Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract)

*Manuel Blum**
Computer Science Dept.
Univ. of Calif.
Berkely, CA

Paul Feldman
MIT Lab. for Computer Sci
Cambridge, MA

Silvio Micali†
MIT Lab. for Computer Sci
Cambridge, MA

Abstract

We show that interaction in *any* zero-knowledge proof can be replaced by sharing a common, short, random string. We use this result to construct the *first* public-key cryptosystem secure against chosen ciphertext attack.

1 Introduction

Recently [GMR] have shown that it is possible to prove that some theorems are true without giving the slightest hint of why this is so. This is rigorously formalized in the somewhat paradoxical notion of a *zero-knowledge proof system*

If secure encryption schemes exist, though, these proof systems are far from being a rare and bizar event. In fact, under this assumption, [GMW] demonstrate that any language in NP possesses zero-knowledge proof systems.

*Supported by NSF Grant # DCR85-13926
†Supported by ARO Grant # DAAL03-86-K-0171

Actually, as recently pointed out by Ben-Or, Goldreich, Goldwasser, Hastad, Micali and Rogaway [BGGHMR], the same is true for all languages in IP; also, as pointed out by Blum [B2], any theorem at all admits a proof that conveys zero-knowledge other than betraying its own length.

Zero-knowledge proofs have proven very useful both in complexity theory and in cryptography. For instance, in complexity theory, via results of fortnow [F] and Boppana and Hastad [BH], zero-knowledge provides us an avenue to convince ourselves that certain languages are not NP-complete. In cryptography, zero-knowledge proofs have played a major role in the recently proven completeness theorem for protocols with honest majority [GMW2]. They also have inspired rigorously-analyzed identification schemes [FFS] that are as efficient as folklore ones.

Despite its wide applicability, zero-knowledge remains an intriguing notion: What makes zero-knowledge proofs work?

Three main features differentiate all known zero-knowledge proof systems from more traditional ones:

1. *Interaction:* The prover and the verifier talk back and forth
2. *Hidden Randomization:* The verifier tosses coins that are hidden from the prover and thus unpredictable to him.
3. Computational Difficulty: The prover imbeds in his proofs the computational difficulty of some other problem.

At a first glance, all of these ingredients appear to be necessary. This paper makes a first, important step in distilling what is essential in a zero-knowledge proof. We show that computational diffi-

culty alone (for instance the hardness of distinguishing products of 2 primes from products of 3 primes) may make *inessential* the first resource (interaction) and and *eliminate the secrecy* of the second resource (randomness). That is, if the prover and the verifier share a common random string, the prover can non-interactively and yet in zero-knowledge convince the verifier of the validity of any theorem he may discover. A bit more precisely, for any constants c and d, sharing a k-bit long random string allows a prover p to prove in zero-knowledge to a poly(k)-time verifier V any k^c theorems of k^d size non-interactively; that is, without ever reading any message from V.

A Conceptual Scenario: Think of P and V as two mathematicians. After having played "heads and tails" for a while, or having both witnessed the same random event, P leaves for a long trip along the world, during which he continues his mathematical investigations. whenever he discovers a theorem, he writes a postcard to v proving the validity of his new theorem in zero-knowledge. Notice that this is necessarily a non-interactive process; better said, it is a mono-directional interaction: From P to V only. in fact, even if V would like to answer or talk to P, he couldn't: P has no fixed (or predictable) address and will move away before any mail can reach him.

1.1 Our Model Versus the Old One

While the definition of zero-knowledge remains unchanged, the mechanics of the computation of the prover and verifier changes dramatically.

Notice that sharing a random string σ is a weaker requirement than being able to interact. In fact, if P and V could interact they would be able to construct a common random string by coin tossing over the phone [B1]; the converse, however, is not true.

Also notice that sharing a common random string is a requirement *even weaker* than having both parties access a random beacon in the rabin's sense (e.g., the same geiger counter). In this latter case, in fact, all made coin tosses would be seen by the prover, but the future ones would still be unpredictable to him. by contrast, our model allows the prover to see in advance all the coin tosses of the verifier. That is the zero-knowledgeness of our proofs does not depend on the secrecy, or unpredictability of σ, but on the "well mixedness" of its bits! This curious property makes our result potentially applicable. For instance, all libraries in the country possess identical copies of the random tables prepared by the rand corporation. Thus, we may think of ourselves as being already in the scenario needed for non-interactive zero-knowledge proofs.

1.2 The Robustness of Our Result

As we have already said, we guarantee that all theorems proved in our proof systems are correct and zero-knowledge if the string σ is a truly random one. We may rightly ask what would happen if σ was not, in fact, truly randomly selected. fortunately, the poor randomness of σ may upset the zero-knowledgeness of our theorems, but not their correctness. That is, for almost all (poorly random) σ's, there is no wrong statement that can be accepted by the verifier. This is indeed an important property as we can never be sure of the quality of our natural sources of randomness. Unfortunately, due to the limitations of an extended abstract, we cannot further elaborate on this and similar points. We wish, however, to point out the following important corollary of our result.

1.3 Applications of our Result

A very noticeable application of non-interactive zero-knowledge is the construction of encryption schemes á la diffie and hellman that are secure against chosen ciphertext attacks. Whether such schemes existed has been a fundamenatal open problem ever since the appearence of complexity-based cryptography. We will discuss this application in Section 3.

1.4 What's Coming

The next section is devoted to set up our notation, recall some elementary facts from Number Theory and state the complexity assumption which sufficies to show the existence of non-interactive, zero-knowledge proofs.

In Section 3, we show that if a k^4-bit string is randomly selected and given to both the proven and the verifier, then the first can prove in zero-knowledge, for any single string x (of length k) belonging, to a NP-language L, that indeed $x \in L$.

Only in Section 4 we show that, for each fixed polynomial $Q(\cdot)$, using the *same* randomly chosen k^4-bit string, the prover can show in zero-knowledge membership in NP languages for any $Q(k)$ strings of length $Q(k)$.

2 Preliminaries

2.1 Notations and Conventions

Let us quickly recall the standard notation of [GoMiRi].

We emphasize the number of inputs received by an algorithm as follows. If algorithm a receives only one input we write "$A(\cdot)$", if it receives two inputs we write "$A(\cdot,\cdot)$" and so on.

If $A(\cdot)$ is a probabilistic algorithm, then for any input x, the notation $A(x)$ refers to the probability space that assigns to the string σ the probability that A, on input x, outputs σ. If S is a probability space, then $PR_S(e)$ denotes the probability that S associates with the element e.

If $f(\cdot)$ and $g(\cdot,\ldots,\cdot)$ are probabilistic algorithms then $f(g(\cdot,\ldots,\cdot))$ is the probabilistic algorithm obtained by composing f and g (i.e. running f on g's output). For any inputs $x, y, \ldots$ the associated probability space is denoted by $f(g(x, y, \ldots))$.

If s is any probability space, then $x \leftarrow S$ denotes the algorithm which assigns to x an element randomly selected according to S. If f is a finite set, then the notation$x \leftarrow f$ denotes the algorithm which assigns to x an element selected according to the probability space whose sample space is f and uniform probability distribution on the sample points.

The notation $Pr(x \leftarrow S; y \leftarrow T; \ldots : p(x, y, ..))$ denotes the probability that the predicate $p(x, y, \ldots)$ will be true after the ordered execution of the algorithms $x \leftarrow S$, $y \leftarrow T, \ldots$

The notation $\{x \leftarrow S; y \leftarrow T; \ldots : (x, y, \ldots)\}$ denotes the probability space over $\{(x, y, \ldots)\}$ generated by the ordered execution of the algorithms $x \leftarrow S$, $y \leftarrow T, \ldots$.

Let us recall the basic definitions of [GMR]. We address the reader to the original paper for motivation, interpretation and justification of these definitions.

Let $U = \{U(x)\}$ be a family of random variables taking values in $\{0,1\}^*$, with the parameter x ranging in $\{0,1\}^*$. $U = \{U(x)\}$ is called poly-bounded family of random variables, if, for some constant $e \in \backslash$, all random variables $U(x) \in u$ assign positive probability only to strings whose length is exactly $|x|^e$.

Let $C = \{C_x\}$ be a poly-size family of boolean circuits, that is, for some constants $c, d > 0$, all C_x have one boolean output and at most $|x|^c$ gates and $|x|^d$ inputs. In the following, when we say that a random string, chosen according to $U(x)$, where $\{U(x)\}$ is a poly-bounded family of random variables, is given as input to C_x, we assume that the length of the strings that are assigned positive probability by $U(x)$ equals the number of boolean inputs of C_x.

Definition 2.1 (Indistinguishability) . *Let $L \subset \{0,1\}^*$ be a language. Two poly-bounded families of random variables $U = \{U(x)\}$ and $V = \{V(x)\}$ are indistinguishable on L if for all poly-size families of circuits $C = \{C_x\}$,*

$$\Big|Pr(A \leftarrow U(x) : C_x(a) = 1) - Pr(a \leftarrow V(x) : C_x(a) = 1)\Big| < |x|^c$$

For all positive constants c and sufficiently large $x \in L$.

Definition 2.2 (Approximability) . *Let $L \subset \{0,1\}^*$ be a language. a family of random variables $U = \{U(x)\}$ is approximable on L if there exists a probabilistic turing machine M, running in expected polynomial time, such that the families $\{U(x)\}$ and $\{M(x)\}$ are indistinguishable on L.*

2.2 Number Theory

Let $Z_s(k)$ denote the set of integers product of $s \geq 1$ distinct primes of length k.

Let N be the set of the natural numbers, $x \in N$, $Z_x^* = \{y \mid 1 \leq y < x, \ gcd(x, y) = 1\}$ and $Z_x^{+1} = \{y \in Z_x^* | (y \mid x) = +1\}$, where $(y \mid x)$ is the jacobi symbol. We say that $y \in Z_x^*$ is a quadratic residue modulo x iff there is $w \in Z_x^*$ such that $w^2 \equiv y \bmod x$. If this is not the case we call w a quadratic non residue modulo x.

Define the quadratic residuosity predicate to be

$$Q_x(y) = \begin{cases} 0, & \text{if } y \text{ is a quadratic residue modulo } x; \\ 1, & \text{otherwise;} \end{cases}$$

and the languages QR and QNR as

$$QR = \{(y, x) | Q_x(y) = 0\}$$

$$QNR = \{(y, x) | y \in Z_x^{+1} \text{ and } Q_x(y) = 1\}.$$

Fact 1: Let $\sim$ be the relation so defined: $y_1 \sim y_2$ iff $Q_x(y_1 y_2) = 0$. Then $\sim$ is an equivalence relation in Z_x^{+1}. Two elements are equivalents if they have the same quadratic character modulo each of the prime divisors of x. Thus, if $x \in Z_2(k)$ there are 2 equivalence classes, if $x \in Z_3(k)$ there are 4; in general if $x = p_i^{k_1} \cdots, p_n^{k_n}$ where each p_i is a prime > 2 and $p_i \neq p_i$ if $i \neq j$, then there are 2^n equivalence classes.

Fact 2: For each $y_1, y_2 \in Z_x^{+1}$ one has

$$Q_x(y_1 y_2) = Q_x(y_1) \oplus Q_x(y_2).$$

Fact 3: Where "$\oplus$" denotes the *exclusive or* operator. the jacobi symbol function $x|n$ is polynomial-time computable.

We now formalize the complexity assumption that is sufficient for non-interactive zero-knowledge. Namely, that it is computationally hard to distinguish the integers product of 2 primes leftarrow the ones product of 3 primes.

2.3 A Complexity Assumption

2OR3A: for each poly-size family of circuits $\{C_k | k \in N\}$

$$|P_{Z_2(k)} - P_{Z_3(k)}| < k^{-c}$$

for all positive constants c and sufficiently large k; where

$$P_{Z_2(k)} = P_R(x \leftarrow Z_2(k) : C_k(x) = 1) \text{ and}$$
$$P_{Z_3(k)} = P_R(x \leftarrow Z_3(k) : C_k(x) = 1).$$

2OR3A is a stronger assumption than assuming that deciding quadratic residuosity is hard. (Having an oracle for $Q_n(\cdot)$, allows one to prbabilistically count the number of $\sim$ equivalence in Z_n^{+1} and thus, by fact 1, to distinguish whether $n \in Z_2(k)$ or $n \in Z_3(k)$). Thus we can freely use that quadratic residuosity is computationally hard (as formalized below) without increasing our assumption set.

Quadratic Residuosity Assumption(QRA): For each poly-size family of circuits $\{C_k \mid k \in \mathcal{N}\}$,

$$Pr(x \leftarrow Z_2(k); y \leftarrow Z_x^{+1} : C_k(x, y) = Q_x(y)) < 1/2 + 1/k^{-O(1)}.$$

The QRA was introduced in [GM] and is now widely used in Cryptography. The current fastest algorithm to compute $Q_x(y)$ is to first factor x and then compute $Q_x(y)$, while it is well known that, given the factorization of x, $Q_x(y)$ can be computed in $O(|x|^3)$ steps. In what follows, we choose $x \in Z_2(k)$ since these integers constitute the hardest input for any known factoring algorithm.

3 Single-Theorem Non-Interactive Zero-Knowledge Proofs

To prove the existence of single-theorem Non-Interactive Zero-Knowledge Proof Systems (single-theorem non-interactive ZKPS) for all NP languages, it is enough to prove it for $3COL$ the NP-complete language of the 3-colorable graphs [GJ]. For $k > 0$, we define the language $3COL_k = \{x \in 3COL \mid |x| \leq k\}$.

Definition 3.1 . *A Single-Theorem Non-Interactive ZKPS is a pair (A,B) where A is a Probabilistic Turing Machine and $B(\cdot,\cdot,\cdot)$ is a deterministic algorithm running in time polynomial in the length of its first input, such that:*

1. **Completeness.** *(The probability of succeeding in proving a true theorem is overwhelming.)*

   $\exists c > 0$ such that $\forall x \in 3COL_k$

   $$Pr(\sigma \leftarrow \{0,1\}^{n^c}; y \leftarrow A(\sigma, x) : B(x, y, \sigma) = 1) > 1 - n^{-O(1)}.$$

2. **Soundness.** *(The probability of succeeding in proving a false theorem is negligible.)*

   $\exists c > 0$ such that $\forall x \notin 3COL_k$ and for each Probabilistic Turing Machine A'

   $$Pr(\sigma \leftarrow \{0,1\}^{n^c}; y \leftarrow A'(\sigma, x) : B(x, y, \sigma) = 1) < n^{-O(1)}.$$

3. **Zero-Knowledge.** *(The proof gives no information but the validity of the theorem.)*

   $\exists c > 0$ such that the family of random variables $V = \{V(x)\}$ is approximable over $3COL$. Where

   $$V(x) = \{\sigma \leftarrow \{0,1\}^{|x|^c}; y \leftarrow A(\sigma, x) : (\sigma, y)\},$$

Remark: Notice that, as usual, the zero-knowledge condition guarantees that the verifier's *view* can be well simulated; that is, all the verifier may see can be reconstructed with essentially the same odds. In our scenario, what the verifier sees is only the common random string and the proof, i.e., the string, received by A. Notice that in our scenario, the definition of zero-knowledge is simpler. As there is no interaction between B and A, we do not have to worry about possible cheating by the verifier to obtain a "more interesting view." That is, we can eliminate the quantification "$\forall B'$" from the original definition of [GMR].

Theorem 3.1 . *Under the QRA, there exists a Single-Theorem Non-Interactive ZKPS for 3-COL.*

This theorem will be rigorously proven in the final paper. Here we restrict ourselves to informally describe the programs P and V of a single-theorem non-interactive ZKPS (P,V) and, even more informally, to argue that they posses the desired properties.

3.1 The Proof-System (P,V)

Instructions for P

1. Randomly select $n_1, n_2, n_3 \in Z_2(k)$
2. For $i = 1, 2, 3$ randomly select q_i such that $(q_i|n_i) = 1$ and q_i is a quadratic non-residue mod n_i.
3. Color G with colors 1,2,3.

4. For each node v of G whose color is i, label v with a randomly selected triplet $(v_1, v_2, v_3) \in Z^{+1}_{n_1} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$ such that $Q_n(v_i) = 0$ and $Q_{n_j}(v_j) = 1$ for $j \neq i$. Call G' the so labeled G

   {**Remark 1:** WLOG (else purge σ in the "right way") let $\sigma = \sigma_1 \circ \sigma_2 \circ \sigma_3 \circ \sigma_4, \cdots$, where all triplets $(\sigma_1, \sigma_2, \sigma_3)(\sigma_4, \sigma_5, \sigma_6), \cdots$ belong to $Z^{+1}_{n_1} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$.}

   {**Convention:** The first $8k$ triplets are assigned to the first edge of G (in the lexicographic order), the next $8k$ triplets to the second edge, and so on.}

5. For each edge (a, b) of G' (where node a has label (a_1, a_2, a_3) and node b (b_1, b_2, b_3)) and each of its $8k$ assigned triplets (z_1, z_2, z_3) compute one of the following types of *signature*.

   {**Comment:** Only one is applicable if steps 1-4 are performed correctly)}

$(\sqrt{z_1}, \sqrt{z_2}, \sqrt{z_3})$	type 0
$(\sqrt{q_1 z_1}, \sqrt{z_2}, \sqrt{z_3})$	type 1
$(\sqrt{z_1}, \sqrt{q_2 z_2}, \sqrt{z_3})$	type 2
$(\sqrt{z_1}, \sqrt{z_1}, \sqrt{q_3 z_3})$	type 3
$(\sqrt{a_1 z_1}, \sqrt{a_2 z_2}, \sqrt{a_3 z_3})$	type 4
$(\sqrt{b_1 z_1}, \sqrt{b_2 z_2}, \sqrt{b_3 z_3})$	type 5
$(\sqrt{a_1 b_1 z_1}, \sqrt{a_2 b_2 z_2}, \sqrt{a_3 b_3 z_3})$	type 6
$(\sqrt{q_1 z_1}, \sqrt{q_2 z_2}, \sqrt{q_3 z_3})$	type 7

   {**Notation "by example":** Let z_1 be a quadratic non residue mod n_1, z_2 a quadratic residue mod n_2, and z_3 is a quadratic residue mod n_2. Then the signature of the triplet (z_1, z_2, z_3) a triplet of type 1: $(\sqrt{q_1 z_1}, \sqrt{z_2}, \sqrt{z_3})$ where $\sqrt{q z_1}$ denotes a randomly selected square root of the quadratic residue $q_1 \cdot z_1 \bmod n_1$; and for $i = 2, 3$ $\sqrt{z_i}$ denotes a randomly selected square root of $z_i \bmod n_i$}

6. Send V $n_1, n_2, n_3, q_1, q_2, q_3, G'$, and the signature of the triplets composing σ.

   {**Comment:** Note that the edges of G' are labelled with triples, not with colors!}

Instructions for V

1. Verify that n_1, n_2, and n_3 are not even and not integer powers. Verify that G' is a proper labelling of G. That is, each node v has assigned a triplet (v_1, v_2, v_3) such that $v_i \in Z^{+1}_{n_i}$ for $i = 1, 2, 3$.

2. Break σ into triplets, verify that for each edge you received a signature of some type for each of its $8k$ triplets.

3. If all the above verifications have been successfully made, accept that G is 3-colorable.

3.2 A Rough Idea of why (P,V) is a Single-Theorem Non-Interactive ZKPS

First notice that the communication is monodirectional: From P to V. Then let us convince ourselves that the statement of Remark 1 really holds without loss of generality. In our context, WLOG means with overwhelming probability.

If G has a edges, our protocol assumes σ to consist of $8 \cdot k \cdot a$ triplets in $Z^{+1}_{n_1} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$. Such a string σ is easily obtainable from a (not too much larger) random string ρ. Consider ρ to be the concatentation of k-bit strings grouped into triplets

$$\rho = (\rho_1, \rho_2, \rho_3)(\rho_4, \rho_5, \rho_6) \cdots$$

Then obtain σ by "purging" ρ. That is, obtain σ from ρ by discarding all triplets not in $Z^{+1}_{n_1} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$. We now argue that ρ is not much longer than σ. Let n be either n_1 or n_2 or n_3. Now a random k-bit integer (with possible leading 0's) is less than n with probability $\geq \frac{1}{2}$; a random integer less than n belongs to Z^*_n with probability $\geq \frac{1}{2}$; a random element of Z^*_n belongs to Z^{+1}_n with probability $\geq \frac{1}{2}$. Thus, we expect that at least 1 in 64 of the triplets of ρ not to be discarded.

Now let us consider the question of V's running time. V can verify in poly-time whether $n_i = x^\alpha$ (where x, α integers; $\alpha > 1$) as only values $1, \cdots, \log n_i$ should be tried for α and binary search can be performed for finding x, if it exists. All other steps of V are even easier.

Now let us give some indication that (P,V) constitute a single-theorem non-interactive ZKPS.

Completeness: Assuming that σ is already consiting of triplets in $Z^{+1}_{n_1} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$, if P operates correctly, V will be satisfied with probability 1.

Soundness: If the verification step 1 is successfully passed, by fact 1, there must be $\geq 2 \sim$ equivalence classes in each $Z^{+1}_{n_i}$ (exactly two if P honestly chooses all the n_i's in $Z_2(k)$).

Thus, if we define two of our triplets (z_1, z_2, z_3) (w_1, w_2, w_3) to be equivalent if $z_i w_i \bmod n_i$ is a quadratic residue for $i = 1, 2, 3$, we obtain ≥ 8 equivalence classes among the triplets (exactly 8 if P is honest).

To exhibit a signature of a given type for a triplet, essentially means to put the triplet in one out of ≤ 8 possible "drawers". (there are 8 types of signatues, but they may not be mutually exclusive; thus two drawers may be equal). Moreover, it is easy to see that if two triplets are put in the same drawer, they must belong to the same equivalence class.

As σ is randomly selected, each of its triplets in $Z^{+1}_{n_i} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$ is equally likely to belong to any of

the ≥ 8 equally-numerous equivalence classes. However, since if there were > 8 classes, there would be (by fact 1) at least 16, the fact that all triplets can be fit in ≤ 8 drawers, "probabilistically proves" several facts:

1. There are exactly 8 equivalence classes among the triplets and exactly 8 distinct drawers.
2. The n_i's are product of two distinct prime powers.
3. $Q_{n_1}(q_1) = Q_{n_2}(q_2) = Q_{n_3}(q_3) = 1$
4. $Q_{n_1}(q_1) + Q_{n_2}(q_2) + Q_{n_3}(q_3) = 2$
   That is, (a_1, a_2, a_3) is a proper color (i.e., properly encodes a color: Either 1,2, or 3).
5. That (b_1, b_2, b_3) is a proper color.
6. That (a_1, a_2, a_3) and (b_1, b_2, b_3) are different colors. Else drawer 6 and drawer 0 would be the same.

Item 6 being true for all edges in G' implies that G is 3-colorable which is what was to be proven.

Zero-Knowledgeness

Let us specify the simulating machine M that, under the QRA, generates a pair (σ, proof) with the "right odds" on input G (without any coloring!)

Instructions for M

1. Randomly select $n_1, n_2, n_3, \in Z_2(k)$ together with their prime factorization.
2. Randomly select q_1, q_2, q_3 so that $Q_{n_1}(q_1) = Q_{n_1}(q_2) = Q_{n_3}(q_3) = 0$
3. For each node v of G, label v with a triplet $(v_1, v_2, v_3) \in Z^*_{n_1} \times Z^*_{n_2} \times Z^*_{n_3}$ such that $Q_{N_1}(v_1) = Q_{n_2}(v_2) = Q_{n_3}(v_3) = 0$. Call G' the so labelled graph.
4. Construct $\sigma = (\sigma_1, \sigma_2, \sigma_3)(\sigma_4, \sigma_5, \sigma_6)\cdots$, such that each triplet $(\sigma_{3j+1}, \sigma_{3j+2}, \sigma_{3j+3})$ is randomly selected so that $Q_{n_i}(\sigma_{3j+i}) = 0$ for $i = 1, 2, 3$.

   {**Remark:** Also in the simulation we only deal with already "purged strings". It is not hard to see that M could also handle generating "unpurged strings".}
5. For each edge (a, b) of G' and each of its assigned $8k$ triplets (z_1, z_2, z_3), choose an integer i at random between 0 and 7, and compute a signature of type i.

   {**Comment:** By using the prime factorization of the n_i.}
6. Output $\sigma, n_1, n_2, n_3, q_1, q_2, q_3, G'$, and the computed signatures.

We now informally argue that M is a good simulator for the view of V. Essentially, this is so because efficiently detecting that the triplets of σ are not randomly and independently drawn from the space $Z^{+1}_{n_1} \times Z^{+1}_{n_2} \times Z^{+1}_{n_3}$ is tantamount as violating the QRA (to be explained in the final paper). For the same reason, it cannot be detected efficiently that G' is an illegal labelling or that q_1, q_2, q_3 are squares mod, respectively, n_1, n_2, n_3. Given that, the distribution of the various types of signature looks "perfect".

{**Remark:** the reader is encouraged to verify that if (P,V) uses part of the used σ to show that another graph is 3-colorable, then extra knowledge would leek. For instance that there exists 3-coloring of G and H in which nodes v_1 and v_2 in H respectively have the same clolors as nodes w_1 and w_2 in G.}

4 Non-Interactive ZKPS

The Single-Theorem Non-Interactive ZKPS of Section 3 has a limited applicability. This is best illustrated in terms of our conceptual scenario where the prover P is leaving for a trip. It is unlikely that for each theorem T that P finds, a string σ_T comes from the sky "devoted" to T and is presented to (is read by) both P and V. It is instead more probable, that P and V have witnessed or generated (i.e., by flipping a coin), the same common random event of "size n" when they were together.

However, the Proof System of Section 3 will enable P to subsequently prove in Zero-Knowledge to V only a single theorem of size, smaller than n. He is out of luck should he discover the proofs of many theorems or of a theorem of bigger size.

This drawback is eliminated by the following notion of non-interactive ZKPS.

Our formal definition is slightly oriented towards our solution. Namely, at the beginning, independently of the theorems T_i's we care about, we let the prover choose a *random* theorem T and use the common string σ to compute a string y_o proving that T is true.

Subsequently, for each *desired* and important theorem T_i, the prover will produce a proof y_i. The correctness of y_i is checked by the verifier, not only on input T_i and σ but also y_o, the proof of the initial, random theorem.

This somewhat awkward mechanics, justified by the technical needs of our proof, does not change the rules of the game of our conceptual scenario in any essential way. In fact, notice in the definition that every important theorem is proven ALONE. That is, P

is able to select the zero-knowledge proof of each important theorem INDEPENDENTLY from the proofs or the statements of every other important theorem. In other words, P may have forgotten what important theorems he has already proved, and does not yet know what other important theorems he will discover: A true mathematician!

Only the proof, y_o, of the initial random theorem needs to be remembered. This random theorem and its proof are selected before and independently of every important theorem.

Definition 4.1 *A Non-Interactive ZKPS is a pair (P,V) where P is a pair, (P_0,P_1), of Probabilistic Turing Machines and $V(\cdot,\cdot,\cdot,\cdot)$ is a deterministic algorithm running in time polynomial in the length of its first input, such that:*

1. **(Completeness)** *For all polynomials P,Q, and for all $(x_1,x_2,\ldots,x_{Q(n)}) \in (3COL_{P(n)})^{Q(n)}$*

$$Pr(\ \sigma \leftarrow \{0,1\}^{n^{O(1)}};\ y_0 \leftarrow A_0(\sigma);\ y_1 \leftarrow A_1(\sigma,x_1,y_0);\ \vdots\ \ \vdots\ \ y_{Q(n)} \leftarrow A_1(\sigma,x_{Q(n)},y_0) : \bigwedge_{j=1}^{Q(n)} B(x_j,y_j,y_0,\sigma)=1\ ) > 1-n^{-O(1)}.$$

2. **(Soundness)** *For all polynomials P,Q, for all $(x_1,x_2,\ldots,x_{Q(n)}) \notin (3COL_{P(n)})^{Q(n)}$ and for each $A'=(A'_0,A'_1)$*

$$Pr(\ \sigma \leftarrow \{0,1\}^{n^{O(1)}};\ y_0 \leftarrow A'_0(\sigma);\ y_1 \leftarrow A'_1(\sigma,x_1,y_0);\ \vdots\ \ \vdots\ \ y_{Q(n)} \leftarrow A'_1(\sigma,x_{Q(n)},y_0) : \bigwedge_{j=1}^{Q(n)} B(x_j,y_j,y_0,\sigma)=1\ ) < n^{-O(1)}.$$

3. **(Zero-Knowledge)**
For each polynomial Q, the family of random variables $V=\{V(x_1,\ldots,x_{Q(n)})\}$, where

$$V(x_1,\ldots,x_{Q(n)}) = \{\ \sigma \leftarrow \{0,1\}^{n^{O(1)}};\ y_0 \leftarrow A_0(\sigma);\ y_1 \leftarrow A_1(\sigma,x_1,y_0);\ \vdots\ \ \vdots\ \ y_{Q(n)} \leftarrow A_1(\sigma,x_{Q(n)},y_0) : (\sigma,y_0,y_1,\ldots,y_{Q(n)})\ \}$$

is approximable over $\bigcup_n (3COL)^{Q(n)}$.

4.1 The Proof System (P,V)

Below *Gen* is a cryptographically strong pseudo-random bits generator [BM] [Y]. (Not to increase our assumptions, *Gen* could be the generator suggested in [BBS] that is based on quadratic residuosity , actually on factoring as shown in [ACGS].)

Common Inputs to P and V: A random string $\sigma \circ \rho$, a security parameter k, a sequence of 4-colorable graphs $G_1, G_2, \cdots$

Stage 1

P Chooses at random $n \in Z_3(k)$ and non-interactively, in zero-knowledge proves to V that indeed $n \in Z_3(k)$. P does so as in Section 3 by reducing the statement "$n \in Z_3(k)$" to the 3-colorability of an auxiliary graph H. P proves that H is 3-colorable by only using σ, the first segment of the common random string. **Remark:** This is another example of the fact that proving a more general theorem is easier. Here we only needed to prove membership in $Z_3(k)$. However, we were not able to find a direct non-interactive zero-knowledge proof of it. (What is easy by using a guaranteed random string, is proving membership in $Z_2(k)$ by sampling. Namely, by fact 1, to prove that $n \in Z_2(k)$ is enough to show that half of the elements in Z_n^{+1} are quadratic residues mod n). Only when we thought of generalizing the problem of membership in $Z_3(k)$ to the more general 3-colorability problem, we succeeded in proving the desired result.

Stage 2

For each input graph, $G \in 4-COL_k, G_1$ P's and V's programs are as follows:

Instructions for P

1. Number the equivalence classes of Z_n^{+1} 1 through 4.
2. Find a 4-coloring for G.
3. For any vertex v in G, if v is colored i, randomly choose an element e_v in class i and label v with e_v. Call G' the so labeled graph.

4. Send G' to V.

5. For each edge (u, v) in G, randomly choose $y_{uv} \in Z_n^{+1}$ so that $e_u \cdot e_v \cdot y_{uv}$ mod n is a square, compute a random square root of it, x_{uv} and send y_{uv} and x_{uv} to V.

6. for each y_{uv} do: Output the next k^h bits of *Gen* on input ρ (here h is a constant to be determined later); group these bits into consecutive *blocks* of k bits each; consider all blocks that represent elements in Z_n^{+1}; for each block representing a square mod n, send a random square root of it to V; for each block that is in the same $\sim$ equivalence class as y_{uv}, send V a square root of its product with y_{uv}.

Instructions for V

1. Check that all labels of G are Jacobi symbol 1 elements of Z_n^*.

2. For all edges (u, v), check that x_{uv} is a square root of $e_u \cdot e_v \cdot y_{uv}$ mod n

3. For each y_{uv}, check to have received correcrt square roots for more than $\frac{k^h}{5}$ of its associated blocks and for more than $\frac{k^h}{5}$ other blocks "times" y_{uv}.

4. If all checks are passed "accept" that G is 4-colorable.

4.2 The Zero-Knowledgeness of (P,V)

We now *very informally* argue that (P,V) is a non-interactive ZKPS.

First notice that the communication is mono-directional: from P to V. Second that all of V's computation can be done in probabilistic polynomial time.

Completeness

If G is 4-colorable and P and V follow their instructions, V will accept with probability essentially 1. The reader can easily derive a proof of it. (Reading the ideas of the proof about the soundness property may help.)

Soundness

Since n has passed stage 1 successfully, with probability essentially 1 it is the product of three distinct primes[1]. (All modular operations mentioned are mod n)

Second, with probability essentially equal to 1, the sequence of blocks associated with each y_{uv} contains more than $\frac{k^h}{5}$ elements in each of the 4 equivalence classes. In fact, a random sequence of Jacobi symbol 1 elements mod n would "visit" each class with probability 1/4. This is also true for the output of *Gen* since it is poly-indistinguishable from a truly random sequence[2] If $\frac{k^h}{5}$ blocks "times" y_{uv} have square roots mod n, then they all belong to the same equivalence class as y_{uv}. Moreover, if another $\frac{k^h}{5}$ elements have square roots mod n by themselves, then, with probability essentially 1, y_{uv} is a non-square mod n. (Otherwise $\frac{2 \cdot k^h}{5}$ blocks would be squares mod n rather than the expected $\frac{k^h}{4}$.) Finally, if y_{uv} is a non-square mod n, then the edge (u,v) is properly colored; that is, e_u and e_v belong to different classes. In fact, since $e_u \cdot e_v \cdot y_{uv}$ has a square root, $(e_u \cdot e_v)$ belongs to the same class as y_{uv}; and if e_u and e_v belonged to the same class, their product would be a square and so would y_{uv}. Each edge being correctly colored, so is G.

Zero-Knowledgeness

We must now argue that the above proof system is zero-knowledge. That is, that there exists an efficient simulator that, given any sequence of 4-colorable graphs (but not their colorings!), a probability distribution on the pairs (σ, proof,) that is computationally indistinguishable from the one V^{prime} would "see" if listening to P. What V' would see is stage 1 and in stage 2, messages from P about each input graph. The proof of zero knowledge is quite delicate. We restrict ourselves to merely outlining its high level steps, without further details. We do point out, though, which parts of the proof are easy and which are hard.

4.2.1 The Simulation of Stage 1

The first message a verifier receives from P is a random member of $Z_3(k)$. The simulating machine M, instead, randomly generates two primes and multiplies them together to generate a random member of $Z_2(k)$. So far, because of the 2or3A, this will fool any polynomially-bounded judge.

Then M in a standard way constructs a graph H that is 3-colorable if and only if $n \in Z_3(k)$.

Since the latter statement is false, H *will not* be 3-colorable. Nonetheless, M follows the protocol described in Section 3 where H is the input graph.

Given the 2or3A, the distribution so obtained is polynomial-time indistinguishable from random a correct execution of Stage 1 (including the choice of σ)!

[1] n is or is not product of 3 primes wheter or not it passed the first stage; but you know what I mean and it is easier to read!

[2] A subtle point: this is so even if in our application *Gen*'s seed, ρ, is not secret. In fact, all efficiently checkable statistical properties hold for *Gen*'s output if the random seed is kept secret, and the particular statistical property of interest to us cannot "disappear" if the seed is made public!

This may appear paradoxical,. How can M generate such an "indistinguishable" distribution on input $n \in Z_2(k)$, if, after all, P's message (which is an integral part of V's view) was *proving* that $n \in Z_3(k) \neq Z_2(k)$?

The paradox disappears when we consider that P's message was convincing since the random choice of σ was not under its control. In the simulation, instead, M chooses σ![3]

In fact, in stage 2, the simulator will label all vertices of any graph G by squares mod n ($\in C_2(k)$). That is, to each vertex u he associates a randomly selected square e_u. (No efficient judge may reject this labelling, since the hardness of quadratic residuosity implied by our assumption.) Then, to each edge (u, v), he associates a randomly selected square y_{uv}. Now the simulator correctly runs *Gen* on its random seed to obtain a pseudo-random k^h-long block-sequence. Roughly half of the elements of Jacobi symbol 1 of these blocks will be squares mod n, as n is the product of 2 primes. For a randomly selected half of them the simulator will extract a square root, which it can easily done as he chose n in factored form. For each block in the remaining half, he extracts a square root of its product with y_{uv}. Again this will fool the judge as he cannot efficiently decide quadratic residuosity.

Notice that faking the proof of a single theorem (membership in Z_3^k) allowes us to fake the proof of an arbitrary number of other theorems. This is one of the reasons to choose the computational difficulty of distinguishing products of 2 primes from products of 3 primes.

5 A No-Longer Long-Standing Open Problem

One of the most beautiful gifts of complexity-based cryptography is the notion of a public-key cryptosystem. As proposed by Diffie and Hellman [DH], each user U publicizes a string P_U and keeps secret an associated string S_U. Another user, to secretely send a message m to U, computes $y = E(P_U, m)$ and sends y; upon receiving y, U retrieves m by computing $D(S_U, y)$; here E and D are polynomial-time algorithms chosen so that it will be infeasible, for any other user, to compute m from y.

Notice that in this set-up any other user is thought to be a "passive" adversary who tries to retrieve m by computing solely on inputs y and P_U. This is indeed a mild type of adversary and other types of attacks have been considered in the literature. It is widely believed that the strongest type of attack among all the natural ones is the *chosen-ciphertext attack*. In such an attack, someone tries to break the system by asking and receiving decryptions of ciphertexts of his choices. Rivest has shown that Rabin's scheme (whose breaking is, for a passive adversary, as hard as factoring if the messages are uniformily selected strings of a given length) is easily vulnerable to such an attack. Indeed, this is an attack feasible to any employee who works at the decoding equipment of, say, a large bank. The power by this attack is very well exemplified by an elegant scheme of Rabin [R] that is as secure as factoring in the passive adversary model but is easily broken by chosen-ciphertext attack. Since observing this phenomenon, people tried to design cryptosystems invulnerable to such attacks, but in vain. A positive answer has been found [GMT] only allowing interaction, during the encryption process, between legal sender and legal receiver. However, for the standard (non-interactive) Diffie-and-Hellman model, the existence of a cryptosystem invulnerable to chosen ciphertext attack has been an open problem since 1978.

Non-interactive zero-knowledge proofs allow us to finally solve this problem. The essence of our solution (instead of its details) is informally described as follows. Instead of sending U an encryption, y, of a message m, one is required to send two strings: y and and σ, where σ is a *zero-knowledge and non-interactive proof that the sender knows the decoding of* y. The "decoding equipment" (read: the decoding function) checks that σ is convincing and, if so, outputs m, the decoding of y; Otherwise, it outputs nothing. Notice that, now, being able to use the decoding equipment provably is of no advantage! In fact, only when we feed it with ciphertexts whose decoding we can prove we know, does the decoding equipment output these decodings! In other words, the decoding equipment can only be used to output what we already know. A detailed discussion of this powerful application will appear in the final paper.

(A formal setting and the proof require some care. For instance, the decoding equipment may be used as an oracle to check whether a given string σ is a "correct proof of knowledge". Thus, in particular, one should prove that such an oracle cannot help. In the final paper we will essentially show that if one

[3] It should be noted where the 2or3A comes into play. Let L is a poly-time language, $x \notin L$ and G is a graph 3-colorable if and only if $x \in L$. Let M, or input G, follow the protocol in Section 3 to (necessarily) *fake* P's proof that G is 3-colorable.

Such proof will not fool a poly-time judge not because the quadratic-residousity labeling would give away that the graph is not 3-colored; but because he can easily check that $x \notin L$ (and thus that the underlying graph, without any labeling, is not 3-colorable).

The 2or3A guarantees that this easy check is not available to a poly-time judge. In the final paper we essentially show that there are no other easy checks.

can generate a legal (y, σ) pair without having m as an input, then one can easily decrypt all messages on input y and P_U only.)

6 Improvements

It has very often been the case in cryptography that new notions and results have been first obtained under a specific intractability assumption. This is so because one can exploit the additional properties of a specific, candidate intractable problem. Number theory has always played a leading role as a basis of new cryptographic concepts. For instance, cryptographically strong pseudo-random number generators were first exhibited based on the computational difficulty of the discrete logarithm problem [BM]. Only later a construction was presented based on a more general assumption: the existence of one-way permutations [Y]. Finally it has been established that cryptographically strong pseudo-random number generation is possible if and only if one-way functions exist [L].

Non-interactive ZKPS have been introduced and still are based on the intractability of algebraic problems. Very recently, our intractability assumption has been relaxed. DeSantis, Micali, and Persiano have exhibited non-interactive ZKPS based only on the quadriatic residensity assumption.

We hope this new notion will be given a sounder foundation; hopefully by basing it on the existence of any general trap-door or one-way function.

7 References

[ACGS] W. Alexi, B. Chor, O. Goldreich, and C. Schnorr *RSA/Rabin Bits Are* $1/2 + 1/_{poly}(\log N)$ *Secure*, To appear SIAM J. on Computing.

[B1] M. Blum, *Coin Flipping by Telephone*, IEEE COMPCON 1982, pp. 133-137.

[B2] M. Blum, unpublished manuscript

[BBS] M. Blum, L. Blum and M. Shub,*A simple and secure pseudo-randomnumber generator,*SIAM Journal of Computing, 1986

[BGGHMR] M. Ben-Or, O. Goldreich, S. Goldwasser, J. Hastad, S. Micali, and P. Rogaway, to appear

[BH] R. Boppana, J. Hastad and S. Zachos, *Interactive Proofs Systems for CO-NP Imply Polynomial Time Hierarchy Collapse*, In preperation.

[BM] M. Blum and S. Micali, *How To Generate Sequences Of Cryptographically Strong Pseudo-Random Bits*, SIAM J. on Computing, Vol. 13, Nov 1984, pp. 850-864

[DH] Diffie, W., and M.E. Hellman, *New Directions in Cryptography,*IEEE Trans. on Inform. Theory,

[F] L. Fortnow, *The Complexity of Perfect Zero-Knowledge*, Proc. 19th ann. Symp. on Theory of Computing, New York, 1987.

[FFS] Feige, Fiat and A. Shamir, *Zero-knowledge proofs of identity*, Proceedings of the 19th Annual ACM Symp. on Theory of Computing, 1987, pp. 210–217

[GM] S. Goldwasser, and S. Micali, *Probabilistic Encryption*, JCSS Vol. 28, No. 2, April 1984.

[GMR] S. Goldwasser, S. Micali and C. Rackoff, *The Knowledge Complexity of Interactive Proof-Systems*, To appear SIAM J. on Computing (manuscript available from authors).

[GoMiRi] S. Goldwasser, S. Micali, and R. Rivest, *A Digital Signature Scheme Secure Against Adaptive, Chosen Cyphertext Attack* To appear in SIAM J. on Computing (available from authors)

[GMT] S. Goldwasser, S. Micali, and P. Tong, *Why and how to establish a perivate code in a public network*, Proc. 23rd Symp. on Foundations of Computer Science, Chicago, Ill., 1982

[GMW] O. Goldreich, S. Micali and A. Wigderson, *Proofs that Yield Nothing but their Validity and a Methodology of Cryptographic Design*, Proc. of FOCS 1986.

[GMW2] O.Goldreich, S. Micali and A. Wigderson, *How to Play Any Mental Game*, Proceedings of the 19th Annual ACM Symp. on Theory of Computing, 1987, pp. 218–229.

[GS] S. Goldwasser and M. Sipser, *Private Coins versus Public Coins in Interactive Proof Systems*, Proceedings of the 18th Annual ACM Sympl on Theory of Computing, 1986, pp. 59–68.

[R] M. Rabin, *Digitalized signatures and public-key functions as intractable as factorization*, MIT/LCS/TR-212, Technical report MIT, 1978

[Y] A.Yao, *Theory and Application of Trapdoor Functions*, Proc. of 23rd FOCS, IEEE, Nov., 1982, pp. 80-91.