}, \tilde{B})$ can be recreated with an indistinguishable probability distribution *under no additional assumptions* by a polynomial time probabilistic Turing machine M (which cannot access the oracle tape).

Remarks:

1. It is easy to show that a predicate $P(I, S)$ can have a zero knowledge interactive proof system of knowledge only when the set of I for which S exists is recognizable in random polynomial time (otherwise the existence of S which is implied by its knowledge is the result of an infeasible computation, and thus constitutes knowledge).
2. To prove that a particular proof system is zero knowledge in our model we use GMR's idea of resetting the simulation whenever it gets stuck. It is interesting to notice that in proofs of knowledge the resettable simulation is used in two very different ways: In proofs of validity we use it to extract from $\tilde{A}$ as much information as possible about S, while in proofs of zero knowledge we use it to avoid situations in which $\tilde{B}$'s questions cannot be answered. In proofs of membership the validity is based on $\tilde{A}$'s inability to satisfy $P(I, S)$, and the resettable simulation is used only to prove the zero knowledge character of the protocol.

Theorem 1: Any problem in NP has an interactive proof system of knowledge which is zero knowledge under GMR's definition (but not necessarily under our definition).

Proof (sketch): Due to the constructive nature of all the NP reductions, it suffices to consider Blum's zero knowledge interactive proof system of membership for graph Hamiltonicity. The details of this proof system are not really important, but basically it repeats $|I|$ times a basic step in which A transforms I into a complementary pair of subproblems, and $\overline{B}$ requests and verifies the solution to one of them. In most cases a valid Hamiltonian cycle S can be constructed from pairs of complementary solutions. The only exception is when $\tilde{A}$ manages to convince $\overline{B}$ that an improperly chosen encryption method is valid, but this event has an exponentially small probability.

The executions of $(\tilde{A}, \overline{B})$ on input I and random tape RA can now be described as an incomplete binary tree which describes $\tilde{A}$'s responses to $\overline{B}$'s requests. Each choice of RB corresponds to a particular path in the tree, where left sons correspond to subproblems of the first type and right sons correspond to subproblems of the second type. Any son that corresponds to a problem $\tilde{A}$ does not answer properly is eliminated along with all its descendants, and thus the successful executions correspond to root-to-leaf paths in the full binary tree that survive the truncation.

The machine M we construct explores this tree by repeatedly resetting $\tilde{A}$ to the root, providing the necessary steering requests, and verifying which one of the two sons of each explored vertex corresponds to a correct answer. It is easy to see that if the truncated tree contains any vertices with degree 2, M can find at least one such vertex in $O(|I|^2)$ time. Since a non negligible fraction of the exponentially many leaves of the full binary tree survive by assumption, the polynomial time M is guaranteed to succeed.

To complete the proof, we notice that M's success implies that either S has been computed or that $\tilde{A}$ used an improper encryption scheme. Since the probability of the latter event is negligible, the probability of the former event is overwhelming. Q.E.D.

Theorem 2: Any problem in NP∩co-NP has a zero knowledge interactive proof system of knowledge.

Proof: Since the problem is in NP, it can be represented by the boolean satisfiability formula $P1(I,S)$. Since the problem is also in co-NP, its complement can be represented by the boolean satisfiability formula $P2(I,S)$. Consider now the boolean predicate $P(I,S) = P1(I,S) \vee P2(I,S)$, which is satisfiable for all I. By Theorem 1 its Blum proof system (which makes no sense when used to prove the existence of S) is a valid interactive proof system of knowledge. Since the assumption that S exists is superfluous, the two definitions of zero knowledge coincide, and thus A can prove that he knows whether I is in the language or in its complement without revealing even this single bit of knowledge. Q.E.D.

3. An efficient identification scheme

An Identification scheme is a protocol which enables party A to prove his identity polynomially many times to party B without enabling B to misrepresent himself as A to someone else. Identification schemes are closely related to the notion of digital signatures, but there are no messages judges and disputes: The proof of identity is either accepted or rejected in real time, and as a result the requested access or service is granted or withheld. This is one of the fundamental problems in cryptography, and it has numerous practical applications. The basic problem with most of the current identification techniques (ID cards, credit cards, computer passwords, PIN numbers, etc) is that A proves his identity by revealing a constant S in the form of a printed card or a memorized value. A sophisticated adversary who cooperates with a dishonest verifier B can use a xerox copy of the card or a recording of the secret value to misrepresent himself successfully as A at a later stage. The obvious solution is to use zero knowledge proofs of knowledge, which convince B that A knows S without revealing even a single bit of information about A's key.

Our goal in the next two sections is to develop a truly practical scheme, which can be implemented in software in a fraction of a second even on the weak microprocessors embedded in smart cards. For reasonable choices of the parameters, our scheme is about two orders of magnitude faster than RSA-based identification schemes. If factoring is difficult, the new scheme is provably secure in the strong sense of the Goldwasser, Micali and Rivest [1984] "paradoxical scheme".

The scheme assumes the existence of a trusted center whose sole purpose is to publish a modulus n which is the product of two large primes of the form $4r+3$. Such moduli (which are known as Blum integers) are used in a variety of cryptographic applications, and their most useful property is that -1 is a quadratic non-residue whose Jacobi symbol is $+1$ (mod n). After publishing n, the center can be closed since it has no further role in the protocol. Note that unlike the RSA scheme, everyone can use the same universal n, and no one should know its factorization.

The identification scheme is a special case of Theorem 2, in which $\overline{A}$ proves to $\overline{B}$ that he knows whether a certain number is a quadratic residue or a quadratic non residue mod n without revealing even this single bit of information. It is a slightly modified version of the identification scheme described in Fiat and Shamir [1986] (which leaked nothing but this bit). It incorporates several ideas from GMR's yet unpublished zero knowledge proof of quadratic residuosity and from GHY's "result indistinguishable" residuosity protocol, but it has much lower time and communication complexities. This efficiency is derived primarily from the use of the multiplicative properties of modular square roots to prove the simultaneous knowledge of the quadratic residuosity character of several numbers – something we do not know how to do with non number theoretic problems.

$\overline{A}$'s key generation protocol in the new scheme is:

1. Choose k random numbers $S_1, \ldots, S_k$ in Z_n.
2. Choose each I_j (randomly and independently) as $\pm 1/S_j^2$ (mod n).
3. Publish $I_1, \ldots, I_k$ and keep $S_1, \ldots, S_k$ secret.

The S_j (which are witnesses to the quadratic residuosity character of the I_j) are effectively hidden by the difficulty of extracting square roots mod n, and thus A can establish his identity by proving that he knows these S_j. By allowing I_j to be either plus or minus a square modulo a Blum integer, we make sure that I_j can range over all the numbers with Jacobi symbol $+1$ mod n and thus the S_j exist (from B's point of view) regardless of I_j's character, as required in zero knowledge proofs of knowledge. Our choice of I_j as $\pm 1/S_j^2$ rather than $\pm S_j^2$ has no theoretical significance, but optimizes the practical implementations of the scheme.

To generate and verify a proof of identity, the parties execute the following protocol:

Repeat steps 1 to 4 t times:

1. $\overline{A}$ picks a random R, and sends $X = \pm R^2$ (mod n).
2. $\overline{B}$ sends a random boolean vector $(E_1, \ldots, E_k)$.
3. $\overline{A}$ sends the value $Y = R \cdot \prod_{E_j=1} S_j$ (mod n).
4. $\overline{B}$ verifies that $X = \pm Y^2 \cdot \prod_{E_j=1} I_j$ (mod n).

Theorem 3: This protocol is a zero knowledge proof of knowledge of the S_j for $k = O(\log\log n)$ and $t = O(\log n)$.

Proof (sketch): To prove that $\overline{A}$'s proof always convinces $\overline{B}$, we evaluate the verification condition:

$$Y^2 \cdot \prod_{E_j=1} I_j = \left(R \cdot \prod_{E_j=1} S_j\right)^2 \cdot \prod_{E_j=1} I_j$$
$$= R^2 \prod_{E_j=1} (S_j^2 I_j) = \pm R^2 = \pm X \pmod{n}.$$

Next we show that whenever $\overline{B}$ accepts $\widetilde{A}$'s proof with non negligible probability, M can print out all the S_j with overwhelming probability. Let T be the truncated execution tree of $(\widetilde{A}, \overline{B})$ for input I and random tape RA. Unlike the tree in Theorem 1, $\overline{B}$ may ask $2^k = (\log n)^{O(1)}$ possible questions at each stage, and thus the vertices in T may have polynomially many sons in terms of $|I|$. A vertex is called "heavy" if its degree is larger than $2^k/2$ (i.e., if more than half the executions of $(\widetilde{A}, \overline{B})$ at this state are successful). Our goal in this part of the proof is to show that all the S_j can be computed from the sons of a heavy vertex and that a polynomial time M can find a heavy vertex in T with overwhelming probability.

Let V be any heavy vertex in T and let Q be the set of queries in the form of boolean vectors $(E_1, \ldots, E_k)$ which are properly answered by $\widetilde{A}$. It is easy to show that for any $1 \leq j \leq k$ a set Q of more than $2^k/2$ boolean vectors of length k must contain two vectors $(E'_1, \ldots, E'_k)$ and $(E''_1, \ldots, E''_k)$ in which $E'_j = 0$, $E''_j = 1$, and $E'_i = E''_i$ for all $i \neq j$. Since both queries were properly answered, the two verification conditions imply:

$$X' = \pm Y'^2 \cdot \prod_{E'_j=1} I_j \pmod{n}$$

and

$$X'' = \pm Y''^2 \cdot \prod_{E''_j=1} I_j \pmod{n}.$$

However, $\widetilde{A}$ must choose X before he obtains $\overline{B}$'s query, and thus $X' = X''$. By manipulating the equations we get

$$(Y''/Y')^2 = \pm 1/I_j \pmod{n},$$

and thus Y''/Y' is the desired S_j (recall that n is a Blum integer, and thus exactly one of $+1/I_j$ and $-1/I_j$ has a square root).

Next we show that at least half the vertices in at least one of the levels in T must be heavy. Let α_i be the ratio between the number of vertices at level $i+1$ and the number of vertices at level i in T. If $\alpha_i \leq (3/4)2^k$ for all $1 \leq i \leq t$, then the total number of leaves in T (which is the product of all these α_i) is bounded by $(3/4)^t 2^{kt}$, which is a negligible fraction of the 2^{kt} possible leaves. Since we assume that this fraction is polynomial, $\alpha_i > (3/4)2^k$ for at least one i, and thus at least half the vertices at this level must contain more than $2^k/2$ sons.

To actually find a heavy vertex in T, M chooses polynomially many random vertices at each level, and determines their degrees by repeated resets and executions of $\widetilde{A}$. To ensure a uniform probability distribution in spite of the uneven degrees of the vertices, M should explore random paths in the untruncated tree, and restart from the root whenever the path encounters an improperly answered query. Since a non negligible fraction of the leaves is assumed to survive the truncation, this blind exploration of T can be carried out in polynomial time.

The last part of the proof deals with the zero knowledge aspect of the protocol. $\widetilde{A}$ can easily cheat $\overline{B}$ with probability 2^{-k} per iteration by guessing the $(E_1, \ldots, E_k)$ vector, preparing $X = \pm R^2 / \prod_{E_j=1} I_j \pmod{n}$ in step 1, and providing $Y = R$ in step 3. By using GMR's idea of resettable simulation, M can mimic the communication in $(\overline{A}, \widetilde{B})$ with an indistinguishable probability distribution in $O(t \cdot 2^k)$ expected time, which is polynomial in $|I|$ by our assumptions on k and t. Since the existence of S_j which satisfy $I_j S_j^2 = \pm 1 \pmod{n}$ is guaranteed for any I_j with Jacobi symbol $+1$ and this property is checkable in polynomial time, the protocol is zero knowledge even in our stronger sense. Q.E.D.

4. The parallel version of the identification scheme

Interactive proofs are usually iterated versions of some basic protocol with a constant number of elementary operations and a constant probability of cheating. This can be done either by repeating the protocol as a whole (sequential execution) or by repeating each elementary operation by itself (parallel execution). Parallel executions change the direction of the communication only a constant number of times (which is preferable for theoretical as well as practical reasons), but unfortunately they are not zero knowledge for very subtle technical reasons.

The problem of sequential vs. parallel executions of protocols has attracted considerable attention, and was dubbed "the case where intuition fails" in the literature. The information $\widetilde{B}$ can compute as a result of his parallel interaction with $\overline{A}$ seems to be very specialized, and thus it is natural to speculate that parallel versions of zero knowledge protocols release only "unusable" information. The problem of course is how to formally distinguish between "useful" and "useless" information. Our approximation to this notion is:

Definition: The protocol $(\overline{A}, \overline{B})$ releases no transferable information if it succeeds with overwhelming probability but there is no coalition of $\widetilde{A}, \widetilde{B}$ with the property that after polynomially many executions of $(\overline{A}, \widetilde{B})$ it is possible to execute $(\widetilde{A}, \overline{B})$ with a non negligible probability of success.

In other words, transferable information enables $\widetilde{B}$ to repeat $\overline{A}$'s proof to someone else. This is the most natural use of the information that $\overline{A}$ may inadvertantly reveal in his proof, and its absence is a certain indication

of cryptographic strength. In the context of identification schemes this can be viewed as the formal definition of security, and thus the following result proves that the parallel version of our identification scheme (which is not zero knowledge) is provably secure:

Theorem 4: If factoring is difficult, the parallel version of the identification scheme releases no transferable information.

In fact, we can prove a stronger result which enables us to turn any assumed lower bound on the complexity of factoring n into proven lower bounds on the complexities of successful $\widetilde{A}$ and $\widetilde{B}$:

Theorem 5: If $(\widetilde{A}, \overline{B})$ can be executed with probability $\varepsilon > 2^{-kt+1}$ after e executions of $(\overline{A}, \widetilde{B})$, then n can be factored by a coalition of $\overline{A}$, $\widetilde{A}$, $\overline{B}$ and $\widetilde{B}$ in time $O(|\widetilde{B}|e + |\widetilde{A}|/\varepsilon)$ and constant probability.

Proof: In proofs of zero knowledge $\overline{A}$ knows the secret S_j values and thus it is essential to simulate its role by an external M with indistinguishable probability distribution. This simulation is possible only when $k = O(\log \log n)$ and the execution of the protocol is sequential. We sidestep all these complications by changing the goal from the computation of the S_j to the factorization of n. Since $\overline{A}$ does not know this factorization, it cannot inadvertantly leak it to $\widetilde{B}$ during the executions of $(\overline{A}, \widetilde{B})$, and thus $\overline{A}$ can join the coalition of $\widetilde{A}$, $\overline{B}$ and $\widetilde{B}$ in their search for the factorization of n.

Given any pair of unusually successful programs $\widetilde{A}$ and $\widetilde{B}$, we start the factorization by executing $(\overline{A}, \widetilde{B})$ e times and relaying the transcript of the communication to $\widetilde{A}$. Since $\overline{A}$ itself can be used in this part and its time complexity $|\overline{A}|$ is assumed to be dominated by $|\widetilde{B}|$, these executions require $O(|\widetilde{B}|e)$ time.

The possible outcomes of the executions of $(\widetilde{A}, \overline{B})$ at this stage can be summarized in a large boolean matrix H whose rows correspond to all possible choices of RA, its columns correspond to all the 2^{kt} possible choices of RB, and its entries are 0 if $\overline{B}$ rejects $\widetilde{A}$'s proof and 1 if $\overline{B}$ accepts $\widetilde{A}$'s proof. Note that this value is well defined since the executions become deterministic once RA and RB are chosen.

To factor n, the coalition tries to find two 1's along the same row in H. They can probe H by executing $(\widetilde{A}, \overline{B})$ with properly chosen RA and RB tapes, and their goal is to minimize the number of probes.

The fraction of 1's in H is at least ε, but their locations can be chosen by an adversary who knows the probing strategy and tries to foil it. We call a row "heavy" if the fraction of 1's along it is at least $\varepsilon/2$. Since the width of H is 2^{kt} and $\varepsilon > 2^{-kt+1}$, a heavy row contains at least two 1's.

The obvious probing strategy uses $O(1/\varepsilon^2)$ probes in the following way:

1. choose $O(1/\varepsilon)$ random rows.
2. Probe $O(1/\varepsilon)$ random entries in each row.

Since the fraction of heavy rows in H is at least $\varepsilon/2$, the first step chooses at least one heavy row with constant probability and the second step finds two 1's along this row with constant probability. However, a better probing strategy is:

1. Probe $O(1/\varepsilon)$ random entries in H.
2. After the first 1 was found, probe $O(1/\varepsilon)$ random entries along the same row.

Since at least half the 1's in H are located in heavy rows, this strategy succeeds with constant probability in just $O(1/\varepsilon)$ probes. Again we assume that $|\overline{B}|$ is dominated by $|\widetilde{A}|$, and thus the time complexity of this part of the algorithm is $O(|\widetilde{A}|/\varepsilon)$.

The two successes found by this probing strategy are located in the same row, and thus $\widetilde{A}$ uses the same set of X's but manages to answer two different queries. As shown in the proof of Theorem 3, this enables $\widetilde{A}$ to get an equality

$$Y'^2 \cdot \prod_{E'_j=1} I_j = \pm Y''^2 \cdot \prod_{E''_j=1} I_j \pmod{n}$$

which is equivalent to

$$(Y'/Y'')^2 = \pm \prod_j I_j^{c_j} \pmod{n}$$

where $c_j \in \{-1, 0, +1\}$ and not all of them are zero.

$\widetilde{A}$'s ability to compute the square root of $Q = \pm \prod I_j^{c_j} \pmod{n}$ does not imply that $\widetilde{A}$ can factor n, since $\widetilde{A}$ was assisted by $\widetilde{B}$ who communicated with $\overline{A}$ who knew this square root in a protocol $(\overline{A}, \widetilde{B})$ which is not necessarily zero knowledge. To complete the proof we show that the square roots of Q which are known by $\overline{A}$ and computed by $\widetilde{A}$ are totally independent, and thus the *coalition* of $\overline{A}$ and $\widetilde{A}$ can factor n with probability 1/2.

Each $\pm I_j$ has four square roots mod n, but only two of them are known to $\overline{A}$ (otherwise it could have factored n by itself). We claim that even an infinitely powerful $\widetilde{B}$ cannot determine from the X's and Y's sent by $\overline{A}$ during the execution of $(\overline{A}, \widetilde{B})$ which square roots $\overline{A}$ actually uses. To prove this, consider the defining equation for Y:

$$Y = R \cdot \prod_{E_j=1} S_j \pmod{n}.$$

If $\overline{A}$ replaces S_j by one of the other three square roots, the Y is multiplied by one of the three non trivial square roots of 1. This effect can be cancelled by dividing R by the same square root, which leaves $X = \pm R^2 (\mathrm{mod}\ n)$ unchanged. Since the R's are randomly chosen. $\overline{A}$ produces the same X, Y values with the same probability distribution in both cases. This symmetry argument proves that $\overline{A}$ cannot leak to $\widetilde{B}$ during the executions of $(\overline{A}, \widetilde{B})$ which square root of Q he can compute from the S_j he knows. Q.E.D.

Theorem 5 proves that the identification scheme posseses a sharp security threshold: Anyone can misrepresent himself with probability 2^{-kt}, but no one can double it without factoring n. The factor of 2 is arbitrary, and can be replaced by any constant larger than 1. The non asymptotic nature of Theorem 5 makes it possible to prove the practical security of schemes for fixed k and t. For example, when $kt = 20$ and the running times of $\widetilde{A}$ and $\widetilde{B}$ are bounded by 2^{80}, $\widetilde{A}$ cannot misrepresent itself with probability better than 2^{-20} after $\widetilde{B}$ has verified 2^{20} proofs of identity unless n can be factored in 2^{100} steps.

The choice of $kt = 20$ can be obtained with $k = 5$ keys and $t = 4$ iterations. The expected number of modular multiplications per party is $(k + 1)t/2 = 12$, which compares very favorably with the 1000 modular multiplications required in the RSA scheme with a 200 digit modulus. The 2^{-20} security level suffices to deter cheaters in most practical applications: No one will try to pay with a forged credit card or try to enter a restricted area with a forged ID badge if he knows that his probability of success in each attempt is only one in a million, and a failed attempt will have unpleasant consequences.

An interesting modification can eliminate the public key directory and lead to a "keyless" identification scheme. It assumes that the trusted center (which knows the factorization of n) issues smart cards to users after properly checking their physical identity. No further interaction with the center is required either to generate or to verify proofs of identity. In this version of the scheme the center creates a string I which contains the user's name, address, ID number, physical description, and any other information provers or verifiers may want to establish. The public keys I_j are then derived pseudo randomly from I, and the secret square roots S_j are computed and stored in the card by the center. When A wants to prove his identity, B can derive the I_j directly from A's claimed identity rather than from a public key directory. The actual proof remains the same, and it convinces B that A knows the S_j that correspond to these I_j. These values could only be computed by the trusted center when the real A requested a card. This convenient scheme offers provable security for everyone involved: Provers cannot cheat verifiers, verifiers cannot later misrepresent themselves as provers, and even coalitions of provers and verifiers cannot create new identities, modify existing identities, or find out the secret factorization of n.

More information on the practical aspects of the scheme and its optimized implementations can be found in Fiat and Shamir [1986].

Acknowledgements: We would like to thank the anonymous readers of an earlier version of this paper for being so counterproductive.

Bibliography

1. Brassard, G. and C. Crepeau, "Nontransitive Transfer of Confidence: A Perfect Zero Knowledge Interactive Protocol for SAT and Beyond", Proceedings of FOCS 1986, pp. 187–195.
2. Chaum, D., "Demonstrating That a Public Predicate Can Be Satisfied Without Revealing Any Information About How", Proceedings of CRYPTO 1986.
3. Chor, B., S. Goldwasser, S. Micali and B. Awerbuch, "Verifiable Secret Sharing and Achieving Simultaneity in the Presence of Faults", Proceedings of FOCS 1985, pp. 383–395.
4. Fiat, A. and A. Shamir, "How To Prove Yourself: Practical Solutions to Identification and Signature Problems", Proceedings of CRYPTO 1986.
5. Galil, Z., S. Haber and M. Yung, "A Private Interactive Test of a Boolean Predicate and Minimum-Knowledge Public Key Cryptosystems", Proceedings of FOCS 1985, pp. 360–371.
6. Goldreich, O., S. Micali and A. Wigderson, "Proofs That Yield Nothing But Their Validity and a Methodology of Cryptographic Protocol Design", Proceedings of FOCS 1986, pp. 174–187.
7. Goldwasser, S., S. Micali and C. Rackoff, "Knowledge Complexity of Interactive Proof Systems", Proceedings of STOC 1985, pp. 291–304.
8. Goldwasser, S., S. Micali and R. Rivest, "A Paradoxical Solution to the Signature Problem", Proceedings of FOCS 1984, pp. 441–448.
9. Goldwasser, S. and M. Sipser, "Arthur Merlin Games versus Interactive Proof Systems", Proceedings of STOC 1986, pp. 59–68.
10. Halpern, J. and Y. Moses, "Knowledge and Common Knowledge in a Distributed Environment", Proceedings of PODC 1984, pp. 50–61.
11. Rosenschein, S., "Formal Theories of Knowledge in AI and Robotics", New Generation Computing 3, 1985, pp. 345–357.