Recycling Random Bits in Composed Perfect Zero-Knowledge

Giovanni Di Crescenzo *

Abstract. In this paper we give techniques for recycling random bits both in the interactive and in the non-interactive model for perfect zero-knowledge proofs. Our first result is a non-interactive perfect zero-knowledge proof system for proving that at least one out of any given polynomial number of statements is true, in which the amount of public random bits used is the same as that for proving a single statement. Our second result is an interactive perfect zero-knowledge proof system for proving any given polynomial number of statements, in which the amount of private random bits used by the prover is, apart from a constant factor, the same as that for proving a single statement. In order to get a randomness-efficient proof system, we also reduce the random string of the verifier by using a multi-bit commitment scheme. The statements considered are of quadratic non residuosity modulo a Blum integer.

1 Introduction

Quantitative aspects of randomness in cryptographic protocols are now emerging as a new interesting research area in cryptology (see, e.g., [5, 13, 25, 26, 23, 1]). In perfect zero-knowledge proof systems the randomness of the prover is crucial to obtain the perfect zero-knowledge property. This paper investigates quantitative aspects of randomness in perfect zero-knowledge proof systems both in the interactive and in the non-interactive model.

Zero-knowledge.

Goldwasser, Micali and Rackoff [19] introduced the concept of interactive proof systems as a method for proving the veridicity of membership of a string to a language. In the same paper, they introduced zero-knowledge proof systems as a method for proving such statements without revealing any additional information. The model for zero-knowledge proofs considers an all-powerful prover interacting with a poly-bounded verifier; moreover, both parties are allowed to flip coins.

Any language having an interactive proof has a computational zero-knowledge proof (see [18, 2, 21]), that is a proof which does not reveal additional information to a poly-bounded verifier. On the other hand, only few languages (basically relying on random self-reducible [27] properties) have been proved to have a perfect zero knowledge proofs (see [19, 18, 17, 27, 3, 12]). Perfect zero-knowledge

* Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92093; and Dipartimento di Informatica ed Applicazioni, Università di Salerno, 84081 Baronissi (SA), Italy

L.C. Guillou and J.-J. Quisquater (Eds.): Advances in Cryptology - EUROCRYPT '95, LNCS 921, pp. 367-381, 1995.

proofs are proofs which do not reveal additional information even to an infinitely-powerful verifier. It is unlikely that they can be given for NP-complete languages, as this would imply that the polynomial hierarchy collapses to its second level ([16, 7]). Thus giving a perfect zero-knowledge proof for a language allows to give evidence that a language is not NP-complete. Moreover, perfect zero-knowledge proofs do not rely on any unproven hypothesis and really capture the intrinsic properties of the concept of zero-knowledge proof systems. For all these reasons, giving new techniques for perfect zero-knowledge proofs is still an interesting research area.

Randomness.

The soundness of interactive proof systems is strongly based on the unpredictability of the random questions that the verifier makes to the prover. On the other hand, any language having a proof system with a probabilistic prover has one with a deterministic prover: we can choose the prover that maximizes the acceptance probability of the verifier. In [1] techniques for recycling the randomness of the verifier have been given for Arthur-Merlin proof systems. In zero-knowledge proof systems, instead, the prover has to use randomness in computing his messages not to reveal information to the verifer, as he might do in an interactive proof. In [20] the necessity of randomness for provers and verifiers in zero-knowledge proof systems is shown.

In the non-interactive model (see [6, 4]) for zero-knowledge proofs, the prover and the verifier share a random reference string and the proof is a single message sent by the prover to the verifier. One of the main problems of the non-interactive model is that often the size of the public random string bounds the length of the theorem that can be proved. Thus it is very desirable to give non-interactive zero-knowledge proofs for many statements using the same public random string. (see also [4, 15] for discussions). This problem was solved in the case of computational zero-knowledge in [4, 14, 15]. However, in the case of perfect zero-knowledge, a solution to this problem is still unknown. Our first result in this paper is a non-interactive perfect zero-knowledge proof system for proving that at least one out of any polynomial number of statements is true, in which the length of the public random bits is the same as that for proving a single statement.

In the interactive model for perfect zero-knowledge proofs, no technique has been given in order to recycle the randomness of the prover. Thus, the better way for a prover to prove many statements of a certain language in perfect zero-knowledge was using different and independently chosen random strings for each new statement. Our second result in this paper is an interactive perfect zero-knowledge proof system for proving any polynomial number of statements, in which, up to a constant factor, the random string used by the prover is the same as for proving a single statement. To make our proof system randomness-efficient, we also reduce the random string of the verifier by using a multi-bit (weak-to-strong) commitment scheme.

The statements considered are of quadratic non residuosity modulo a Blum integer. Our results are for specific languages, while the result in [15] is given for all languages in NP. On the other hand, non-interactive perfect zero-knowledge proofs have been given so far only for languages that are composition of quadratic non residuosity statements. Also, interactive zero-knowledge proofs for quadratic residuosity are among the most used for cryptographic applications like identification schemes.

In all our proof system the prover runs in probabilistic polynomial time, when given the factorization of the Blum modulus as private input.

Organization of the paper:

In Section 2 we describe some number theoretic properties about quadratic residuosity and Blum integers that will be useful in our protocols, and review the definition of non-interactive perfect zero-knowledge proof systems and the non-interactive perfect zero-knowledge proof system of [4] for the language of quadratic non residuosity. In Section 3 we give our result of recycling public random bits in the non-interactive model for perfect zero-knowledge proofs. In Section 4 we give a multi-bit commitment scheme whose security is based on the difficulty of factoring Blum integers. In Section 5 we give our result of recycling the private random bits of the prover in the interactive model for perfect zero-knowledge proofs.

2 Background and Definitions

2.1 Quadratic residuosity and Blum integers

Quadratic Residuosity. For each integer $x > 0$, the set of integers less than x and relatively prime to x form a group under multiplication modulo x denoted by Z_x^*. We say that $y \in Z_x^*$ is a *quadratic residue* modulo x if and only if there is a $w \in Z_x^*$ such that $w^2 \equiv y \bmod x$. If this is not the case, then y is a *quadratic non residue* modulo x. The quadratic residuosity predicate of an integer $y \in Z_x^*$ can be defined as $\mathcal{Q}_x(y) = 0$ if y is a quadratic residue modulo x and 1 otherwise. Define Z_x^{+1} and Z_x^{-1} to be, respectively, the sets of elements of Z_x^* with Jacobi symbol $+1$ and -1 (see [24] for the definition of Jacobi symbol). The Jacobi symbol can be computed in deterministic polynomial time. Also, define the set $QR_x = \{y \in Z_x^* \mid \mathcal{Q}_x(y) = 0\}$ of quadratic residues modulo x, and the set $NQR_x = \{y \in Z_x^{+1} \mid \mathcal{Q}_x(y) = 1\}$ of quadratic non residues modulo x. The quadratic residuosity predicate defines the following equivalence relation in Z_x^*: $y_1 \sim_x y_2$ if and only if $\mathcal{Q}_x(y_1 y_2) = 0$. Thus, the quadratic residues modulo x form a $\sim_x$ equivalence class. If $y \in Z_x^{-1}$, then y is a quadratic non residue modulo x. However, if $y \in Z_x^{+1}$, no efficient algorithm is known to compute $\mathcal{Q}_x(y)$. The fastest way known for computing $\mathcal{Q}_x(y)$ consists of first factoring x.

Blum integers. In this paper we will consider the special moduli called Blum integers. An integer x is a Blum integer, in symbols $x \in$ BL, if and only if $x = p^{k_1} q^{k_2}$, where p and q are different primes both $\equiv 3 \bmod 4$, and k_1 and k_2 are odd integers. If x is a Blum integer, Z_x^* is partitioned by $\sim_x$ into 4 equally large equivalence classes. Also, $|Z_x^{+1}| = |Z_x^{-1}|$ and Z_x^{+1} is partitioned into 2 equally large equivalence classes, one made of quadratic residues modulo x and the other made of quadratic non residues modulo x. Thus, for this special class of integers we have that for any $y_1, y_2 \in Z_x^*$, $\mathcal{Q}_x(y_1) = \mathcal{Q}_x(y_2) \Longrightarrow \mathcal{Q}_x(y_1 y_2) = 0$, and $\mathcal{Q}_x(y_1) \neq \mathcal{Q}_x(y_2) \Longrightarrow \mathcal{Q}_x(y_1 y_2) = 1$. Then a quadratic residue modulo a Blum integers x has exactly four square roots, one in each $\sim_x$ equivalence class, and exactly one of them will be a quadratic residue modulo x. Moreover, if x is a Blum integer, then $-1 \bmod x$ is a quadratic non residue with Jacobi symbol $+1$. This implies that on input a Blum integer x, it is easy to generate a random quadratic non residue in Z_x^{+1}: randomly select $r \in Z_x^*$ and output $-r^2 \bmod x$. Finally, if x is a Blum integer, given its prime factors p, q, it is possible to compute square roots modulo x in deterministic polynomial time.

We refer the reader to [24, 4] for a more formal treatment and for proofs.

2.2 Perfect Zero-Knowledge Proof Systems

Now we review the definition of perfect zero-knowledge proof systems of [19]. Let L be a language and x be an instance to it. Let P a probabilistic Turing machine and V a deterministic Turing machine that runs in time polynomial in the length of its first input.

Definition 1. We say that (P, V) is a Perfect Zero-Knowledge Proof System for the language L if

1. *Completeness.* $\forall x \in L$, $|x| = n$ and for all sufficiently large n,
$$\mathbf{Pr}(P \leftrightarrow V)(x) = \text{ACCEPT} \geq 1 - 2^{-n}.$$
2. *Soundness.* $\forall x \notin L$, $|x| = n$ and for all sufficiently large n, for all probabilistic algorithms P$'$,
$$\mathbf{Pr}(P' \leftrightarrow V)(x) = \text{ACCEPT} \leq 2^{-n}.$$
3. *Perfect Zero Knowledge.* for each V$'$ there exists a probabilistic machine $S_{V'}$ running in expected polynomial time such that $\forall x \in L$, the two probability spaces $S_{V'}(x)$ and $View_{V'}(x)$ are equal, where by $View_V(x)$ we denote the probability space $(R; conv)$, where R is the random tape of V, and $conv$ is the transcript of a conversation between P and V on input x given that R is the random tape of V.

Now we review the definition of non-interactive perfect zero-knowledge proof systems of [4], referring the reader to the original paper for motivations and discussions. Let L be a language and x be an instance to it. Let P a probabilistic Turing machine and V a deterministic Turing machine that runs in time polynomial in the length of its first input.

Definition 2. We say that (P, V) is a Non-Interactive Perfect Zero-Knowledge Proof System for the language L if there exists a positive constant c such that:

1. *Completeness.* $\forall x \in L$, $|x| = n$ and for all sufficiently large n,

$$\mathbf{Pr}(\sigma \leftarrow \{0,1\}^{n^c}; Proof \leftarrow P(\sigma, x): V(\sigma, x, Proof) = 1) \geq 1 - 2^{-n}.$$

2. *Soundness.* For all probabilistic algorithms *Adversary* giving pairs $(x, Proof)$ as output, where $x \notin L$, $|x| = n$, and all sufficiently large n,

$$\mathbf{Pr}(\sigma \leftarrow \{0,1\}^{n^c}; (x, Proof) \leftarrow Adversary(\sigma): V(\sigma, x, Proof) = 1) \leq 2^{-n}.$$

3. *Perfect Zero Knowledge.* There exists an efficient simulator algorithm S such that $\forall x \in L$, the two probability spaces $S(x)$ and $View_V(x)$ are equal, where by $View_V(x)$ we denote the probability space $View_V(x) = \{\sigma \leftarrow \{0,1\}^{|x|^c};\ Proof \leftarrow P(\sigma, x) : (\sigma, Proof)\}$.

We call the random string σ, input to both P and V, the *reference string.*

2.3 Quadratic non-residuosity

In [4] a non-interactive perfect zero-knowledge proof system for the languages of quadratic non residuosity modulo a *Regular*(2) integer was given (a *Regular*(2) integer is odd, is not a perfect square and has two prime factors). We briefly review it here, as it will be useful to better understand our protocols.

On input an n-bit integer x and an integer $y \in Z_x^{+1}$, the prover takes $2n$ integers $\sigma_1, \ldots, \sigma_{2n}$ from the 20^2-bit long reference string σ such that $\sigma_i \in Z_x^{+1}$, for $i = 1, \ldots, 2n$. Then, for each σ_i, the prover does the following. If σ_i is a quadratic residue modulo x, then he uniformly chooses an integer $r_i \in Z_x^*$ such that $r_i^2 = \sigma_i \bmod x$. On the other hand, if σ_i is a quadratic non residue modulo x, then he uniformly chooses an integer $r_i \in Z_x^*$ such that $r_i^2 = y \cdot \sigma_i \bmod x$. Finally, the prover sends $r_i \in Z_x^*$ to the verifier. The verifier checks that for each $i = 1, \ldots, 2n$, $r_i^2 = y^{b_i} \cdot \sigma_i \bmod x$, for some bit b_i. If so, then he accepts, otherwise he rejects (see [4] for proofs).

If we consider the language of quadratic non residuosity modulo a Blum integer (instead of a *Regular*(2) integer) then we have to add a phase to this protocol, in which it is proved that x is a Blum integer, as in [10]. In this case the length of the reference string is $50n^2$. From now on, we will call this protocol (C,D).

3 OR of many statements on a single random string

In this section we give a non-interactive perfect zero-knowledge proof system for proving the OR of any polynomial number (in the size of the input) of statements of quadratic non residuosity modulo a Blum integer x in which the length of the random reference string used is the same as that of the protocol (C,D) for the language of quadratic non residuosity modulo x.

For simplicity, we consider the language of triples (x, y_1, y_2), such that x is a Blum integer, and at least one of y_1, y_2 is a quadratic non residue modulo x:

$$\text{OR} = \{(x, y_1, y_2) \,|\, x \in \text{ BL}, y_1, y_2 \in Z_x^{+1}, (y_1 \in NQR_x) \vee (y_2 \in NQR_x)\}.$$

We give a non-interactive perfect zero-knowledge proof system (A,B) for the language OR and then briefly explain how our construction can be easily extended to any polynomial number (in $|x| = n$) of integers y_i.

An informal description. On input (x, y_1, y_2), A writes the reference string σ as the concatenation of two sufficiently long strings σ_1, σ_2. Then, A uses σ_1 to give a non-interactive perfect zero-knowledge proof that x is a Blum integer, for instance, using the proof system given in [10]. Now, we make a simplifying assumption by considering the reference string σ_2 written as the concatenation of $2n$ integers $\sigma_{2i} \in Z_x^{+1}$.[2] The main idea for proving that at least one of y_1, y_2 is a quadratic non residue modulo x is the following. The prover P computes in a careful way (to be specified later) two strings ρ_1, ρ_2 of the same length as σ, such that $\rho_{1i} \cdot \rho_{2i} = \sigma_{2i} \bmod x$, for $i = 1, \ldots, 2n$, and sends a 'proof' that y_1, y_2 are quadratic non residues modulo x, computed using as reference strings ρ_1, ρ_2, respectively. V verifies that the 'proofs' are correctly constructed on the reference strings ρ_1, ρ_2 and that $\rho_{1i} \cdot \rho_{2i} = \sigma_{2i} \bmod x$, for $i = 1, \ldots, 2n$. Of course, P has to convince V even if, say, y_1 is a quadratic residue modulo x. Thus, he computes the two strings ρ_1, ρ_2 in the following way: the string ρ_1 is made of integers of the same quadratic residuosity as y_1, and the string ρ_2 is computed from ρ_1 and σ in order to satisfy $\rho_{1i} \cdot \rho_{2i} = \sigma_{2i} \bmod x$, for $i = 1, \ldots, 2n$. We observe that if y_1 is a quadratic residue, then the string ρ_1 is made of all quadratic residues. Thus P can compute a faked 'proof' of quadratic non residuosity for y_1 using ρ_1 as a reference string. Then, the string ρ_2 is a uniformly distributed string, and if y_2 is a quadratic non residue modulo x, then P can give a non-interactive proof of quadratic non residuosity for y_2 using ρ_2 as a reference string. If B cannot distinguish between integers with the same quadratic residuosity as y_1 and integers with the same quadratic residuosity as y_2, then the faked 'proof' for y_1 and the 'real' proof for y_2 will appear indistinguishable to him. That is, each of the two will appear as a proof of quadratic non residuosity (as in [4]) for y_j using ρ_j as a reference string, for $j = 1, 2$.

Let (E,F) be a non-interactive perfect zero-knowledge proof system for the language BL of Blum integers (see, e.g., [10]). Now we give a formal description of (A,B).

[2] Even if this is not true in general, (for instance, in a uniformly distributed random string there may be integers in Z_x^{-1}) in [4, 10] a technique preserving perfect zero-knowledge is given for transforming a uniformly distributed string into a string made of integers in Z_x^{+1}.

Input to A and B: $(x, y_1, y_2) \in$ OR, such that $|x| = n$, and the reference string $\sigma = \sigma_1 \circ \sigma_{21} \circ \cdots \circ \sigma_{2,2n}$, where $\sigma_{2i} \in Z_x^{+1}$, for $i = 1, \ldots, 2n$.

Input to A: x's factorization.

Instructions for A.

A.1 Prove that x is a Blum integer by running algorithm E on input x and using σ_1 as a reference string. Send E's output Pf to B.

A.2 If $y_2 \in QR_x$ then set $z_2 = y_1$ and $z_1 = y_2$;
else set $z_1 = y_1$ and $z_2 = y_2$.

A.3 For $i = 1, \ldots, 2n$,
uniformly choose $c_{1i} \in \{0, 1\}, r_{1i} \in Z_x^*$;
set $\rho_{1i} = z_1^{c_{1i}} \cdot r_{1i}^2 \bmod x$ and $\rho_{2i} = \sigma_{2i} \cdot \rho_{1i}^{-1} \bmod x$;
compute $c_{2i} \in \{0, 1\}$ and a randomly chosen $r_{2i} \in Z_x^*$ such that $r_{2i}^2 = z_2^{c_{2i}} \cdot \rho_{2i} \bmod x$.
if $z_1 = y_1$ then send $(r_{1i}, r_{2i}), (c_{1i}, c_{2i}), (\rho_{1i}, \rho_{2i})$ to B;
else send $(r_{2i}, r_{1i}), (c_{2i}, c_{1i}), (\rho_{2i}, \rho_{1i})$ to B.

Input to B: The proof Pf and the set $\{(u_{1i}, u_{2i}), (d_{1i}, d_{2i}), (\tau_{1i}, \tau_{2i})\}_{i=1,\ldots,2n}$ sent by A.

Instructions for B.

B.1 Verify that Pf is a proof that x is a Blum integer by running algorithm F on input x and using σ_1 as a reference string.

B.2 For $i = 1, \ldots, 2n$,
verify that $u_{1i}^2 = y_1^{d_{1i}} \cdot \tau_{1i} \bmod x$, and $u_{2i}^2 = y_2^{d_{2i}} \cdot \tau_{2i} \bmod x$;
verify that $\sigma_{2i} = \tau_{1i} \cdot \tau_{2i} \bmod x$;

B.3 If all the verifications are successful then accept else reject.

Let (C,D) be the non-interactive perfect zero-knowledge proof system for the language NQR. Now we prove the following

Theorem 3. *(A, B) is a non-interactive perfect zero-knowledge proof system for the language OR, such that the length of the random reference string used by (A,B) is the same as that used by (C,D).*

Proof. *Randomness.* As for (C,D), the random reference string is divided into two parts. The first is $30n^2$-bits long and it is used to prove that x is a Blum integer, as in (C,D). The second is used to prove that at least one of y_1, y_2 is a quadratic non residue modulo x, and is $20n^2$-bits long, that is, exactly as the remaining part of the reference string in (C,D) used to prove the quadratic non residuosity of only one integer.

Completeness. If at least one of y_1, y_2 is a quadratic non residue modulo x, then A, which is given x's factorization as private input, can set z_1 equal to this integer. Also, he can run algorithm E to prove that x is a Blum integer, and finally compute a random square root of $z_2^{c_{2i}} \cdot \rho_{2i} \bmod x$, for some bit c_{2i} and any $i = 1, \ldots, 2n$. Then completeness follows from these observations and the completeness of (E,F).

Soundness. We distinguish two cases. First, assume that x is not a Blum integer. Then from the soundness of (E,F), we have that B accepts with negligible probability. Then, assume that y_1, y_2 are quadratic residues modulo x and that B accepts. Then the two strings ρ_1, ρ_2 can be written as the concatenation of integers that are the product of a quadratic residue and $y_j^{d_{ji}}$, for some bit d_{ji}. Thus each string ρ_j is made of $2n$ quadratic residues modulo x, and so is also σ_2, as B verifies that $\sigma_{2i} = \rho_{1i} \cdot \rho_{2i} \bmod x$, for every $i = 1, \ldots, 2n$. The event that σ_2 is made of all quadratic residues happens with probability $1/2^{2n}$. Then the probability that there exists a modulus x such that B accepts is at most $2^n/2^{2n} = 1/2^n$, which is negligible.

Perfect Zero-Knowledge. We give a simulator M such that, for each $(x, y_1, y_2) \in$ OR, the output of M on input (x, y_1, y_2) and the view of B in the protocol (A,B) are identically distributed. Let N be the simulator for the non-interactive perfect zero-knowledge proof system (E,F) for the language of Blum integers.

Input to M: $(x, y_1, y_2) \in$ OR, where $|x| = n$.
Instructions for M:

1. Run the algorithm N on input x obtaining as output (σ_1, Pf); set $Proof = Pf$;
2. For $i = 1, \ldots, 2n$,
   uniformly choose $u_{1i}, u_{2i} \in Z_x^*$, and $d_{1i}, d_{2i} \in \{0, 1\}$;
   set $\tau_{1i} = y_1^{-d_{1i}} \cdot u_{1i}^2 \bmod x$ and $\tau_{2i} = y_2^{-d_{2i}} \cdot u_{2i}^2 \bmod x$;
   set $\sigma_{2i} = \tau_{1i} \cdot \tau_{2i} \bmod x$;
   set $Proof = Proof \circ (u_{1i}, u_{2i}) \circ (d_{1i}, d_{2i}) \circ (\tau_{1i}, \tau_{2i})$.
3. Set $\sigma = \sigma_1 \circ \sigma_{21} \circ \cdots \circ \sigma_{2,2n}$ and output $(\sigma, Proof)$.

It is easy to see that the simulator runs in probabilistic polynomial time. Now we prove that the output of M and the view of B in the protocol are equally distributed. First of all let us prove that for each $i = 1, \ldots, 2n$, it holds that d_{1i}, d_{2i} are uniformly distributed bits both in the output of M and in the view of B. This happens as in the output of M both d_{1i} and d_{2i} are uniformly chosen over $\{0, 1\}$; and in the view of B one of them, say d_{1i}, is uniformly chosen over $\{0, 1\}$, and the other, d_{2i}, satisfies $u_{2i}^2 = z_2^{d_{2i}} \cdot \sigma_{2i} \cdot \rho_{1i}^{-1} \bmod x$, where z_2 is a quadratic non residue modulo x. We see from the above equation that the value of d_{2i} depends from the quadratic residuosity of σ_{2i}, and so it is uniformly distributed over $\{0, 1\}$. Then we see that for each $i = 1, \ldots, 2n$, it holds that $u_{1i}, u_{2i}, \tau_{1i}, \tau_{2i}$ are randomly distributed integer in Z_x^* such that $u_{1i}^2 = y_1^{d_{1i}} \cdot \tau_{1i} \bmod x$ and $u_{2i}^2 = y_2^{d_{2i}} \cdot \tau_{2i} \bmod x$ both in the output of M and in the view of B. Finally, also σ_1 is equally distributed in both spaces, from the perfect zero-knowledge of (E,F). □

Now, let us briefly explain how this protocol easily extends to proving an OR of any polynomial number $m = |x|^c$ of quadratic non residuosity modulo x statements. The input is then $(x, y_1, \ldots, y_m)$. Assume that y_m is a quadratic non residue modulo x. Then the prover computes a string τ_j for each y_j in a way much similar as in the algorithm A. More precisely, strings τ_j, for $j = 1, \ldots, m-1$ are computed as ρ_1, and the last string τ_m is computed similarly to ρ_2: by multiplying all elements in previous strings τ_j and the relative element of σ. The remaining parts of the protocol are constructed exactly as in (A,B).
Let $(\text{A,B})_m$ be the above protocol, and let OR_m be the language of $(m+1)$-tuples $(x, y_1, \ldots, y_m)$ such that x is a Blum integer and at least one of $y_1, \ldots, y_m$ is a quadratic non residue modulo x. Also, let (C,D) be the non-interactive perfect zero-knowledge proof system for the language NQR. Then we have the following

Theorem 4. *$(A, B)_m$ is a non-interactive perfect zero-knowledge proof system for the language OR_m, such that the length of the random reference string used by $(A,B)_m$ is the same as that used by (C,D).*

This result improves the protocol to prove an OR of m quadratic non residuosity statements given in [10], which needs a reference string of length $O(m)$ times that of the random string used by (C,D).

4 A multi-bit commitment scheme

In this section we describe a scheme (S,R) in which a sender S can commit to many bits and reveal one of them at each round to a receiver R. The main property of this scheme is that it can be implemented with a small amount of randomness. The scheme will be used to reduce the randomness of the verifier in the construction of a randomness-efficient perfect zero-knowledge proof system for proving any polynomial number of quadratic non residuosity statements. Similar techniques have already been used in literature, e.g. in [9] where efficient weak-to-strong bit commitment schemes are presented, using universal hash functions. Also, in [22] general techniques for efficient weak-to-strong bit commitment schemes have been given. First of all, let us define a multi-bit (weak-to-strong) commitment scheme.

Definition 5. A *(weak-to-strong)*[3] *multi-bit commitment scheme* is a two-phase protocol with two participants: a (weak) sender with probabilistic polynomial-time computing power and a (strong) receiver with unlimited computing power. In the first phase (*the commitment phase*), the sender has m bits $b_1, \ldots, b_m$ and commits to them by computing an $(m+1)$-tuple of "keys" $(Com, Dec_1, \ldots, Dec_m)$ and sends Com (the commitment key) to the receiver. The second phase (*decommitment phase*) can be divided in m subphases. In each of these m subphases

[3] Such commitment schemes have been referred in the literature also as ***blob schemes***, see, e.g., [8], and ***statistically hiding bit commitment schemes***, see, e.g. [9].

the sender reveals the bit b_i along with Dec_i (the i-th decommitment key) to the receiver. A (weak-to-strong) multi-bit commitment scheme has the following two main properties: security and correctness. The *security* property states the following: for each $i = 0, \ldots, m-1$, from Com and $Dec_1, \ldots, Dec_i$, the receiver cannot guess b_{i+1} with probability significantly better than $1/2$. The *correctness* property states the following: for each $i = 1, \ldots, m$, the receiver obtains a valid decommitment key for a bit c_i, and he is sure that the sender is revealing the same bit b_i to which he committed before.

We observe that given a (weak-to-strong) 1-bit commitment scheme, it is possible to obtain a (weak-to-strong) multi-bit commitment scheme by just using the 1-bit scheme for each of the many bits. In this case, however, the amount of randomness used in an m-bit commitment scheme is m times that used in a 1-bit scheme. Our m-bit commitment scheme is also derived from a 1-bit scheme, but uses an amount of randomness equal to only twice the same amount of the 1-bit commitment scheme. The 1-bit scheme that we use is the following folklore scheme: on input a Blum integer x, and in order to commit to a bit b, the sender uniformly chooses an integer $r \in Z_x^{+1}$ if $b = 0$ and $r \in Z_x^{-1}$ if $b = 1$ and outputs $w = r^2 \bmod x$. In order to reveal bit b, the sender sends r and the receiver sets $b = 0$ if $r \in Z_x^{+1}$, and $b = 1$ if $r \in Z_x^{-1}$. It is easy to see that the receiver obtains a uniformly distributed quadratic residue modulo x for both values of b. Also, if the sender can reveal the commitment in two different ways, then he knows two different square roots modulo x of w and thus he can factor x. Now we extend this commitment scheme to a multi-bit commitment scheme (S,R), using only twice the same amount of randomness. The correctness property is still based on the intractability of factoring Blum integers.

Input to S and R: A Blum integer x and m bits $b_1, \ldots, b_m$.

Commitment Phase

S: Uniformly choose $w_0 \in Z_x^{+1}$ and $s \in Z_x^{-1}$;
for $i = 1, \ldots, m$,
set $r_i = w_{i-1} \cdot s^{b_i} \bmod x$, $w_i = r_i^2 \bmod x$ and $Dec_i = r_i$;
set $Com = (s, w_m)$ and send Com to R.
R: Verify that $s \in Z_x^{-1}$.

Decommitment Phase

For $i = m, \ldots, 1$,
S: Send (b_i, Dec_i) to R.
R: Let $z_i = Dec_i$; verify that $z_i^2 = z_{i+1} \cdot s^{b_i} \bmod x$;
verify that $(z_i \in Z_x^{+1}$ AND $b_i = 0)$ OR $(z_i \in Z_x^{-1}$ AND $b_i = 1)$.

The security property of the above scheme follows from the following observation: for any m-tuple of bits $b_1, \ldots, b_m$ input to (S,R), the integers w_m and

$Dec_2, \ldots, Dec_m$ sent by S are uniformly distributed quadratic residues modulo x. The correctness property of the above scheme follows from the following observation: If any sender can correctly reveal two different bits in the i-th decommitment subphase, then he sends two different square root modulo x of a same integer z_i^2, and thus he can factor x. The above discussion informally proves the following

Theorem 6. *If factoring Blum integers is hard, then (S,R) is a (weak-to-strong) multi-bit commitment scheme such that the number of random bits used in (S,R) does not depend on the number of bits committed.*

A formal proof will appear in the final paper.

5 Many statements on a single random string

In this section we give an interactive perfect zero-knowledge proof system (P,V) for proving any polynomial number (in the size of the input) of statements of quadratic non residuosity modulo a Blum integer x, in which the length of the private random string used by the prover is, apart for a small constant factor, the same as that used in [19] for proving a single quadratic residuosity statement.

We consider the language of $(m+1)$-tuples $(x, y_1, \ldots, y_m)$, such that x is a Blum integer, and $y_1, \ldots, y_m$ are quadratic non residues modulo x; formally:

$$\text{AND}_m = \{(x, y_1, \ldots, y_m) \mid x \in \text{BL}, y_i \in Z_x^{+1}, y_i \in NQR_x, \text{for } i = 1, \ldots, m\}.$$

An informal description. The main ideas of this protocol are: 1) a four round interactive perfect zero-knowledge proof by combining a coin-tossing protocol between P and V with a non-interactive proof by P (a similar technique has been used in [11]), and 2) P uses the non-interactive proof of quadratic non residuosity of an integer y_k and V's challenges in order to compute the next non-interactive proof for the integer y_{k+1}, without using any random bits. More precisely, in the first three rounds P and V run a coin-tossing protocol, in which V commits to his random bits using the scheme (S,R) of previous section. After the coin-tossing protocol both parties can compute a random reference string σ. P writes the string σ as $\rho \circ \tau$; then, τ will be used only once to prove that x is a Blum integer using, e.g., the proof system in [11], and ρ will be used to prove that all the y_i's are quadratic non residues modulo x. Now, P proves that the first integer y_1 is a quadratic non residue modulo the Blum integer x, by using a modification of the non-interactive proof system (C,D) of [4] and ρ as a reference string. More precisely, he takes $2n$ integers $\rho_i \in Z_x^{+1}$ from the string ρ; then he randomly chooses a square root $s_{i,1} \in Z_x^{+1}$ of $\rho_i \cdot y_1^{d_{i,1}} \bmod x$, for some bit $d_{i,1}$, and sends $s_{i,1}$ to V. Now, observe that the proof for y_1 constituted by the $s_{i,1}$'s looks very similar to the random integers ρ_i. In fact, both the $s_{i,1}$'s and the ρ_i's are integers in Z_x^{+1}. Now, V will reveal other $2n$ bits $b_{i,2}$ to which he committed in the first round, and P will give a non-interactive proof for y_2 as for y_1, but using as a random reference string the $2n$ integers $u_{i,2} = (-1)^{b_{i,1}} \cdot s_{i,1} \bmod x$. We observe that as -1 is a quadratic non residue modulo x, then the quadratic residuosity of the $u_{i,2}$'s is thus uniformly chosen by V. The proof system (P,V) continues analogously for the other integers y_j. For a better exposition, we avoid to be very formal in our step-by-step description of (P,V).

Input to P and V: $(x, y_1, \ldots, y_m)$ such that $|x| = n$, and $m = n^c$.
Input to P: x's factorization.

(*Proving the first statement* '$y_1 \in NQR'_x$.)

V.1 Uniformly choose $2nm$ bits $b_{i,k}$ and $10n^2$ bits e_i;
use algorithm S to commit to them and send *Com* to P.
P.1 Uniformly choose $50n^2$ bits c_i and send them to V.
V.2 Use algorithm S to reveal all bits d_i to P.
P.2 Use algorithm R to check that V had committed to bits e_i;
compute σ as the bitwise xor of the c_i's and the e_i's; let $\sigma = \rho \circ \tau$;
prove that $x \in$ BL, using the algorithm E and τ as a reference string;
for each of the first $2n$ integers $\rho_i \in Z_x^{+1}$ from ρ,
if $\rho_i \in QR_x$ uniformly choose $s_{i,1} \in Z_x^{+1}$ such that $s_{i,1}^2 = \rho_i \bmod x$;
if $\rho_i \in NQR_x$ uniformly choose $s_{i,1} \in Z_x^{+1}$ such that $s_{i,1}^2 = y_1 \cdot \rho_i \bmod x$;
set $u_{i,1} = \rho_i$ and send $s_{i,1}$ to V.
V.3.1 Use algorithm F to verify the proof that x is a Blum integer;
verify that $s_{i,1}^2 = y_1^{d_{i,1}} \cdot u_{i,1} \bmod x$ for some bit $d_{i,1}$, and $i = 1, \ldots, 2n$;
use algorithm S to reveal other $2n$ bits $b_{i,2}$ to P.

(*Proving the k-th statement* '$y_k \in NQR'_x$, *for* $k = 2, \ldots, m$.)

P.3.k For $i = 1, \ldots, 2n$,
use algorithm R to check that V had committed to bit $b_{i,k}$;
set $u_{i,k} = (-1)^{b_{i,k}} \cdot s_{i,k-1} \bmod x$;
if $u_{i,k} \in QR_x$ uniformly choose $s_{i,k} \in Z_x^{+1}$ such that $s_{i,k}^2 = u_{i,k} \bmod x$;
if $u_{i,k} \in NQR_x$ uniformly choose $s_{i,k} \in Z_x^{+1}$ such that
$s_{i,k}^2 = y_k \cdot u_{i,k} \bmod x$;
send $s_{i,k}$ to V.
V.3.k Verify that $s_{i,k}^2 = y_k^{d_{i,k}} \cdot u_{i,k} \bmod x$ for some bit $d_{i,k}$, and for $i = 1, \ldots, 2n$;
use algorithm S to reveal other $2n$ bits $b_{i,k}$ and send their Dec_i to P.
V.4 If all the verifications are successful then accept else reject.

We see that our protocol is very randomness-efficient. In fact, the only random bits used by the prover are those chosen in step P.1. Then, the length of the random string used by the prover in order to prove m quadratic non residuosity statements is the same, apart from a constant factor, as in [19] for proving a single quadratic residuosity statement. Also, the length of the random string used by V during a proof of m quadratic non residuosity statements is equal to a $10n^2$-bit initial random string (needed in the proof that x is a Blum integer), and then only $2n$ random bits for each new statement (needed for the soundness in the proof of each new statement). In [19] the verifier uses n random bits for a single quadratic residuosity statement.

For the *completeness* requirement, observe that if x is a Blum integer and all integers $y_1, \ldots, y_m$ are quadratic non residues modulo x, then P, using x's factorization, can run algorithm E, compute the quadratic residuosity of the u_i's and compute the square roots belonging to Z_x^{+1} of the integers $u_{i,k} \cdot y_k^{d_{i,k}} \bmod x$, for some bits $d_{i,k}$. Thus V accepts with overwhelming probability.

For the *soundness* requirement, first of all assume x is not a Blum integer. Then V accepts with negligible probability from the soundness of (E,F). Now, suppose that $y_1, \ldots, y_{k-1}$ are quadratic non residues modulo x and y_k is a quadratic residue, for some $k \in \{1, \ldots, m\}$. We observe that the proof for y_{k-1} sent by P′ is made of integers $s_{i,k-1} \in Z_x^{+1}$, whose quadratic residuosity is chosen by P′ (we recall that both $s_{i,k-1}$ and $-s_{i,k-1} \bmod x$ are integers in Z_x^{+1} and square roots of a same number). On the other hand, the quadratic residuosity of the $u_{i,k}$'s forming the random string on which P′ has to prove y_k is given by the quadratic residuosity of the $s_{i,k-1}$'s sent by P′ xored with the new random bits $b_{i,k}$ revealed by V. Then if V accepts the $u_{i,k}$'s associated to y_k are all quadratic residues modulo x, and thus P′ has given his $s_{i,k-1}$'s such that the quadratic residuosity of each $s_{i,k-1}$'s is exactly equal to $b_{i,k}$. This implies that P′ has guessed $2n$ bits $b_{i,k}$ to which V committed in the first round, but this happens with probability at most $1/2^{2n}$, for the security property of the multi-bit commitment scheme (S,R). Thus, the probability that there exists an n-bit modulus x such that V accepts the proof of any possible $y_k \in QR_x$ is at most $m \cdot 2^n/2^{2n} = m/2^n$, which is negligible.

For the *perfect zero-knowledge* requirement, we sketch a description of the simulator M on input $(x, y_1, \ldots, y_m) \in \text{AND}_m$. The simulation of the proof of the first statement can be easily derived from that in [11]. Now we describe the simulation of the proof of the k-th statement. Assume that M has successfully simulated the first $k-1$ proofs. This implies that he has learned all the questions $b_{i,j}$ of V′ relative to them, and that he has just sent to V′ the proof for y_{k-1} consisting in $2n$ integers $s_{i,k-1} \in Z_x^{+1}$. Then M receives from V′ other $2n$ bits $b_{i,k}$ to which V′ had committed in the first round. After learning bits $b_{i,k}$, M simulates again all P's messages of the proofs of the j-th statements, for $j = 1, \ldots, k-1$, in such a way that he can simulate also the proof of the k-th statement. He does this in the following way: let $d_{i,k}$ be the bits sent by V′ relative to the j-th proof. Then M uniformly chooses $2n$ integers $s_{i,k} \in Z_x^{+1}$ and computes $u_{i,k} = y_k^{d_{i,k}} \cdot s_{i,k}^2 \bmod x$, and $s_{i,k-1} = u_{i,k} \cdot (-1)^{b_{i,k}} \bmod x$. M computes the $s_{i,j}$ and $u_{i,j}$ for $j < k$ analogously to the above $s_{i,k-1}$ and $u_{i,k}$. Also, he computes the bits c_i analogously as in the simulation of the proof for y_1. Now M rewinds V′ to the state just after his first step and sends the messages just computed to V′ in the proper succession in order to simulate P's messages. We observe that if V′ does not change any of his decommitted bits, then M succeeds in simulating all proofs of the first k statements. On the other hand, if V′ reveals some bits in different way, then by the security property of the multi-bit commitment scheme (S,R), V′ sends to M two different square roots of a same quadratic residue modulo x. Thus M can factor x and simulate perfectly the protocol by just running the algorithm of P.

The above discussion informally proves the following

Theorem 7. *(P, V) is a perfect zero-knowledge proof system for the language AND_m such that the length of the random string used by P is, up to a small constant factor, the same as in the proof system in [19] for the language QR. Moreover, the number of the random bits used by V is $2|x|m + O(|x|^2)$.*

A formal proof will appear in the final paper.

Acknowledgements

Many thanks go to Alfredo De Santis, Russell Impagliazzo, Markus Jakobsson and Giuseppe Persiano for useful discussions, and to an anonymous referee, whose careful comments have improved the exposition of this paper.

References

1. M. Bellare, O. Goldreich, and S. Goldwasser, *Randomness in Interactive Proof Systems*, Proceedings of the 31th Annual IEEE Symposium on Foundation of Computer Science, 1990, pp. 563–572.
2. M. Ben-Or, O. Goldreich, S. Goldwasser, J. Hastad, S. Micali, and P. Rogaway, *Everything Provable is Provable in Zero Knowledge*, in "Advances in Cryptology – CRYPTO 88", vol. 403 of "Lecture Notes in Computer Science", Springer Verlag, pp. 37–56.
3. J. Boyar, K. Friedl, and C. Lund, *Practical Zero-Knowledge Proofs: Giving Hints and Using Deficiencies*, Journal of Cryptology, n. 4, pp. 185–206, 1991.
4. M. Blum, A. De Santis, S. Micali, and G. Persiano, *Non-Interactive Zero-Knowledge*, SIAM Journal of Computing, vol. 20, no. 6, Dec 1991, pp. 1084–1118.
5. C. Blundo, A. De Santis, and U. Vaccaro, *Randomness in Distribution Protocols*, Proceedings of International Colloquium of Algorithms, Languages and Programming (ICALP) 1994.
6. M. Blum, P. Feldman, and S. Micali, *Non-Interactive Zero-Knowledge and Applications*, Proceedings of the 20th Annual ACM Symposium on Theory of Computing, 1988, pp. 103–112.
7. R. Boppana, J. Hastad, and S. Zachos, *Does co-NP has Short Interactive Proofs ?*, Inf. Proc. Lett., vol. 25, May 1987, pp. 127–132.
8. G. Brassard, C. Crépeau, and M. Yung, *Perfect Zero-Knowledge Computationally Convincing Proofs for NP in Constant Rounds*, Theoretical Computer Science, vol. 84, n. 1 (1991) pp. 23-52.
9. I. Damgaard, T. Pedersen, and B. Pfitzmann, *On the existence of statistically hiding bit commitment schemes and fail-stop signatures*, in "Advances in Cryptology – CRYPTO 93", vol. 773 of "Lecture Notes in Computer Science", Springer Verlag, pp. 250–265.
10. A. De Santis, G. Di Crescenzo, and G. Persiano, *Secret Sharing and Perfect Zero-Knowledge*, in "Advances in Cryptology – CRYPTO 93", vol. 773 of "Lecture Notes in Computer Science", Springer Verlag, pp. 73–84.

11. A. De Santis, G. Di Crescenzo, and G. Persiano, *The Knowledge Complexity of Quadratic Residuosity Languages*, in Theoretical Computer Science, Vol. 132 pp. 291-317 (1994).
12. A. De Santis, G. Di Crescenzo, G. Persiano, and M. Yung, *On Monotone Formula Closure of SZK*, Proceedings of the 35th Annual IEEE Symposium on Foundation on Computer Science, November 1994.
13. Y. Desmedt, C. Goutier, and S. Bengio, *Special Uses and Abuses of the Fiat-Shamir Passport Protocol*, in "Advances in Cryptology – CRYPTO 87", vol. 293 of "Lecture Notes in Computer Science", Springer Verlag.
14. A. De Santis and M. Yung, *Cryptographic Applications of the Non-Interactive Metaproof and Many-Prover Systems*, in "Advances in Cryptology – CRYPTO 90", vol. 537 of "Lecture Notes in Computer Science", Springer Verlag, pp 366-377.
15. U. Feige, D. Lapidot, and A. Shamir, *Multiple Non-Interactive Zero-Knowledge Proofs Based on a Single Random String*, in Proceedings of 22nd Annual Symposium on the Theory of Computing, 1990, pp. 308–317.
16. L. Fortnow, *The Complexity of Perfect Zero-Knowledge*, Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, pp. 204–209.
17. O. Goldreich and E. Kushilevitz, *A Perfect Zero Knowledge Proof for a Decision Problem Equivalent to Discrete Logarithm*, in "Advances in Cryptology - CRYPTO 88", Ed. S. Goldwasser, vol. 403 of "Lecture Notes in Computer Science", Springer-Verlag, pp. 57–70.
18. O. Goldreich, S. Micali, and A. Wigderson, *Proofs that Yield Nothing but their Validity and a Methodology of Cryptographic Design*, Proceedings of 27th Annual Symposium on Foundations of Computer Science, 1986, pp. 174–187.
19. S. Goldwasser, S. Micali, and C. Rackoff, *The Knowledge Complexity of Interactive Proof-Systems*, SIAM Journal on Computing, vol. 18, n. 1, February 1989.
20. O. Goldreich and Y. Oren, *Definitions and Properties of Zero-Knowledge Proof Systems*, Journal of Cryptology, vol. 7, 1994, pp. 1–32.
21. R. Impagliazzo and M. Yung, *Direct Minimum Knowledge Computations* "Advances in Cryptology – CRYPTO 87", vol. 293 of "Lecture Notes in Computer Science", Springer Verlag pp. 40–51.
22. J. Kilian, *A Note on Efficient Zero-Knowledge Proofs and Arguments* , Proceedings of the 24th Annual ACM Symposium on the Theory of Computing, May 1992.
23. E. Kushilevitz and A. Rosen, *A Randomness-Rounds Tradeoff in Private Computations* "Advances in Cryptology – CRYPTO 94", vol. 293 of "Lecture Notes in Computer Science", Springer Verlag pp. 40–51.
24. I. Niven and H. S. Zuckerman, *An Introduction to the Theory of Numbers*, John Wiley and Sons, 1960, New York.
25. T. Okamoto and K. Ohta, *Disposable Zero-Knowledge Authentications and their Applications to Untraceable Electronic Cash* "Advances in Cryptology – CRYPTO 89", vol. 435 of "Lecture Notes in Computer Science", Springer Verlag.
26. T. Okamoto and K. Ohta, *How to Utilize the Randomness of Zero-Knowledge Proof Systems* "Advances in Cryptology – CRYPTO 90", vol. 537 of "Lecture Notes in Computer Science", Springer Verlag.
27. M. Tompa and H. Woll, *Random Self-Reducibility and Zero-Knowledge Interactive Proofs of Possession of Information*, Proc. 28th Symposium on Foundations of Computer Science, 1987, pp. 472–482.