A New and Efficient All-Or-Nothing Disclosure of Secrets Protocol

Julien P. Stern

[1] UCL Crypto Group,
Batiment Maxwell, Place du levant, 3
B-1348 Louvain-la-Neuve, Belgique
stern@dice.ucl.ac.be
[2] Laboratoire de Recherche en Informatique,
Université de Paris-Sud,
Batiment 490, F-91405 Orsay Cedex, France
stern@lri.fr

Abstract. Two-party protocols have been considered for a long time. Currently, there is a renewed effort to revisit specific protocols to gain efficiency. As an example, one may quote the breakthrough of [BF97], bringing a new solution to the problem of secretly generating RSA keys, which itself goes back to the pioneering work by Yao [Yao86]. The All-Or-Nothing Disclosure of Secrets protocol (ANDOS) was introduced in 1986 by Brassard, Crépeau and Robert [BCR87]. It involves two parties, a vendor and a buyer, and allows the vendor, who holds several secrets, to disclose one of them to the buyer, with the guarantee that no information about the other secrets will be gained. Furthermore, the buyer can freely choose his secret and has the guarantee that the vendor will not be able to find out which secret he picked. In this paper, we present a new protocol which achieves the same functionality, but which is much more efficient and can easily be implemented. Our protocol is especially efficient when a large number of secrets is involved and it can be used in various applications. The proof of security involves a novel use of computational zero-knowledge techniques combined with semantic security.

1 Introduction

The All-Or-Nothing Disclosure of Secrets protocol was introduced in 1986 by Brassard, Crépeau and Robert [BCR87]. It involves two parties, a vendor and a buyer, and allows the vendor, who holds several secrets, to disclose one of them to the buyer, with the guarantee that no information about the other secrets will be gained. Furthermore, the buyer can freely choose his secret and has the guarantee that the vendor will not be able to find out which secret he picked.

As in [BCR87], we make the following assumption: *we assume that the vendor is honest when he claims to be willing to disclose one secret, that is, he is not, for example, going to send junk or to swap several of his secrets.*

We will not discuss the issue of verification: it much depends on the trading environment and can easily be achieved by additional protocols, possibly involving third parties.

K. Ohta and D. Pei (Eds.): ASIACRYPT'98, LNCS 1514, pp. 357–371, 1998.
© Springer-Verlag Berlin Heidelberg 1998

Before presenting our protocol, it is interesting to note that the security we obtain slightly differs from what was obtained in [BCR87]. While their protocol was computationally secure for the vendor and unconditionally secure for the buyer, our protocol provides unconditional security to the vendor and computational security to the buyer.

Our paper is organized as follows: we first present related work. Then we describe the properties required by the encryption systems we are going to use, and give a few examples of such systems. We next describe the algorithm itself, and we finally discuss its complexity and its practical cost.

2 Related Work

ANDOS is also known under the name *One-out-of-t* Strings Oblivious Transfer, denoted $\binom{t}{1} - \mathcal{OT}_2^k$ when t secrets of k bits are involved. Historically, the first case considered was for $t = 2$ [Wie83]. Then came natural restrictions ($t = 2$ and $k = 1$) [EGL83], and natural extensions [BCR87] (ANDOS). A large part of the work done on Oblivious Transfer aimed at finding efficient reductions of $\binom{t}{1} - \mathcal{OT}_2^k$ to $\binom{2}{1} - \mathcal{OT}_2^1$ [BCS96]. In our context, we chose to keep the name ANDOS, as we are building our protocol without using reductions among Oblivious Transfers.

Salomaa and Santean [SS90] have designed a very efficient ANDOS algorithm when several buyers are involved. The drawback is that they need a majority of honest buyers to achieve security.

Another efficient ANDOS protocol is proposed in [NR94], but relies on *ad hoc* assumptions.

The problem of blind decoding was recently introduced by Sakurai and Yamane [SY97] and its goal appears close to ANDOS. In their scheme, the buyer is supposed to have an encrypted secret and has it decoded by the vendor in such a way that the vendor does not get any information either on the plaintext or on the vendor's private key. However, the buyer might be able to combine several secrets in a single decoding, or might try to organize an oracle attack to recover the secret key (as in [Oht97]). Furthermore, to apply the scheme in an ANDOS setting, one has to assume that the buyer can anonymously recover the ciphertext for some specific secret, which, on the web for instance, seems to need in itself an ANDOS protocol if the encrypted secrets are not widely distributed.

Finally, the schemes which are probably the most closely related to the ANDOS problem are the *Private Information Retrieval* (PIR) schemes [CGKS95], and more precisely the computational PIR schemes [CG97]. In PIR protocols, the vendor is a database, which can be modeled as holding bits of information, and the buyer is a user of the database who is willing to query the database privately. Still, there are two major differences between ANDOS and PIR protocols: PIR schemes have only considered bit-per-bit retrieval so far and do not try to enforce any security for the database (a user might recover several bits in a single query).

3 The Encryption System

We now describe the generic system that will be used throughout our protocol. It is a probabilistic encryption system with a few additional properties. The system that we use can be described as follows:

- A security parameter n from which are derived several finite domains, $(R(n), X(n), Y(n))$, which we identify with initial subsets of the integers. Thus we use $R(n)$ for $\{x : 0 < x < r(n)\}$, $X(n)$ for $\{x : 0 \leq x < x(n)\}$, and similar notation for $Y(n)$.
- A public probabilistic encryption function $f : R(n) \times X(n) \to Y(n)$, and a private decryption algorithm $g : Y(n) \to X(n)$, such that :

$$\forall (r, x) \in R(n) \times X(n) \quad g(f(r, x)) = x$$

Note that the existence of a decryption algorithm implies that the function is injective with respect to its second parameter, that is, for $(r_1, x_1), (r_2, x_2) \in R(n) \times X(n)$, if $f(r_1, x_1) = f(r_2, x_2)$ then $x_1 = x_2$.

We now describe the further requirements needed for our protocols.

1. We require that the encryption function is homomorphic, that is:

$$\forall (r_1, x_1), (r_2, x_2) \in R(n) \times X(n),$$

$$f(r_1, x_1) f(r_2, x_2) = f(r_3, x_1 + x_2 \bmod x(n))$$

   where r_3 can be computed in polynomial time from r_1, r_2, x_1 and x_2. (A similar definition was given in [BD90] when $x(n) = 2$.)
2. We ask that the encryption function has *semantic security*. Informally, this means that, for a polynomially bounded adversary, the analysis of a set of ciphertexts does not give more information about the cleartexts than what would be available without knowledge of the ciphertexts. We refer the reader to [GM84] for a formal definition.
3. We assume the existence of a reliable way to prove that the public parameters of the system were correctly constructed. This might be done with the help of a certification authority or by a zero-knowledge proof. For example, to prove the validity of a composite modulus, (e.g. the modulus is the product of *exactly* two primes), one could use the protocol described in [vdGP88].
4. The last and exotic looking property is that the number 2 is invertible in $X(n)$. The reason of this choice will appear later.
5. From the previous properties, we can deduce two more, which we will use. The first one is the existence of a "hiding" function $hide : R(n) \times Y(n) \to Y(n)$, depending only on the public parameters of the system and such that:

$$\forall (r, x) \in R(n) \times X(n),\ \forall s \in R(n) \quad hide(s, (f(r, x)) = f(sr' \bmod r(n), x)$$

   where r' can be computed in polynomial time from r, x. As a matter of fact, $hide$ can be defined by $hide(s, x) = f(s, 0)x$

6. The second property which is a consequence of the previous ones, is the existence of a way to prove that two ciphertexts represent the encryption of the same integer without revealing this integer. Let $(r_1, x_1), (r_2, x_2) \in R(n) \times X(n)$ and consider $y_1 = f(r_1, x_1)$ and $y_2 = f(r_2, x_2)$. Then, there exists r_3 such that $f(r_1, x_1)/f(r_2, x_2) = f(r_3, x_1 - x_2 \bmod x(n))$. In order to prove that $x_1 = x_2$, one can simply reveal r_3. Verification is performed by computing both y_1/y_2 and $f(r_3, 0)$ and checking their equality.

4 Sample Encryption Systems

We present several encryption systems which satisfy the requirements described in the previous section.

4.1 The Goldwasser-Micali Cryptosystem

This encryption system was introduced in [GM84]. We only give a brief sketch of the system here and refer the reader to [GM84] for details. Note that this system satisfy all properties but property 4, and thus cannot be used. We only present it as it was the first example of a probabilistic encryption scheme.

- It can only encrypt single bits. ($x(n) = 2$).
- Let N be the product of two large primes, and y be an non quadratic residue modulo N.
  The encryption function f is $f(r, x) = r^2 y^x \bmod N$.
  Decryption is done by calculating (with the factorization of N) whether or not the ciphertext is a quadratic residue.
- The semantic security of this system is proved in [GM84] under the *Quadratic Residuosity Assumption*.

4.2 The Benaloh Cryptosystem

This encryption system was derived from the previous one and introduced in [Ben87]. We only give a brief sketch of the system here and refer the reader to [Ben87] for details.

- It can encrypt several bits. ($x(n)$ usually varies from 2^{10} to 2^{40} depending on the required speed).
- Let Φ denote the Euler Totient function. Let N be the product of two large primes, choose a prime n and an integer y so that n divides $\Phi(N)$ but n^2 does not divide $\Phi(N)$ and y is a non n^{th} residue modulo N.
  The encryption function f is $f(r, x) = r^n y^x \bmod N$.
  Decryption is done by calculating (with the knowledge of $\Phi(n)$) the residuosity class of the ciphertext modulo n.
- The semantic security is implicit in [Ben87] under the *Prime Residuosity Assumption*.

4.3 The Naccache-Stern Cryptosystem

This system was recently introduced in [NS98]. We only give a brief sketch of the system here and refer the reader to [NS98] for details.

- It can encrypt several bits ($x(n)$ is usually around 2^{160}).
- Let Φ denote the Euler Totient function. Let N be the product of two large primes, choose $n_1, \ldots, n_p$ to be small primes so that for all $i \in \{1, \ldots, p\}$, n_i divides $\Phi(N)$ but n_i^2 does not divide $\Phi(N)$. Also choose y to be, for all i, a non n_i^{th} residue modulo N. Set $n = \prod_{i=1}^{p} n_i$.
  The encryption function f is $f(r, x) = r^n y^x \bmod N$.
  Decryption is done by calculating (with the knowledge of $\Phi(n)$) the residuosity class of the ciphertext modulo each of the n_i and by recovering the cleartext by means of the Chinese remainder theorem.
- The semantic security is proved in [NS98] under the *Prime Residuosity Assumption.*

4.4 The Okamoto-Uchiyama Cryptosystem

We only give a brief sketch of the system here and refer the reader to [OU98] for details.

- It can encrypt several bits ($x(n)$ is usually around 2^{160}).
- Let p and q be two large primes with p and $q - 1$ relatively prime. Let $N = p^2 q$. Let $y \in (\mathbf{Z}/n\mathbf{Z})$ such that the order of $y^p \bmod p^2$ is p.
  The encryption function f is $f(r, x) = r^N y^x \bmod N$.
  Decryption is performed by raising the encrypted message to the power $p - 1$ and using the fact that it is easy to compute discrete logarithm in the subgroup $\{x \in (\mathbf{Z}/p^2\mathbf{Z})^* | x \equiv 1 \bmod p\}$.
- The semantic security is proved in [OU98] under the *p-subgroup Assumption.*

5 The new ANDOS Protocol

5.1 Overview

An ANDOS protocol involves two participants. The first one, who holds several secrets, will be called the vendor. The second one, who is willing to buy one of these secrets, will be called the buyer.

Our basic idea is to have the buyer send an encrypted index of the secret he is willing to buy, and to let the vendor perform all computation on his side. The computation of the vendor takes as entries the buyer's index and his own family of secrets. Provided the buyer's index is valid, the result of this computation will be the corresponding secret, encrypted in such a way that only the buyer can properly decrypt it. The core of the protocol is to efficiently prove the validity of the index. Apart from the use of probabilistic homomorphic encryption, this proof method is our main technical contribution. Similar proofs have already

been investigated in [PS96], but turn out to be much less efficient than the proposed one.

The protocol is a one-round protocol, plus an additional interactive proof of validity.

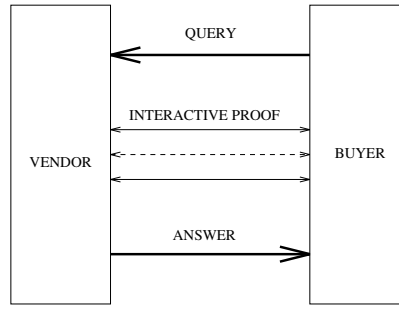

Fig. 1. The ANDOS protocol: the buyer sends his query, proves its validity, and gets the secret he selected

5.2 Preliminary Zero-Knowledge Proof

We present here a protocol which achieves the following: given

- a probabilistic encryption system $\mathcal{E}$ verifying the properties stated in 3,
- an integer t,
- a security parameter k (not necessarily related to the security parameter of $\mathcal{E}$),

we obtain the following:

- the buyer is able to convince the vendor that a sequence of t values represents the encryption of $t-1$ zeros and a single 1, say $x_1, \ldots, x_t$, encrypted under $\mathcal{E}$,
- the protocol does not reveal any information on the index of the non-zero element,
- any cheating buyer who attempts to perform the protocol with a sequence which is not of the proper form will fail with probability $1-(4/5)^k$,

The protocol goes as follows: The buyer first creates an encryption system $\mathcal{E}$ verifying the properties stated in 3, sends its public parameters to the vendor and proves their validity. He then picks t random numbers $h_1, \ldots, h_t$ in $R(n)$, and sends to the vendor the sequence σ of t values $v_1 = f(h_1, x_1), \ldots, v_t = f(h_t, x_t)$, Then, they repeat k times the following steps:

1. The buyer sends a zero and a one in encrypted form.
2. The vendor picks a random number between 1 and 5, and:
   Case 1 if the number is 1, asks the buyer to reveal the cleartexts corresponding to the encrypted pair.
   Case 2 else, he splits the set $\{1, \ldots, t\}$ in two distinct random subsets A and B, computes $a = \prod_{i \in A} v_i, \mathrm{mod} y(n)$ $b = \prod_{i \in B} v_i \bmod y(n)$, sends the buyer the sets A and B and asks him to prove (using property 5) that the pair (a, b) represents the same numbers as the encrypted pair he has sent in step 1.

5.3 Correctness

If the buyer is playing the protocol fairly (with a sequence σ of the correct form), he will be able to answer all the vendor's questions.

Assume that the buyer is dishonest and is not playing the protocol fairly, but uses a sequence which does not have the requested form. We will show that, whatever strategy he uses, the cheating buyer cannot answer one round correctly with probability greater than 4/5.

Let us first deal with the case where the sum (mod $x(n)$) of all the integers in the buyer's sequence is not 1. Recall that property 2 implies that multiplying several encrypted numbers gives the encryption of their sum (mod $x(n)$).

When **case** 2 occurs, whatever the vendor picks for subsets A and B, the cleartexts corresponding to a and b cannot be 0 and 1. Hence, the buyer cannot answer **case** 1 and **case** 2 at the same time.

Now, assume the sum of the numbers in the buyer's sequence is 1, but that the sequence is not valid. Then at least two integers in the sequence are non-zero.

Lemma 1. *Assume that* 2 *is invertible in* $X(n)$. *Let* $\alpha_1, \ldots, \alpha_t$ *be in* $X(n)$ *with* α_i *and* α_j *non zero. Let* A *be a subset of* $\{1, \ldots, t\}$ *such that* $i \notin A, j \notin A$. *Define* $A_0 = A$, $A_1 = A \cup \{i\}$, $A_2 = A \cup \{j\}$, $A_3 = A \cup \{i, j\}$. *Then, none of the four pairs*

$$p_l = \{\sum_{q \in A_l} \alpha_q, \sum_{q \notin A_l} \alpha_q\}$$

can be equal.

Proof. Let $p_0 = \{u, v\}$. Then $p_1 = \{u + \alpha_i, v - \alpha_i\}$, $p_2 = \{u + \alpha_j, v - \alpha_j\}$, $p_3 = \{u + \alpha_i + \alpha_j, v - \alpha_i - \alpha_j\}$. Assume all the pairs coincide, then,

- From p_0 and p_1, we deduce that $\alpha_i = v - u$ (or that $\alpha_i = 0$).
- From p_0 and p_2, we deduce that $\alpha_j = v - u$ (or that $\alpha_j = 0$).
- So, we can rewrite $p_3 = \{u + 2\alpha_i, v - 2\alpha_i\}$. Hence, from p_0 and p_3, we deduce that $2\alpha_i = v - u$ (which leads to $\alpha_i = 0$) or that $2\alpha_i = 0$ (which also leads to $\alpha_i = 0$ as 2 is invertible).

Hence, all cases lead to a contradiction.

From the lemma, it follows that a buyer who can answer **case** 1 is only able to satisfy at most 3/4 of the queries in **case** 2. Hence, at the end of the protocol, the vendor should be convinced, with probability $1-(4/5)^k$, that the encrypted sequence he received is made of one 1 and $t-1$ zeros.

Remark 1. We can note that the previous lemma does not work if we use the Goldwasser-Micali encryption scheme (where $x(n) = 2$). And, as a matter of fact, when using this scheme, any sequence containing a odd number of 1 will pass the protocol.

The proof that the protocol does not leak any information on the position of the non-zero element can be found in the appendix.

5.4 The Protocol

Let t be the number of secrets of the vendor, and let $s_1, \ldots, s_t$ be the secrets themselves.

Initialization The buyer creates an encryption system $\mathcal{E}$, which verifies the properties stated in 3, sends the public parameters of this system and proves their validity or provides a certificate.

Secret Selection The buyer sends a sequence of t numbers, encrypted under $\mathcal{E}$, and gives a zero-knowledge proof that there is actually $t-1$ zeros and a one among them. Let $f(r_1, x_1), \ldots, f(r_t, x_t)$ be the t encrypted values.

Vendor computation The vendor picks a random h and computes:

$$S = hide(h, \prod_{i=1}^{t} f(r_i, x_i)^{s_i}) = f(hr' \bmod r(n), \sum_{i=1}^{t} x_i s_i \bmod x(n))$$

where r' can be computed in polynomial time from the r_i and the x_i.

Secret recovery Upon receipt of S, the buyer can retrieve $\sum_{i=1}^{t} x_i s_i \bmod x(n)$ which is equal to one of the s_i, depending on his initial choice.

5.5 Correctness

The security for the buyer is a consequence of the security of the previous proof. Now, let us analyze what additional information the buyer could try to get.

The questions of the vendor during the interactive proof of knowledge are coming from a random source which is unrelated from his secrets. Hence, the buyer might only be able to gain extra information (than the secret he retrieved) by analyzing the final reply string.

When decrypting this reply string, the buyer gets the value of one secret, which cannot be of any help to figure out what other secrets are. To see this, consider the second parameter of the encryption function, namely: $Z = hr' \bmod r(n)$. Let us fix the r_i and the s_i. r' is thus fixed. When h ranges over $R(n)$, Z ranges over $R(n)$ too. Hence, the distribution of Z on all the possible values of

h is uniform. This means that if we only fix the r_i and the very s_a retrieved by the buyer, the distribution of Z on all the possible values of h and the other s_i is also uniform. Consequently, the buyer is unable to get any information from the analysis of S.

5.6 Recovery of Large Secrets

A problem occurs in case the secrets are larger than $x(n)$. As a matter of fact, our protocol only allows the recovery of a secret modulo $x(n)$. Nevertheless, this difficulty can easily be avoided. Simply split each secret into blocks of size lower than $x(n)$, and apply the vendor computation, with the *same* query string, on the different sequences of blocks. The fact that the same query string is used provides the security, and allows a very small overhead: the query cost remains the same, and the reply cost is simply multiplied by the size of the larger secret divided by $x(n)$.

5.7 The Recursive Protocol

The complexity of the protocol is $O(t)$, while in fact, we are only wishing to send an index, which could be achieved in $O(\log t)$ without encryption (nor privacy). By using the same trick as Kushilevitz and Ostrovsky [KO97], we can reduce the query cost to $y(n)2^{O(\sqrt{\log t})}$ while the reply cost is only increased to $y(n)2^{O(\sqrt{\log t})}$. It is interesting to note that, by using the Naccache-Stern cryptosystem instead of the quadratic residuosity cryptosystem, we slightly improve their complexity result.

We now proceed as follows:

Let t be the number of secrets of the vendor, and let $s_1, \ldots, s_t$ be the secrets themselves. Fix an integer m and split the t secrets into t/m buckets of m elements. The distribution of the secrets in each bucket is known by both participant (say the m first secrets in the first bucket, and so on), and the order of the secrets in each bucket is also fixed by the protocol.

We now apply almost the same protocol as before, with a sequence of m integers:

- The buyer sends his query and proves its validity.
- The vendor performs the previous computation, for each bucket, *with the same query string* but keeps all the results.

By applying the vendor computation independently to each bucket, we have virtually extracted a set of t/m secrets out of t secrets. So, we can apply the protocol again to this new set of secrets until the size of the set is less than m. The sole difference is that the secrets in the new set are encrypted and will necessarily be of size greater than $x(n)$. This is not a problem as we know how to recover large secrets. Ultimately, when the number of secrets in the set is less than m, we apply the basic protocol.

Of course, the size of the secrets will grow at each step. After, t/m steps, their size will be $(y(n)/x(n))^{t/m}$ times bigger than the original ones, so the value of m should be chosen to obtain the best tradeoff between the size of the query strings, and the reply string.

Let us now analyze the complexity of this algorithm and discuss the values for m.

5.8 Complexity

Let L be the number of steps of the protocol. At each step we apply the protocol with $m = m_i$ (hence $t \leq \prod_{i=1}^{L} m_i$). Let us choose all the m_i equal, that is $m_i = t^{1/L}$.

- The size of each of the query strings is: $y(n)m_i$ bits, so the total communication cost of the queries is: $y(n) \sum_{i=1}^{L} m_i = y(n)t^{1/L}L$.
- If k is fixed, the communication cost of the zero-knowledge proof is of the form $C + Dm_i$, so, the cost of the zero-knowledge proofs is $(C + Dt^{1/L})L$, where C and D are some constants (which depend on k).
- The communication cost of the final reply string, (if we assume that the size of the original secrets is less than $x(n)$), is $y(n)(y(n)/x(n))^L$.

The global communication cost of the protocol, for t secrets, is

$$CC = (t^{1/L}(y(n) + D) + C)L + y(n)(y(n)/x(n))^L$$

When we use the Naccache-Stern or the Okamoto-Uchiyama cryptosystem, the ratio $y(n)/x(n)$ remains constant when n grows. Let $E = y(n)/x(n)$.

If we solve for the value of L which makes both terms equal, we get:

$$L = O(\sqrt{\log t})$$

which gives a complexity of:

$$CC = y(n)2^{O(\sqrt{\log t})}$$

This result is to be compared with the complexity obtained in [KO97], which is $2^{O(\sqrt{\log t \log y(n)})}$.

6 Applications

The possible applications of our ANDOS protocol are numerous, and we only briefly comment on them. It allows, for instance, the implementation of multi-players mental games as claimed in the original ANDOS paper [BCR87].

It can also be used to implement a pay-per-access database with private queries. This case is a very straightforward application. The implementation issues, which are discussed in the next section, show that this would lead to a really practical protocol.

It can also serve as a building block for more complicated protocols, such as electronic voting, where the basic idea is to use an ANDOS protocol to distribute "eligibility tokens", which are used later to prove one's right to vote [NSS91,Ive91].

This protocol can also be used as an important building block in asymmetric traitor tracing [CFN94,Pfi96] or asymmetric fingerprinting schemes [PS96,PW97]. In traitor tracing, the buyer can use our protocol in order to get a part of his key, say the half, while the other half would be chosen and kept by the vendor along with the query string. Tracing would be performed by comparing a set of recovered keys with the known halves, trial can be achieved by exhibiting the recovered key and the query string. An innocent buyer can prove himself innocent by revealing his cleartexts to show they do not match the recovered key. Details about these families of protocols, resistance against collusions, and good choices for sets of keys can be found in [Pfi96,CFN94,BS95].

For fingerprinting purposes, we can apply the very same process, but instead of sending back the selected secret, which would be chosen by the buyer among a set of acceptable fingerprints, the vendor would insert the (encrypted) secret inside the data to be fingerprinted, using the homomorphic properties of the encryption function (Prop. 2) and send this data to the buyer. The reader can find additional precision in [BM97] where the authors show the existence of a protocol for collusion-secure asymmetric fingerprinting with the help of what they define as a "committed" ANDOS protocol. This means that the buyer has to commit to the secret he is willing to buy at the beginning of the protocol, which is the case in our scheme.

7 Implementation Issues

Our protocol is very efficient for real-life applications, since it does not hide a very large constant beyond its asymptotic behavior. We give here a few examples of costs when the Naccache-Stern cryptosystem is used.

A reasonable size for the modulus N, in order to prevent factorization, is 640 bits. As suggested in [NS98], we will choose n around 2^{160}.

Note that in order to perform an implementation of the protocols, several small tricks can be used to improve the communication cost by a constant factor. For instance, at the beginning of each round of the zero-knowledge proof, the buyer can send one encrypted bit, say $f(h_1, x_1)$, instead of two, and then, in **case** 2, can show the mapping between the pair $\{a, b\}$, and the pair $\{f(h_1, x_1)/f(1, 1), f(1, 1)/f(h_1, x_1)\}$. We assume that these tricks are used when computing the real costs.

The table below summarizes the communication costs, in kilobits for different values of the number of secrets and of the security parameter of the zero-knowledge proof.

	$t = 1000$	$t = 10000$	$t = 100000$	$t = 1000000$
$k = 20$	94	128	176	224
$k = 30$	119	166	214	275
$k = 40$	145	205	252	326

With a 33600 modem, the protocol takes between 3 and 10 seconds to complete. This is a reasonable time to retrieve a key of up to 160 bits. If this key is used to decrypt a piece of data that has previously been retrieved anonymously (say on a newsgroup), the key retrieval time is likely to be small compared to the time needed to retrieve the encrypted data.

If one is willing to directly retrieve large data, (e.g. images), the overhead due to our protocol is roughly a multiplicative factor of 16, which is important but reasonable. Of course, for too large data, the vendor computation becomes very important and might not be negligible any more compared to the transmission time.

8 Conclusion

The primary contribution of this paper is a new and efficient zero-knowledge protocol which allows to prove that a committed string contains one 1 and otherwise 0's. We would like to stress that this protocol is an important building block for many schemes. ANDOS in a straightforward manner, and from there asymmetric fingerprinting, asymmetric traitor tracing or electronic voting. As pointed out in the last section, it is efficient enough to allow, for the first time, a viable and provably secure implementation of an ANDOS protocol.

Acknowledgments

I am grateful to Jean-Jacques Quisquater for fruitful advices and many helpful hints, as well as to my fellow students at the UCL Crypto group for valuable discussions. I also wish to thank Markus Jakobsson and Miklos Santha for proofreading the manuscript.

References

BCR87. G. Brassard, C. Crépeau, and Jean-Marc Robert. All-or-nothing disclosure of secrets. In A.M. Odlyzko, editor, *Proc. CRYPTO 86*, pages 234–238. Springer-Verlag, 1987. Lecture Notes in Computer Science No. 263.

BCS96. G. Brassard, C. Crépeau, and M. Sántha. Oblivious transfers and intersecting codes. In *IEEE Transactions on Information Theory*, pages 1769–1780, 1996.

BD90. M. V. D. Burmester and Y. Desmedt. All languages in NP have divertible zero-knowledge proofs and arguments under cryptographic assumptions. In *Advances in Cryptology — Eurocrypt '90*, pages 1–10, 1990.

Ben87. J. D. C. Benaloh. Verifiable Secret-Ballot Elections. PhD thesis, Yale's University, 1987.

BF97. D. Boneh and M. Franklin. Efficient generation of shared RSA keys. In B. S. Kaliski Jr, editor, *Proc. CRYPTO '97*, number 1294 in Lecture Notes in Computer Science, pages 425–439, Springer-Verlag, 1997.

BM97. I. Biehl and B. Meyer. Protocols for collusion secure asymmetric fingerprinting. In *STACS '97*, pages 399–412, 1997.

BS95. D. Boneh and J. Shaw. Collusion-secure fingerprinting for digital data. pages 452–465. Springer, 1995. Lecture Notes in Computer Science No. 963.

CFN94. B. Chor, A. Fiat, and M. Naor. Tracing traitors. In Y. G. Desmedt, editor, *Proc. CRYPTO '95*, pages 257–270. Springer, 1994. Lecture Notes in Computer Science No. 839.

CG97. B. Chor and N. Gilboa. Computationally private information retrieval (extended abstract). In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 304–313, El Paso, Texas, 4–6 May 1997.

CGKS95. B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *36th Annual Symposium on Foundations of Computer Science*, pages 41–50, Milwaukee, Wisconsin, 23–25 October 1995. IEEE.

EGL83. S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. In R. L. Rivest, A. Sherman, and D. Chaum, editors, *Proc. CRYPTO 82*, pages 205–210, New York, 1983. Plenum Press.

FFS88. U. Feige, A. Fiat, and A. Shamir. Zero knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, 1988.

GM84. S. Goldwasser and S. Micali. Probabilistic encryption. *JCSS*, 28(2):270–299, April 1984.

Ive91. K. R. Iversen. A cryptographic scheme for computerized general elections. In J. Feigenbaum, editor, *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 405–419. Springer-Verlag, 1992, 11–15 August 1991.

KO97. E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval (extended abstract). In *38th Annual Symposium on Foundations of Computer Science*, pages 364–373, Miami Beach, Florida, 20–22 October 1997. IEEE.

NR94. V. Niemi and A. Renvall. Cryptographic protocols and voting. In *Result and Trends in Theoretical Computer Science*, number 812 in Lecture Notes in Computer Science, pages 307–316, 1994.

NS98. D. Naccache and J. Stern. A new candidate trapdoor function. To appear in *5th ACM Symposium on Computer and Communications Security*, 1998.

NSS91. H. Nurmi, A. Salomaa and L. Santean. Secret ballot elections in computer networks. In *Computers and Security*, volume 10, pages 553–560, 1991.

Oht97. K. Ohta. Remarks on blind decryption. In *Information Security Workshop*, pages 59–64, 1997.

OU98. T. Okamoto and S. Uchiyama. An efficient public-key cryptosystem. In *Advances in Cryptology—EUROCRYPT 98*, pages 308–318, 1998.

Pfi96. B. Pfitzmann. Trials of traced traitors. In R. Anderson, editor, *Information Hiding*, volume 1174 of *Lecture Notes in Computer Science*, pages 49–64, Springer-Verlag, 1996.

PS96. B. Pfitzmann and M. Schunter. Asymmetric fingerprinting (extended abstract). In Ueli Maurer, editor, *Advances in Cryptology—EUROCRYPT 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 84–95. Springer-Verlag, 12–16 May 1996.

PW97. B. Pfitzmann and M. Waidner. Asymmetric fingerprinting for larger collusions. In *4th ACM Conference on Computer and Communications Security*, 1997.

SS90. A. Salomaa and L. Santean. Secret selling of secrets with several buyers. In *42th EATCS Bulletin*, pages 178–186, 1990.

SY97. K. Sakurai and Y. Yamane. Blind decoding, blind undeniable signatures, and their applications to privacy protection. In R. Anderson, editor, *Information Hiding*, pages 257–264. Springer-Verlag, 1997. Lecture Notes in Computer Science No. 1174.

vdGP88. J. van de Graaf and R. Peralta. A simple and secure way to show the validity of your public key. In C. Pomerance, editor, *Proc. CRYPTO '87*, pages 128–134. Springer-Verlag, 1988. Lecture Notes in Computer Science No. 293.

Wie83. S. Wiesner. Conjugate coding. In *Sigact News*, volume 18, pages 78–88, 1983. Original manuscript written circa 1970.

Yao86. A. C. Yao. How to generate and exchange secrets. In *Proc. 27th IEEE Symp. on Foundations of Comp. Science*, pages 162–167, Toronto, 1986. IEEE.

A Proof of Claims of Section 5.2

We now show that the protocol does not leak any information on the position of the non-zero element by constructing a simulator. We use a zero-knowledge type argument. (For more information on zero-knowledge proofs and simulators, see [FFS88]). However, it is important to stress that we do not prove in zero-knowledge that the commitment σ is correct. This would entail a simulation of the protocol with input σ. Rather, we show how to simulate (up to computational indistiguishability) the *whole* protocol, including the commitment creation step. In other words, we show that the *whole* protocol is independent (up to computational indistiguishability) of the choice of the unique index of the buyer.

We will do this in several steps. Consider the following simulator, that we call $S(q, m)$: this simulator is given two inputs, m an integer in $Y(n)$, which is an encryption of 1, and q an index in $\{1, \ldots, t\}$. It works as follows:

– First, he creates a sequence of t integers, all of which represent encryptions of 0, except for the one in position q which is simply the application of the

function *hide* on m. Let $v_1 = f(r_1, x_1), \ldots, v_t = f(r_t, x_t)$ be these integers (with $v_q = hide(r_q, m)$).
- Then the simulator anticipates the question of the vendor: he picks a random number between 1 and 5. He also chooses two random numbers h_1 and h_2 in $R(n)$.
- If he picks 1, he sends the pair $\{f(h_1, 0), f(h_2, 1)\}$ else he sends the pair $\{f(h_1, 0), hide(h_2, m)\}$.
- He waits for the vendor question. If his guess is not correct, he resets and starts again, otherwise:
- In **Case** 1, he reveals the cleartext corresponding to the encrypted pair.
- Else, he receives two subsets A and B which are a partition of $\{1, \ldots, t\}$. Suppose that q is in A. Then, the simulator sends the pair $\{\frac{\prod_{i \in A} r_i}{h_2}, \frac{\prod_{i \in B} r_i}{h_1}\}$; else he swaps h_1 and h_2.

We will now prove that, for any pairs $(q, q') \in \{1, \ldots, t\}$, the outputs of the simulators $S(q, m)$ and $S(q', m)$ are indistiguishable by a polynomially bounded adversary.

Fix $q \in \{1, \ldots, t\}$. The simulator $S(q, m)$ simulates the protocol, (but this does not prove anything as he uses as input the index of the non-zero element in the initial sequence, together with a ciphertext of 1).

Now consider the simulator $S(q, m')$ when m' is an encryption of 0. The existence of a polynomial (probabilistic) time algorithm which distinguishes between the outputs of both simulators would directly yield an algorithm which distinguishes between the encryption of a 0 and the encryption of a 1. This would go against the assumed semantic security of the system (property 3).

But, when m is a ciphertext of 0 the index q does not play any role any more, as each of the t integers of the initial sequence represents an encryption of 0. Let q' be in $\{1, \ldots, t\}$, $q' \neq q$. The output of the simulator $S(q', m')$ is thus indistinguishable from $S(q, m')$. And finally, with the same argument based on the semantic security, the output of $S(q', m')$ is indistinguishable from $S(q', m)$. This shows that our proof does not leak any information on the index of the non-zero element.

Remark 2. The simulators we have considered work in *expected* polynomial time, while the assumption made on the semantic security only requires resistance against polynomial time algorithms. Actually, by limiting the number of resets to say kn (where k and n are the two security parameters in use), we complete the simulation of the k rounds with probability exponentially close to 1. This is enough for computational indistinguishability.