This paper appears in *Advanced in Cryptology – Eurocrypt 2000, LNCS, Springer Verlag, 2000*

Fair Encryption of RSA Keys

Guillaume Poupard and Jacques Stern

École Normale Supérieure, Département d'informatique
45 rue d'Ulm, F-75230 Paris Cedex 05, France
email: {Guillaume.Poupard,Jacques.Stern}@ens.fr

Abstract. Cryptography is more and more concerned with elaborate protocols involving many participants. In some cases, it is crucial to be sure that players behave fairly especially when they use public key encryption. Accordingly, mechanisms are needed to check the correctness of encrypted data, without compromising secrecy. We consider an optimistic scenario in which users have pairs of public and private keys and give an encryption of their secret key with the public key of a third party. In this setting we wish to provide a publicly verifiable proof that the third party is able to recover the secret key if needed. Our emphasis is on size; we believe that the proof should be of the same length as the original key.
In this paper, we propose such proofs of fair encryption for El Gamal and RSA keys, using the Paillier cryptosystem. Our proofs are really efficient since in practical terms they are only a few hundred bytes long. As an application, we design a very simple and efficient key recovery system.

Keywords. Verifiable encryption, Homomorphic cryptosystems, Practical proofs of knowledge, Key recovery.

1 Introduction

In some cryptographic applications it is crucial to be sure that players behave fairly, especially when they use public key encryption. For example, we can consider a voting scheme where each player encrypts the name of his favorite candidate. It can be useful to convince anybody that the encrypted name is indeed in the list of the candidates without revealing any information about this name. Accordingly, mechanisms are needed to check the correctness of encrypted data, without compromising secrecy.

We consider an optimistic scenario in which users have pairs of public and private keys and give an encryption of their secret key with the public key of a third party. In this setting we wish to provide a publicly verifiable proof that the third party is able to recover the secret key if needed. We use the term *fair encryption* for such a verifiable encryption. Note that the third party is not involved during encryption or during verification of the proof. In optimistic systems like [1], the third party is active only in case of dishonest behavior of one participant; it is implicitly assumed that the knowledge that the third party is able to solve any conflict is enough to deter anybody from cheating.

Our emphasis is on size; we believe that the proof should be approximately of the same length as the original key. Consequently, general techniques of zero-knowledge proofs cannot be used and we have to design specific proof systems which are very efficient.

Previous work.
Publicly verifiable encryption is not a new concept and it has been used in applications like secret sharing or key escrow. In 1998, Young and Yung [29] proposed auto-recoverable auto-certifiable public key cryptosystems based on verifiable encryption of secret keys using double decker exponentiation which makes the proofs efficient but certainly not really practical, in a natural sense that is defined below. Furthermore, this system does not separate the recoverability verification from the ability to certify public keys.[1]

Efficient vs Practical protocols.
Following the tradition of complexity theory, cryptographers generally consider that a protocol is *"efficient"* when both its running time and the size of the transmitted data are polynomial in some typical parameters such as the bit length of the integers in use and other security parameters. This approach enables to eliminate non polynomial schemes that are obviously not usable in practice but those which survive cannot necessarily be considered practical. For example we can think about general multi-party computation protocols.

In this paper, we focus on protocols that are efficient but also really practical. As an example, let us consider the Fiat-Shamir identification scheme [10]; if we note k the security parameter and $|N|_{\mathrm{b}}$ the size of used integers, its time complexity is $O(k \times {|N|_{\mathrm{b}}}^2)$ and the communication complexity measuring the size of the exchanged data is $O(k \times |N|_{\mathrm{b}})$. Thus this protocol is efficient but not very practical. As another example, the Schnorr scheme [24] is efficient and even practical since its time complexity is $O({|N|_{\mathrm{b}}}^2)$ and its communication complexity is $O(k + |N|_{\mathrm{b}})$; the security parameter k may be raised with only a modest increase in size.

Our aim is to design proof systems that are practical at least in terms of communication, i.e. such that the size of the proofs are of the same order than the size of the underlying objects. This is motivated by the scenario that we have in mind since we wish to turn our proofs into non interactive "certificates". In this setting, the optimization of the size of transmitted data is of crucial importance.

Our results.
In this paper, we propose proofs of fair encryption for secret keys of any encryption scheme based on the discrete logarithm problem or on the intractability of the factorization, including RSA and its variants. The asymmetric secret keys are encrypted using any homomorphic public key cryptosystem like those of

[1] Those results have been recently improved in [30]. See also [27].

Naccache-Stern [17], Okamoto-Uchiyama [18] or Paillier [19]. In this paper we only focus on the Paillier scheme but we can immediately adapt the protocols in order to use the Okamoto-Uchiyama cryptosystem which onewayness is based on the well studied factorization problem instead of the new one introduced in [19].

More precisely, we give a protocol to prove that a ciphertext enables a third party to recover the El Gamal secret key related to a public one. Such a proof is very short and the workload of the third party during recovery is very small. We also propose a scheme for fair encrypting the factorization of a public modulus. Such a proof is also very small but the workload of the third party is much more important than for El Gamal keys since, from a theoretical point of view, the recovery time and the cheating time are polynomially related. However, we describe practical parameters to show that actual applications are feasible.

Finally, as an application, we design a very simple and efficient key recovery system that can be used with any kind of keys, including RSA keys. We propose the first non-interactive proof of recoverability of RSA keys short enough (a few hundred bytes) to be appended as a certificate to any ciphertext. A consequence is that the recoverability verification is no longer performed by the certification authority. Consequently, this approach is more flexible than auto-recoverable cryptosystems [29] and more secure than binding cryptography [28].

Those results follow from a careful analysis of previously proposed building blocks: homomorphic cryptosystems based on exponentiation modulo composite integers, the so-called bounded range commitment schemes that tries to prove the knowledge of a discrete logarithm in a given range and short proofs of knowledge for factoring proposed in [23].

Outline of the paper.
In section 2 we describe notations and give a precise description of the three building blocks: trapdoor discrete logarithm cryptosystems, Diophantine commitment and short proof of knowledge for factoring. Security proofs for those two last protocols appear in appendix. Next, in section 3, we describe our fair encryption protocols first for El Gamal and then for RSA. Finally, in section 4 we show how fair encryption enables the design of very simple and efficient key escrow systems.

2 Preliminary tools

Throughout this paper, we use the following notation: for any integer n,
- we use $\varphi(n)$ to denote the Euler totient function, i.e. the cardinality of ${\mathbb{Z}_n}^*$,
- we use $\lambda(n)$ to denote Carmichael's lambda function defined as the largest order of the elements of ${\mathbb{Z}_n}^*$.

It is well known that if the prime factorization of an odd integer n is $\prod_{i=1}^{\eta} {q_i}^{f_i}$ then $\varphi(n) = \prod_{i=1}^{\eta} {q_i}^{f_i-1}(q_i-1)$ and $\lambda(n) = \mathrm{lcm}_{i=1\ldots\eta}\left({q_i}^{f_i-1}(q_i-1)\right)$.

For any integer x, $|x|_\mathrm{b} = (\lfloor \log_2(x) \rfloor + 1)$ is the number of bits of x. Finally, a prime number p is a *strong prime* if $p = 2p' + 1$ and p' is also prime. Our computing model is the probabilistic polynomial time Turing machine (Pptm), whose running time is a polynomial in specified parameters.

2.1 Homomorphic cryptosystems

Various cryptosystems which encrypt a message M by raising a base g to the power M modulo some integer have been proposed [15, 3, 17–19]. Their security is related to the intractability of computing discrete logarithm in the base g. As usual, the computation becomes easy using a trapdoor. As an important consequence of this encryption technique, those schemes have homomorphic properties that can be informally stated as follows:

$$E(M_1 + M_2) = E(M_1) \times E(M_2) \quad \text{and} \quad E(k \times M) = E(M)^k$$

The early examples of such schemes could only achieve bit by bit encryption [15] or had very limited bandwidth [3]. However, recently, three cryptosystems with significant bandwidth have been proposed: one by Okamoto and Uchiyama [18] based on the exponentiation modulo P^2Q of messages from $\mathbb{Z}_P$ where P and Q are prime numbers, the second by Naccache and Stern [17] based on the exponentiation modulo PQ of messages from $\mathbb{Z}_\sigma$ with σ a smooth divisor of $\varphi(PQ)$ and finally a proposal of Paillier [19] which extends the system of [18] by using exponentiation modulo P^2Q^2 and messages from $\mathbb{Z}_{PQ}$. In the following, we only describe protocols based on the Paillier cryptosystem but we insist on the fact that any of those three cryptosystems could be used.

The Paillier cryptosystem is based on the properties of the Carmichael lambda function in $\mathbb{Z}_{N^2}{}^*$. We recall here the main two properties: for any $w \in \mathbb{Z}_{N^2}{}^*$, $w^{\lambda(N)} = 1 \bmod N$ and $w^{N\lambda(N)} = 1 \bmod N^2$.

Key Generation. Let N be an RSA modulus $N = P \times Q$, where P and Q are prime integers s.t. $\gcd(N, \varphi(N)) = 1$. Let G be an integer of order multiple of N modulo N^2. The public key is (N, G) and the secret key is $\lambda(N)$.

Encryption. To encrypt a message $M \in \mathbb{Z}_N$, randomly choose u in $\mathbb{Z}_N{}^*$ and compute the ciphertext $c = G^M \times u^N \bmod N^2$.

Decryption. To decrypt c, compute $M = \dfrac{L(c^{\lambda(N)} \bmod N^2)}{L(G^{\lambda(N)} \bmod N^2)} \bmod N$ where the L-function takes as input an element equal to 1 modulo N and computes $L(u) = \frac{u-1}{N}$.

The integer $g^{\lambda(N)} \bmod N^2$ is equal to 1 modulo N so there exists $\beta \in \mathbb{Z}_N$ such that $g^{\lambda(N)} = 1 + \beta N \bmod N^2$. Furthermore, we note that $\beta = L(g^{\lambda(N)} \bmod N^2)$. Consequently, $c^{\lambda(N)} = \left(g^M u^N\right)^{\lambda(N)} = \left(g^{\lambda(N)}\right)^M = (1 + \beta N)^M = 1 + M\beta N \bmod N^2$. So $M \times L(g^{\lambda(N)}) = L(c^{\lambda(N)}) \bmod N$.

Security. It is conjectured that the so-called composite residuosity class problem, that exactly consists in inverting the cryptosystem, is intractable. The semantic security is based on the difficulty to distinguish N^{th} residues modulo N^2. We refer to [19] for details.

2.2 Diophantine Commitment

In 1989, Schnorr [24] proposed his famous signature scheme which may be viewed as proof of knowledge of a discrete logarithm modulo a prime number. Since then, many authors have tried to adapt the scheme in order to add control over the size of the secret value. Such a *bounded-range commitment* has many applications and it has been used for group signature by Camenisch and Michels [6], for electronic cash by Chan, Frankel and Tsiounis [8], for verifiable secret sharing by Fujisaki and Okamoto [12] and finally for proving that a modulus is the product of two safe primes by Camenisch and Michels [7]. However no satisfactory solution has appeared at the moment[2]. Known proposals are only able to prove that the discrete logarithm is not "too far" from a fixed range, their analysis is complex (and sometimes erroneous as in the Eurocrypt '98 version of [8]) and their security is often based on non-standard assumptions such as the so-called "strong RSA assumption" needed to make proofs efficient. In this paper, we adopt a totally different strategy in the analysis of bounded-range commitment schemes.

Let $\mathcal{G}$ be a multiplicative finite group. As an example of such a group, in the next sections we use groups of unknown order $\mathcal{G} = {\mathbb{Z}_{N^2}}^*$ where N is an RSA modulus. Let S be an integer and G be an element of $\mathcal{G}$. We consider a player who has a secret integer x that lies in the range $[0, S[$ and who computes, in $\mathcal{G}$, the public value $\Gamma = G^x$.

We do not know how to prove the knowledge of a discrete logarithm in the range $[0, S[$ of Γ in base G. Consequently we only prove a weaker property. Let A and B be two parameters whose values are analyzed later. We describe a practical statistically zero-knowledge interactive proof of knowledge of $\sigma \in]-A, A[$ and $\tau \in]0, B[$ such that $G^\sigma = \Gamma^\tau$ in $\mathcal{G}$. It should be clear that we do not prove the knowledge of $x \in [0, S[$ such that $\Gamma = G^x$. However, in practice, the prover needs to know such an x in order to be able to perform the proof.

> **<u>Protocol 1</u>**: The following round is repeated ℓ times. At each round, the prover randomly chooses an integer r in $[0, A[$ and computes the commitment $t = G^r$ in $\mathcal{G}$. Then he sends t to the verifier who answers a challenge e randomly chosen in $[0, B[$. The prover computes $y = r + ex$ (an integer in $\mathbb{Z}$) and sends it to the verifier who checks $t = G^y \times \Gamma^{-e}$ in $\mathcal{G}$ and $0 \leq y < A$.

A security analysis of this scheme is proposed in appendix A. Note that this protocol is similar to previous proposals for bounded-range commitment [6, 8, 12, 7] but that the analysis is really different and does not use non-standard hypothesis like the strong-RSA assumption.

Let us summarize the security results. A prover who knows $x \in [0, S[$ is accepted with probability higher than $1 - \ell SB/A$ so A must be much larger than ℓSB in order to make the protocol correct. Furthermore, the protocol is

[2] Last minute : see F. Boudot's paper [5] in this proceeding.

sound, i.e. a prover who convinces a verifier with probability higher that $1/B^\ell$ must know $\sigma \in]-A, A[$ and $\tau \in]0, B[$ such that $G^\sigma = \Gamma^\tau$. Finally, in the complexity theory setting, if we consider a security parameter k and if all the parameters A, B, S and ℓ are viewed as functions of k, the protocol is statistically zero-knowledge if $\ell \times B$ is polynomial in k and if $\ell SB/A$ is negligible.

When S is chosen, the choice of the remaining parameters is directed by those results. From a theoretical point of view, if we consider that the security parameter k is related to the cheating probability $1/2^k$ for an adversary, the soundness implies that $B^\ell \geq 2^k$. Furthermore, the protocol is zero-knowledge if it can be simulated in polynomial time. Since the time complexity of the simulation is $O(\ell \times B)$, the parameters ℓ and B must be polynomial in k. Finally, the correctness and the zero-knowledge property show that A must be such that $\ell SB/A$ is negligible.

From a practical point of view, we can fix the security parameter k to 80 for example. If $|S|_\mathrm{b} = 160$, the following practical values for the other parameters are reasonable: $B = 2^{20}$, $\ell = 4$, $A = 2^{2+160+20+80} = 2^{262}$.

Let ω be the order of G in $\mathcal{G}$. Note that the relation $G^\sigma = \Gamma^\tau$ does not even imply the existence of a discrete logarithm for Γ with base G but, if $\Gamma = G^x \bmod N^2$, we have $x\tau = \sigma \bmod \omega$ and consequently $x = \sigma \times \left(\frac{\tau}{\gcd(\tau,\omega)}\right)^{-1} \bmod \frac{\omega}{\gcd(\tau,\omega)}$.

The Diophantine commitment can be made non-interactive using the Fiat-Shamir heuristic [10]. The verifier's challenge e is replaced with the hash value of the commitment t and of the public data using a collision-resistant hash function H. It is widely believed that such a transformation guarantees an accurate level of security as soon as H is random enough. Furthermore, the security of this approach can be formalized using the random oracle model [20].

2.3 Short proof of knowledge for factoring

Proofs of knowledge for the factorization of an integer n have been known for a long time. But, even if they are claimed efficient according to complexity theoretical arguments, none of them can be considered practical for many applications because of their significant communication complexity: the proof is much longer than the object it deals with.

A new strategy have been used in [23]. The protocol is a proof of knowledge of a small common discrete logarithm of $z^n \bmod n$ for a few randomly chosen elements z modulo n. This scheme is very efficient; when suitably optimized, its communication complexity is only $O(k + |n|_\mathrm{b})$ bits, where k is a security parameter. In this setting, the size of the proof is similar to the size of the integer n. The improvement in comparison with the previously known schemes can therefore be compared with the difference of efficiency between the Fiat-Shamir scheme and the Schnorr one. Furthermore, the computational complexity is proved to be $O((|n|_\mathrm{b} + k) \times k)$ multiplications modulo n both for the prover and the verifier but strong heuristic evidence shows that $O((|n|_\mathrm{b} + k) \times k/\log k)$ is enough. This might appear a small improvement but it has drastic consequences in practical terms: only three modular exponentiations both for the prover and the verifier are needed to obtain a very high level of security.

> **<u>Protocol 2</u>**: First the prover and the verifier agree on mutually randomly chosen integers $z_i \in \mathbb{Z}_n{}^*$ for $i = 1..K$. Then the following elementary round is repeated ℓ times. The prover randomly chooses an integer r in $[0, A[$ and computes, for $i = 1..K$, the commitments $t_i = z_i^r \bmod n$. Then he sends the t_is to the verifier who answers a challenge e randomly chosen in $[0, B[$. The prover computes $y = r + (n - \varphi(n)) \times e$ (in $\mathbb{Z}$) and sends it to the verifier who checks $0 \leq y < A$ and, for $i = 1..K$, $z_i^{y-n\times e} = t_i \bmod n$.

A security analysis of this scheme appears in [23]. The choice of the parameters ℓ and B must be such that $B^\ell > 2^k$, where k is a security parameter, in order to make the protocol sound. Furthermore, the parameter A must be much larger than $(n-\varphi(n))\ell B$ to guarantee the completeness and the zero-knowledge property but A must also be smaller than n to guarantee the soundness. Consequently, n must verify $(n - \varphi(n))\ell B \ll n$. For the applications we consider in this paper, such a proof is used to prove the knowledge of integers like RSA modulus with large prime factors so this condition is always satisfied. Note that if n has small prime factors, the proof is no longer zero-knowledge but it is still sound so a prover cannot try to cheat choosing an integer n with small factors. Such a choice would only compromise his own security.

Using classical techniques, the commitments t_i can be hashed. This trick makes the communication complexity independent of K. Accordingly, the protocol is really practical in term of communication whatever K may be. Furthermore, the proof can be made non-interactive; the z_i are chosen by means of a hash function repeatedly applied to the integer n and the verifier's challenge e is replaced with the hash value of the commitments and of the public data. The size of such a proof is very small in practice, i.e. similar to the size of n.

3 Fair Encryption of secret keys

We consider a third party who chooses his own private and public keys in the Paillier cryptosystem. Let (N, G) be his public key. We also consider a user who has a pair (SK, PK) of related secret and public keys for any cryptosystem (not necessarily Paillier's one). A fair encryption of SK consists of a ciphertext Γ and of a non-interactive proof of fairness; Γ encrypts some secret data related to SK with the public key of the third party and the proof convinces anybody that the third party is able to recover SK using PK, Γ and his Paillier secret key.

Note that the third party is not involved in the process of fair encryption or during verification of the proof of fairness. As in many "optimistic" systems like [1], the third party is active only in case of dishonest behavior of one participant;

it is implicitly assumed that the knowledge that the third party is able to solve any conflict is enough to deter anybody from cheating.

We propose fair encryption scheme for secret keys of all known public key encryption schemes based on the discrete logarithm problem or on the difficulty of factorization. We first give the application to the case of El Gamal keys. Then, we study the case of RSA keys.

3.1 Fair encryption of El Gamal type keys

A fair encryption of an El Gamal secret key x consists of an encryption Γ of x, obtained with the public key of a third party, and of a publicly verifiable non-interactive proof that the third party would be able to recover x from Γ and Y if needed. Such systems have already been proposed (see for example the attempt in [2]) but we explain at the end of this section why previous solutions are not satisfactory. Note that, in order to simplify the presentation of the protocol, we only consider non-randomized Paillier scheme but a semantically secure version can be used as shown in the next section for the case RSA case.

A third party first chooses his public key (N, G) and the related private key to be used with the Paillier cryptosystem, i.e. $N = PQ$ an RSA modulus and G an element of order multiple of N in $\mathbb{Z}_{N^2}{}^*$. Let us further consider a strong prime number $p = 2q + 1$ and a generator g of $\mathbb{Z}_p{}^*$. Each user chooses a private key $x \in \mathbb{Z}_{p-1}$ and computes his public key $Y = g^x \bmod p$. In order to make a fair encryption of his secret key x, he computes the ciphertext $\Gamma = G^x \bmod N^2$ and a non-interactive proof of third party's ability to compute x from Y and Γ. We now describe an interactive version of such a proof that will further be made non-interactive.

We define $S = p - 1$ and $\mathcal{G} = \mathbb{Z}_{N^2}{}^*$. Let A, B and ℓ be Diophantine commitment parameters as described in section 2.2.

Protocol 3:
The following round is repeated ℓ times. At each round, the prover randomly chooses an integer r in $[0, A[$ and sends the commitment $t = (\Gamma^r \bmod N^2, g^r \bmod p)$ to the verifier who answers an integer e randomly chosen in $[0, B[$. The prover computes $y = r + ex$ and sends it to the verifier who checks $t = (G^y \times \Gamma^{-e} \bmod N^2, g^y \times Y^{-e} \bmod p)$ and $0 \leq y < A$.

This protocol runs in parallel the Girault scheme [13] analyzed in [22] and Diophantine commitment. Just as for each of the two schemes separately, we can prove correctness and statistical zero-knowledge property provided $\ell SB/A$ is negligible and $\ell \times B$ is polynomial in the security parameter k. Furthermore, if a prover is accepted with probability $> 1/B^\ell$ then he must know (σ, τ) with $|\sigma| < A$, $0 < \tau < B$, $G^\sigma = \Gamma^\tau \bmod N^2$ and $g^\sigma = Y^\tau \bmod p$. In other words, if an adversary viewed as a probabilistic Turing machine is accepted with probability $> 1/B^\ell$, we can use it in order to make an extractor that computes such a pair (σ, τ).

Theorem 1. *The third party can recover a fair encrypted secret key x from Y and Γ in time $O(|N|)$ if the recoverability proof is valid, provided $N \geq 2\sqrt{2}AB$.*

Proof. First note that the discrete logarithm x of Y in base g modulo p, i.e. the secret key x related to the El Gamal public key Y, exists because g generates $\mathbb{Z}_p{}^*$. The proof associated with the encryption Γ shows that there exists (σ, τ) such that $g^\sigma = Y^\tau \bmod p$ so we have $\sigma = \tau \log_g Y \bmod p-1$. As $q = (p-1)/2$ is a prime number, we obtain $\sigma/d = \tau/d \times x \bmod q$, where $d = \gcd(\sigma, \tau)$. Consequently, the knowledge of $(\sigma/d, \tau/d)$ enables to recover x because we can compute $x_0 = (\sigma/d) \times (\tau/d)^{-1} = \sigma_0 \times \tau_0{}^{-1} \bmod q$ and the secret key $x \bmod p-1$ is x_0 or $x_0 + q$.

Finally, it is enough to show that a third party can recover $(\sigma_0, \tau_0) = (\sigma/d, \tau/d)$ from Γ and Y. We show that this can be efficiently done, provided $N \geq 2\sqrt{2}AB$.

First, he decrypts Γ and obtains $\gamma = \dfrac{L(\Gamma^{\lambda(N)} \bmod N^2)}{L(G^{\lambda(N)} \bmod N^2)} \bmod N$ so $\Gamma^{\lambda(N)} = G^{\gamma\lambda(N)}$. Since $G^\sigma = \Gamma^\tau \bmod N^2$, the previous equation implies $\sigma - \gamma\tau = 0 \bmod N$. Let us consider the solutions of the equation $x - \gamma y = 0 \bmod N$ where x and y are the unknowns. They are elements of a lattice with basis $((N,0),(\gamma,1))$. Since the dimension of the lattice is 2, we can use Gauss' algorithm [9, p.23] in order to find its shortest vector. When running this algorithm, we need to specify the inner product; for the sake of optimization, we replace the standard inner product by $(x,y).(x',y') = xx' + A^2/B^2 \times yy'$. The corresponding norm is $||(x,y)|| = \sqrt{x^2 + A^2/B^2 \times y^2}$. Receiving basis $((N,0),(\gamma,1))$ as its input, the algorithm outputs the shortest vector (σ_0, τ_0) of the lattice. The (unknown) vector (σ, τ) is also in the lattice so that $||(\sigma_0, \tau_0)|| \leq ||(\sigma, \tau)|| < \sqrt{A^2 + A^2/B^2 \times B^2} = \sqrt{2}A$. This means that $|\sigma_0| < \sqrt{2}A$ and $|\tau_0| < \sqrt{2}B$.

From the equations $\gamma\tau - \sigma = \gamma\tau_0 - \sigma_0 = 0 \bmod N$ we obtain $\sigma_0\tau = \tau\tau_0\gamma = \sigma\tau_0 \bmod N$. But $|\sigma_0\tau - \sigma\tau_0| \leq |\sigma_0||\tau| + |\sigma||\tau_0| < 2\sqrt{2}AB$ so, if $N \geq 2\sqrt{2}AB$, $\sigma_0\tau = \sigma\tau_0$ in $\mathbb{Z}$. Furthermore, (σ_0, τ_0) is the shortest vector of the lattice so $\gcd(\sigma_0, \tau_0) = 1$. Finally, the output of the algorithm leads to the computation of the pair $(\sigma/d, \tau/d)$ where d if the gcd of σ and τ. Furthermore, since $0 < \tau < B$, d is less than B.

Classical results about the complexity of Gauss' algorithm (see for example [25]) prove that the number of repetitions needed to find (σ_0, τ_0) is $O(\log(N))$. □

As a consequence, the key recovery process is efficient from a theoretical point of view. Furthermore, practical experiments confirms very high efficiency since a few milliseconds computation can recover the key, whatever the encryption may be but provided the proof is valid.

In conclusion, the protocol is secure both for the prover and the verifier. An dishonest verifier cannot obtain any extra information about SK and if the proof is accepted the third party can recover SK whatever the encryption Γ, even if the prover is dishonest and have unlimited computation power.

Non-interactive version and Optimizations. Many well known optimizations can be applied to the previous proof. The commitment can be replaced by its hash value as described in [14] and it can be precomputed in order to reduce the on-line computation to a very simple non-modular arithmetic operation. We can also reduce the size of the secret key x to about 160 bits as explained in [26]. Finally, this proof can be made non-interactive in order to obtain a very short certificate of fair encryption.

Comparison with previous proposals. At first sight, the key recovery procedure based on lattice reduction might seem overly intricate. We explain why a simple decryption of Γ (as proposed in [2]) presumably does not always enable to recover the secret key.

Let us consider the following cheating strategy based on the ability to extract f-th root, where f is small, without being able to factor. This is a plausible assumption as explained in [4]. The (dishonest) prover chooses an x, computes Y and $\Gamma = G^x \bmod N^2$. Then he extracts an f-th root $\widetilde{\Gamma}$ of Γ modulo N^2. When f divides the challenge e, which happens with probability $1/f$, the prover answers $z = r + (e/f)x$. The verification is still correct but, when the third party decrypts Γ, he obtains a value that has nothing to do with the the discrete logarithm of Y in base g modulo p.

In order to overcome the difficulty one can use a non-standard intractability assumption, the so-called "strong RSA problem", which appears in several papers [12, 6]. With our system, under standard assumption, the third party would find σ and τ such that $g^\sigma = Y^\tau \bmod p$, since $\sigma = (e - e')/f \times x = (\tau/f)x$, and consequently the correct value of the secret key x as was previously explained.

3.2 Fair encryption of RSA keys

We now turn to fair encryption of RSA keys. Using Diophantine commitment and short proofs of knowledge for factoring, we design a fair encryption system which enables the third party to recover the factorization of the RSA modulus, even if it is not of a correct form, i.e. if it is not the product of two large safe primes of approximately the same length. The originality of our solution, in comparison with other proposals is that it does not include any proof that the RSA modulus has exactly two different prime factors. This has important consequence on efficiency.

We consider a scenario where each user chooses two k'-bit prime numbers p and q and computes his RSA modulus $n = pq$. He also computes $x = n - \varphi(n) = p + q - 1$ and the encryption $\Gamma = G^x \bmod N^2$.

We now describe a scheme that enables the user to convince a verifier that the third party is able to factor n using Γ and is Paillier secret key. Let A, B, ℓ and K be parameters of short proof of knowledge for factoring as exposed in section 2.3.

Protocol 4: First the prover and the verifier agree on mutually randomly chosen integers $z_i \in \mathbb{Z}_n{}^*$ for $i = 1..K$. Then the following basic round is repeated ℓ times. The prover randomly chooses an integer r in $[0, A[$ and sends the commitment $t = (G^r \bmod N^2, \{z_i^r \bmod n\}_{i=1..K})$. Then the verifier answers an integer e randomly chosen in $[0, B[$. The prover computes $y = r + ex$ and sends it to the verifier who checks $t = (G^y \times \Gamma^{-e} \bmod N^2, \{z_i^{y-en} \bmod n\}_{i=1..K})$ and $0 \leq y < A$.

The protocol is a parallel execution of a Diophantine commitment and of a short proof of knowledge for factoring. If a prover is accepted with probability $> 1/B^\ell$ then, as usual, one can find a round for which he is able to correctly answer y and y' to different challenges e and e' ($e > e'$) following an identical commitment t. Consequently, for all $i = 1..K$, $z_i^{y-y'} = z_i^{n(e-e')} \bmod n$. If we note $\sigma = y - y'$, $\tau = e - e'$ and $L = n\tau - \sigma$, we have

$$\boxed{\sigma \in]-A, A[,\ \tau \in]0, B[,\ G^\sigma = \Gamma^\tau \bmod N^2 \text{ and, for all } i = 1..K,\ z_i^L = 1 \bmod n}$$

If σ and τ, and consequently L, are known, the same technique as for the proof of soundness of protocol 2 shows that the factorization of n can be extracted with $O(|n|_\mathrm{b} \times |L|)$ multiplications modulo n.

Theorem 2. *The third party can factor n from the fair encryption Γ in time $O(|N| + \sqrt{B})$ if the recoverability proof is valid, provided $N \geq 2\sqrt{2}AB$.*

Proof. First, using the same procedure as for El Gamal keys, he computes $(\sigma_0, \tau_0) = (\sigma/d, \tau/d)$ with $d = \gcd(\sigma, \tau)$. Let L_0 be $n\tau_0 - \sigma_0$; since $L = n\tau - \sigma = d \times L_0$ and $d = \gcd(\sigma, \tau) \leq |\tau| < B$, the third party recovers L divided by a factor d smaller than B.

This missing information can be computed using an algorithm which finds the order of the z_is as follows. For any i, we know that the order of $y = z_i^{L_0} \bmod n$ is less than B because $z_i^L = 1 \bmod n$. The λ-method of Pollard [21] enables to find this order in time $O(\sqrt{B})$ with memory complexity $O(1)$. The idea is to choose a randomly looking function f and to iteratively compute $y_{i+1} = y_i \times y^{f(y_i)} \bmod n$, with $y_0 = 1$, for $i = 1..M$ where M is a fixed parameter. Then, just remembering this last value, we compute ${y'}_{i+1} = {y'}_i \times y^{f({y'}_i)} \bmod n$, with ${y'}_0 = y^B$, until we find an index M' such that $y_M = {y'}_{M'} \bmod n$ or until M' exceeds a fixed bound. If a collision $y_M = {y'}_{M'} \bmod n$ is found (see [21] for a precise analysis), it leads to $y^{B+\sum_{i=0}^{M'-1} f({y'}_i)-\sum_{i=0}^{M-1} f(y_i)} = 1 \bmod n$ so $L_0 \times \left(B + \sum_{i=0}^{M'-1} f({y'}_i) - \sum_{i=0}^{M-1} f(y_i)\right)$ is a multiple of the order of z_i modulo n.

Finally, in time $O(\sqrt{B})$ and with a small amount of memory, the third party recovers L and then factors n with high probability. □

As a consequence of the time complexity of the algorithm in $O(\sqrt{B})$, if B is exponential in the security parameter k, the extractor is not efficient from a theoretical point of view. However, we show in the next sections that the parameters B and ℓ can be chosen in order to guarantee a high level of security, to make the key recovery process feasible by the third party and to have short proofs.

We insist on the fact that our system does not require the modulus n to have exactly two factors; a cheating user cannot gain any advantage using a modulus with three or more factors. Furthermore, the protocol can be immediately adapted to cryptosystems like Okamoto-Uchiyama's where the modulus is not an RSA modulus (e.g $n = p^2 q$).

Remark about cheating provers. In order to show why we need a key recovery procedure that might seem at first sight overly intricate, we consider a cheating strategy that enables a dishonest prover to encrypt something different from $n - \varphi(n)$ in Γ and to give a valid proof. Let f be a factor of $\lambda(n)$ and Γ be $G^{n-\alpha\lambda(n)/f} \bmod N^2$, where α is an integer of about the same size as f. The prover follows the protocol but only answers when f divides the challenge e; this happens with probability $1/f$. In this case he returns $y = r + (e/f) \times (n - \alpha\lambda(n))$. Consequently, verifications are correct because $z_i^y = z_i^r \times z_i^{en} \times z_i^{-e\alpha\lambda(n)/f} \bmod N^2$ and the last term is equal to 1 because f divides e but the third party cannot immediately recover the missing factor f. Notice that such a cheating strategy implies a workload $O(f)$ for cheating but only a workload $O(\sqrt{f})$ for the third party to defeat it.

Randomized non-interactive version. In order to prove the semantic security of the Paillier cryptosystem, the encryption has to be probabilistic. This can be done by multiplying with $u^N \bmod N^2$, where u is randomly chosen in $\mathbb{Z}_N{}^*$: $\Gamma = G^{n-\varphi(n)} \times u^N \bmod N^2$. We can easily modify our schemes in order to use this version of the Paillier scheme. Furthermore, when a third party wants to recover a secret key, the randomization does not affect the decryption process so that nothing is changed in the key recovery. Finally, the proof can be made non-interactive. We obtain the following protocol:

<u>**Protocol 5**</u>:
Encryption. Choose $u \in \mathbb{Z}_N{}^*$ and compute $\Gamma = G^{n-\varphi(n)} \times u^N \bmod N^2$

Proof of Fairness.
Choose $(r_i)_{i=1..\ell} \in_R [0, A[^\ell$ and $(v_i)_{i=1..\ell} \in_R \mathbb{Z}_N{}^{*\ell}$
Compute $t = \left(\left(G^{r_i} v_i^N \bmod N^2\right)_{i=1..\ell}, \left(z_j^{r_i} \bmod n\right)_{i=1..\ell, j=1..K}\right)$
and $(e_1, \ldots e_\ell) = H\left(t, N, G, (z_j)_{j=1..K}, n\right)$
Compute $y_i = r_i + e_i(n - \varphi(n))$ and $y_i{}' = u^{e_i} \times v_i \bmod N$ for $i = 1..\ell$

A non-interactive proof of fairness is a 3ℓ-tuple $((y_i, y_i{}', e_i)_{i=1..\ell})$

Verification.
Check $0 \leq y_i < A$ for $i = 1..\ell$
Compute $t' = \left(\left(G^{y_i} \times y_i{}'^N / \Gamma^{e_i} \bmod N^2\right)_{i=1..\ell}, \left(z_j^{y_i - e_i n} \bmod n\right)_{i=1..\ell, j=1..K}\right)$
Check $(e_1, \ldots e_\ell) = H\left(t', N, G, (z_j)_{j=1..K}, n\right)$

Fair encryption of RSA keys in practice. This section is more practical in character; we consider fair encryption, using protocol 5, for a 1024-bit RSA modulus n with two 512-bit prime factors.

Choice of ℓ and B: the probability of a cheating strategy to succeed during a proof of fairness is smaller than $1/B^\ell$ so $\ell \times |B|_\mathrm{b}$ must be large enough, e.g. $\ell \times |B|_\mathrm{b} = 80$, in order to guarantee a high level of security. Furthermore, the workload for the third party is $O(\sqrt{B})$ in worst cases so B may not be too large. The choice $\ell = 2$ and $B = 2^{40}$ seems satisfactory.

Choice of A: this parameter must be smaller than n and much larger than $(n - \varphi(n))\ell B$ in order to make proofs of knowledge for factoring secure. Since n has two prime factors of about the same size, $n - \varphi(n) \approx \sqrt{n}$. Consequently, A must satisfy $512 + 1 + 40 \ll |A|_\mathrm{b} < 1024$; we advise $A = 2^{633}$.

Choice of K: [23] analyzes the choice of K and shows, using heuristic arguments, that $K = 3$ is a good choice. As was already observed, the communication complexity of protocol 5 does not depend of K. Consequently, the use of $K = 80$ in order to reach a high level of provable security does not make proofs longer.

Choice of N: the Paillier modulus N must satisfies $N > 2\sqrt{2}AB$ in order to make the key recovery process possible. With the previously advised values of parameters A and B, this means $|N|_\mathrm{b} > 675$. Consequently, in order to guarantee the security of the Paillier cryptosystem, $|N|_\mathrm{b} = 1024$ seems to be a good choice.

Choice of H: the function H must be a collision-resistant cryptographic hash function; SHA-1 is a good candidate.

Choice of G: the base G must be an element of order multiple of N modulo N^2. It is very simple to find such an element.

Choice of z_js: the z_js must ideally be mutually randomly chosen in the interactive setting. In practice, they can be pseudo-randomly generated, using a hash function H', with a formula like $z_j = H'(N, G, n, \Gamma, j) \bmod n$.

With those parameters, a complete fair encryption, including the RSA modulus n (1024 bits), the encryption Γ of $n - \varphi(n)$ (2048 bits) and the previously described non-interactive proof (2612 bits) is about only 710 bytes long.

4 Application to Key Recovery systems

As an example of application of fair encryption, we now explain its use in designing very efficient key recovery systems. It must be clear that our aim is not to enter into the controversial debate on the notion of key recovery but to give an application of fair encryption. The general criticisms against such systems are still topical questions. Also, we believe that our notion will find other application, e.g. in the areas of electronic cash, voting schemes or lotteries.

We consider three kinds of participants: users which want to exchange encrypted messages, authorities which are seeking the guaranty that they will obtain the decryption of some messages in specific cases and key recovery agents able to decrypt ciphertexts when requested by the proper authority. Our key recovery systems are designed to be used very easily with any cryptosystem, without adding interaction with authorities, third parties or key recovery agents. The basic idea consists in appending to any ciphertext C a fair encryption Γ of the asymmetric secret key that enables the decryption of C. Γ is encrypted with the Paillier public key of a key recovery agent. The proof of fairness provides a way for anyone (including authorities, proxies, users, ...) to check the correctness of Γ without interaction with any kind of centralized authority and consequently to be convinced that the key recovery agent can actually decrypt C.

Using the Young and Yung setting [29], this leads to the design of auto-recoverable auto-certifiable versions of all the cryptosystems based on discrete logarithm or on factoring. This includes all variants of RSA, the homomorphic schemes [17–19] and many other cryptosystems. But the shortness of our proofs, a few hundred bytes, enables more flexible mechanisms where recoverability verification is separated from the ability to certify public keys. It seems realistic to append short non-interactive proofs to any encrypted message; this leads to a very simple and efficient key recovery system which can be used in conjunction with any common cryptosystem.

We consider a new public key scenario in which each user publicizes is public key PK, a certificate for this key, i.e. a signature of an authority that guarantees the authenticity of PK, and a fair encryption of the secret key related to PK that may enable a key recovery agent to decrypt any ciphertext encrypted with PK. The proof of fairness can be checked by anybody, including people who want to send messages encrypted using PK. In the so-called fair public key scenario, the fair encryption of the secret key related to PK is added to any ciphertext encrypted with PK. Of course, this does not guarantees that it has really been encrypted with PK but the aim of key escrow schemes is only to avoid the use of regular public key infrastructure in dishonest ways; we cannot avoid simple countermeasure like over-encryption or steganography for example. The fair public key scenario can for example be used in a network where servers deliver encrypted messages to Alice only if a fair encryption of her secret key is added to ciphertexts.

Note on shadow public keys. Kilian and Leighton have shown in [16] than many key escrow schemes can be easily misused by dishonest users. The basic idea is to use non-escrowed keys that may be computed from regularly escrowed ones. As a first consequence, the secret keys must be jointly generated by users and authorities. Furthermore, in the more specific case of the system we propose, the proof of fairness should not be used as a subliminal channel to publicize a non-escrowed public key. For example, it is easy to fix a few bits, e.g. in the e_i, but we cannot see any way to increase the bandwidth of such a channel to transmit enough information.

Note on chosen ciphertext attacks. All the known cryptosystems based on a trapdoor discrete log [17–19] are only secure against chosen plaintext attacks but not against chosen ciphertext attacks. As an example, if it is possible to obtain the decryption of a ciphertext in the Okamoto-Uchiyama system, this immediately leads to a multiple of P and consequently to the factorization of N. So a "curious" authority can factor N just asking the recovery of a single key! As a consequence the recovery agent must not reveal recovered keys but only decrypted messages.

With the RSA escrowing scheme, the problem is more subtle because the key obtained by the recovery agent are not meaningless since they are the factorization of a large number. Anyway, an attacker could try to use it as an oracle able to factor a modulus $n = pq$ if and only if $x = p + q - 1 < P$; this binary information can be used to recover the exact value of P. A dichotomic algorithm can easily bound P in such a way that after $O(|P|_{\mathrm{b}})$ queries, the attacker recovers the factorization of N.

The Paillier scheme seems much more resistant to such attacks. Of course it is not secure against chosen ciphertext attacks since it is malleable. Furthermore, we cannot use a non-malleable version since we would no longer be able to make proofs. However, we do not know any attack able to recover a Paillier secret key by CCA; this is the main reason why we prefer to use this scheme and not the Okamoto-Uchiyama cryptosystem.

Note on threshold Paillier scheme. The other reason to use the Paillier scheme is that it is the only homomorphic cryptosystem related to be discrete log problem for which a threshold distributed version [11] is known. This may be of crucial importance for practical applications in order to reduce the trust in the recoverability agent.

References

1. N. Asokan, V. Shoup, and M. Waidner. optimistic Fair Exchange of Digital Signatures. In *Eurocrypt '98*, LNCS 1403, pages 591–606. Springer-Verlag, 1998.
2. F. Bao. An Efficient Verifiable Encryption Scheme for Encryption of Discrete Logarithms. In *CARDIS '98*, 1998.
3. J. Benaloh. *Verifiable Secret-Ballot Elections*. PhD thesis, Yale University, 1987. Available from `http://research.microsoft.com/~benaloh`.
4. D. Boneh and R. Venkatesan. Breaking RSA May Not Be Equivalent to Factoring. In *Eurocrypt '98*, LNCS 1403, pages 59–71. Springer-Verlag, 1998.
5. F. Boudot. Efficient Proofs that a Committed Number Lies in an Interval. In *Eurocrypt 2000*, LNCS. Springer-Verlag, 2000.
6. J. Camenisch and M. Michels. A Group Signature Scheme with Improved Efficiency. In *Asiacrypt '98*, LNCS 1514. Springer-Verlag, 1998.
7. J. Camenisch and M. Michels. Proving in Zero-Knowledge That a Number Is the Product of Two Safe Primes. In *Eurocrypt '99*, LNCS 1592, pages 107–122. Springer-Verlag, 1999.
8. A. Chan, Y. Frankel, and Y. Tsiounis. Easy Come – Easy Go Divisible Cash. In *Eurocrypt '98*, LNCS 1403, pages 561–575. Springer-Verlag, 1998. Available as GTE Tech report.
9. H. Cohen. *A Course in Computational Algebraic Number Theory*. Graduate Texts in Mathematics 138. Springer-Verlag, 1993.

10. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto '86*, LNCS 263, pages 186–194. Springer-Verlag, 1987.
11. PA. Fouque, G. Poupard, and J. Stern. Sharing Decryption in the Context of Voting or Lotteries. In *Financial Cryptography 2000*, LNCS. Springer-Verlag, 2000.
12. E. Fujisaki and T. Okamoto. A Practical and Provably Secure Scheme for Publicly Verifiable Secret Sharing and Its Applications. In *Eurocrypt '98*, LNCS 1403, pages 32–46. Springer-Verlag, 1998.
13. M. Girault. Self-certified public keys. In *Eurocrypt '91*, LNCS 547, pages 490–497. Springer-Verlag, 1992.
14. M. Girault and J. Stern. On the Length of Cryptographic Hash-Values used in Identification Schemes. In *Crypto '94*, LNCS 839, pages 202–215. Springer-Verlag, 1994.
15. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28, 1984.
16. J. Kilian and F.T. Leighton. Fair Cryptosystems Revisited. In *Crypto '95*, LNCS 963, pages 208–221. Springer-Verlag, 1995.
17. D. Naccache and J. Stern. A New Public Key Cryptosystem Based on Higher Residues. In *Proc. of the 5th ACM-CCS*, pages 59–66. ACM press, 1998.
18. T. Okamoto and S. Uchiyama. A New Public-Key Cryptosystem as Secure as Factoring. In *Eurocrypt '98*, LNCS 1403, pages 308–318. Springer-Verlag, 1998.
19. P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Eurocrypt '99*, LNCS 1592, pages 223–238. Springer-Verlag, 1999.
20. D. Pointcheval and J. Stern. Security Proofs for Signature Schemes. In *Eurocrypt '96*, LNCS 1070, pages 387–398. Springer-Verlag, 1996.
21. J. M. Pollard. Monte Carlo Methods for Index Computation (mod p). *Mathematics of Computation*, 32(143):918–924, July 1978.
22. G. Poupard and J. Stern. Security Analysis of a Practical "on the fly" Authentication and Signature Generation. In *Eurocrypt '98*, LNCS 1403, pages 422–436. Springer-Verlag, 1998.
23. G. Poupard and J. Stern. Short Proofs of Knowledge for Factoring. In *Proceedings of PKC2000*, LNCS 1751, pages 147–166. Springer-Verlag, 2000.
24. C. P. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.
25. B. Vallée. Gauss' Algorithm Revisited. *Journal of Algorithms*, 12:556–572, 1991.
26. P. C. van Oorschot and M. J. Wiener. On Diffie-Hellman Key Agreement with Short Exponents. In *Eurocrypt '96*, LNCS 1070, pages 332–343. Springer-Verlag, 1996.
27. E. Verheul. Certificates of Recoverability with Scaleable Recovery Agent Security. In *Proceedings of PKC2000*, LNCS 1751. Springer-Verlag, 2000.
28. E. Verheul and H. van Tilborg. Binding ElGamal: A Fraud-Detectable Alternative to Key-Escrow Proposals. In *Eurocrypt '97*, LNCS 1233, pages 119–133. Springer-Verlag, 1997.
29. A. Young and M. Yung. Auto-Recoverable Auto-Certifiable Cryptosystems. In *Eurocrypt '98*, LNCS 1403, pages 17–31. Springer-Verlag, 1998.
30. A. Young and M. Yung. RSA-based Auto-Recoverable Cryptosystems. In *Proceedings of PKC2000*, LNCS 1751. Springer-Verlag, 2000.

A Security analysis of Diophantine commitment

In order to prove the exact security of Diophantine commitment, the approach of Feige, Fiat and Shamir is followed, first proving completeness, then soundness and, finally, the zero-knowledge property.

Theorem 3 (Completeness). *The execution of the protocol between an honest prover who knows the secret value* $x \in [0, S[$ *and a verifier is successful with probability higher than* $1 - \ell SB/A$.

Proof. If the prover knows a secret $x \in [0, S[$ and follows the protocol, he fails only if $y \geq A$ at some round of the proof. For any value $x \in [0, S[$ the probability of failure of such an event taken over all possible choices of r is smaller than SB/A. Consequently the execution of the protocol is successful with probability $\geq \left(1 - \frac{SB}{A}\right)^\ell \geq 1 - \frac{\ell SB}{A}$. □

Theorem 4 (Soundness). *Assume that some adversary* $\widetilde{P}$ *is accepted with probability* $\varepsilon' = 1/B^\ell + \varepsilon$, $\varepsilon > 0$. *Then there exists an algorithm which, with probability* $> \varepsilon^2/(6\varepsilon'^2)$, *outputs a pair* (σ, τ) *with* $-A < \sigma < A$, $0 < \tau < B$ *and* $G^\sigma = \Gamma^\tau$ *in* $\mathcal{G}$. *The expected running time is* $< 2/\varepsilon \times \tau$, *where* τ *is the average running time of an execution of the proof.*

Proof. Assume that some adversary, modeled as a Turing machine $\widetilde{P}(\omega)$ running on random tape ω, is accepted with probability $\varepsilon' = 1/B^\ell + \varepsilon$. We write $Succ(\omega, (e_1, ...e_\ell)) \in \{true, false\}$ the result (successful of not) of the identification of $\widetilde{P}(\omega)$ when the challenges $e_1, ...e_\ell$ are used.

We consider the following algorithm (largely inspired from [24]):

Step 1. Pick a random tape ω and a tuple e of ℓ integers $e_1, ...e_\ell$ in $\{0, ..B - 1\}$ until $Succ(\omega, e)$. Let u be the number of probes.

Step 2. Probe up to u random ℓ-tuples e' different from e until $Succ(\omega, e')$. If after the u probes a successful e' is not found, the algorithm fails.

Step 3. Let j be one of the indices such that $e_j \neq e_j{}'$; we note y_j and $y_j{}'$ the related correct answers of $\widetilde{P}$. If $e_j > e_j{}'$ the algorithm outputs $(\sigma, \tau) = (y_j - y_j{}', e_j - e_j{}')$, otherwise it outputs $(\sigma, \tau) = (y_j{}' - y_j, e_j{}' - e_j)$.

If this algorithm does not fails, the prover is able to correctly answer two challenges e_j and $e_j{}'$ for the same commitment t_j with the answers y_j and $y_j{}'$. This means that $G^{y_j}/\Gamma^{e_j} = t_j = G^{y_j{}'}/\Gamma^{e_j{}'}$ so $G^\sigma = \Gamma^\tau$. Furthermore, $|\sigma| < A$ and $0 < \tau < B$ because integers y_i and $y_i{}'$ are smaller than A and integers e_i and $e_i{}'$ are different and smaller than B.

We now analyze the complexity of the algorithm. By assumption, the probability of success of $\widetilde{P}$ is ε' so the first step finds ω and e with the same probability. The expected number E of repetitions is $1/\varepsilon'$ and the number u of probes is equal to N with probability $\varepsilon' \times (1 - \varepsilon')^{N-1}$.

Let Ω be the set of random tapes ω such that $\Pr_e \{Succ(\omega, e)\} \geq \varepsilon' - \varepsilon/2 = 1/B^\ell + \varepsilon/2$. The probability for the random tape ω found in step 1 to be in Ω

conditioned by the knowledge that $Succ(\omega, e) = true$ can be lower bounded:

$$\Pr_{\omega,e}\{\omega \in \Omega | Succ(\omega,e)\} = 1 - \Pr_{\omega,e}\{\omega \notin \Omega | Succ(\omega,e)\} =$$

$$1 - \Pr_{\omega,e}\{Succ(\omega,e) | \omega \notin \Omega\} \times \frac{\Pr_{\omega,e}\{\omega \notin \Omega\}}{\Pr_{\omega,e}\{Succ(\omega,e)\}} \geq 1 - \left(\frac{1}{B^\ell} + \frac{\varepsilon}{2}\right) \times 1/\varepsilon' = \frac{\varepsilon}{2 \times \varepsilon'}$$

With probability $> \varepsilon/(2\varepsilon')$, the random tape ω is in Ω and in this case, by definition of the set Ω, the probability for a tuple of challenges $e' \neq e$ to lead to success is $\geq \varepsilon/2$. The probability to obtain such a tuple e' after less than N probes is $\geq 1 - (1 - \varepsilon/2)^N$.

Consequently, the probability to obtain a random tape ω in Ω and to find e' is greater than

$$\frac{\varepsilon}{2\varepsilon'} \times \sum_{N=1}^{+\infty} (1 - \varepsilon')^{N-1} \times \varepsilon' \times \left[1 - (1 - \frac{\varepsilon}{2})^N\right] = \frac{\varepsilon^2}{4\varepsilon'(\varepsilon' + \varepsilon/2 - \varepsilon \times \varepsilon'/2)} > \frac{\varepsilon^2}{6\varepsilon'^2}$$

In conclusion, the algorithm finds (σ, τ) with probability $> \varepsilon^2/(6\varepsilon'^2)$ and the total expected number of executions of the proof between $\widetilde{P}$ and a verifier is smaller than $2/\varepsilon'$. □

Finally, in the complexity theory setting, let us consider a security parameter k. All the parameters A, B, S and ℓ are viewed as functions of k.

Theorem 5 (Zero-knowledge). *The protocol is statistically zero-knowledge if $\ell \times B$ is polynomial in k and if $\ell SB/A$ is negligible.*

Proof. We describe the polynomial time simulation of the communication between a prover P and a possibly dishonest verifier $\widetilde{V}$. We assume that, in order to try to obtain information about x, $\widetilde{V}$ does not randomly choose the challenges. If we focus on the i^{th} round, $\widetilde{V}$ has already obtained data, noted $Data_i$, from previous interactions with P. Then the prover sends the commitment t_i and $\widetilde{V}$ chooses, possibly using $Data_i$ and t_i, the challenge $e_i(Data_i, t_i)$.

Here is a simulation of the i^{th} round: choose random values $e_i' \in [0, B[$ and $y_i' \in [0, A[$, compute $t_i' = G^{y_i'} \times \Gamma^{e_i'}$. If $e_i(Data_i, t_i') \neq e_i'$ then try again with another pair (e_i', y_i'), else return (t_i', e_i', y_i'). It can be formally proved that such a simulation is statistically indistinguishable from the transcript of a real proof as soon as $\ell SB/A$ is negligible:

$$\sum_{(\alpha_i,\varepsilon_i,\beta_i), i \leq \ell} \left| \Pr\{(\alpha_i,\varepsilon_i,\beta_i) = (t_i,e_i,y_i)\} - \Pr\left\{(\alpha_i,\varepsilon_i,\beta_i) = (t_i',e_i',y_i)'\right\} \right| < \frac{4\ell SB}{A}$$

Furthermore, the time complexity of the simulation if $O(\ell \times B)$ so the simulation runs in polynomial time in k if $\ell \times B$ is polynomial. □