ANONYMOUS AND VERIFIABLE REGISTRATION IN DATABASES

Jørgen Brandt

Ivan Bjerre Damgård[1]

Peter Landrock

Dept. of Mathematics and Computer Science, Aarhus University
Ny Munkegade,
DK 8000 Aarhus C,
Denmark.

Abstract

Methods are given by which personal data about a large number of individuals can be registered in a large central database without having to trust this register not to give away information linked to a given individual. Personal information arriving from many different sources can be placed correctly in the register. The registration is done in a verifiable way: Each individual can be given access to the register to check that his information is correct, and can even, if he chooses to do so, prove to anyone that he is or is not identical to a given person in the register. This can all be done without compromising the anonymity of any other individual.

1. Introduction

Consider a set of institutions $D_1, \ldots, D_n$, which collect information on a large number of individuals. Examples could be tax authorities, banks, hospitals etc. The institutions would like to set up a large common register C, which is to contain all information from all institutions. There may be numerous reasons for this, C may be convenient for economical or practical reasons, or it may be just a temporary register which is set up for statistical purposes.

This raises of course some security problems: the individuals may be willing to trust each of the D_i, but unwilling to accept a new central register, since

1) Outsiders can now get access to a complete set of personal data about anyone, just by breaking into one database; and
2) The D_i's, who have *legal* access to C may now read data about any individual, including those that they have had no contact with before.

[1]This research was supported by the Danish Natural Science Research Council.

How can we make C secure against unwanted use of the information? It is well known that preventing access, physical or otherwise, to a database is very hard and expensive. A cryptographic solution, however, can make the information useless to intruders, and therefore seems a better alternative.

Recall that in this case the personal information itself is not secret, the confidential part is the linking of names to particular records in the register. What we need is therefore a system by which the D_i's can send information to C in such a way that data arriving from different places concerning the same person can be identified as such, but without this giving away the true identity of the individual involved. In other words, we want the registration to be anonymous: given an individual and a person registered in C, it should be hard to tell whether they are identical. Moreover, it is desirable that the system is *verifiable*, i.e. an individual i can be given access to C to check that his data are correct, and even more important: if needed, i can produce a *proof* that he is or is not identical to a given person registered in C. Of course, this must all be done without compromising the anonymity of anybody else.

2. Related Work

Other researchers, in particular Chaum [Ch], have designed systems to prevent the linking of a large amount of personal data. Chaum's system is based on each individual having different *pseudonyms* with each organisation they talk to. This makes the information unconditionally unlinkable. On the other hand, data which is to be exchanged between organisations must travel through the individual they apply to. With a nationwide database, this may not be a practical solution. In our system the individuals are known by their real name in the institutions we have to begin with $(D_1, \ldots, D_n)$. This means of course that the individuals must trust the D_i's and that we loose the unconditional unlinkability. On the other hand, information can now be sent to the new register directly, and since our system is identity based, it can be verifiable. This is much harder to achieve with a system where individuals choose their own pseudonyms at random: how can person i prove that he did or did not choose this particular random number?

3. Our Solution

We assume that each person is known to each D_i by some unique piece of information, like name, address, ect. For person j this will be called $ID(j)$. Consider now a solution where data will be sent to C such that information about the individual j is accompanied by an "encryption" of $ID(j)$, i.e. the image of $ID(j)$ under some suitable function F. We let J denote the set of all possible individuals. We assume that this set is very large, so that the set of individuals registered in C at any given time is of negligible size compared to $|J|$.

We can now formulate the properties we need a little more precisely:

- Anonymity: given $F(ID(j))$ and the unordered pair $(ID(j), ID(j'))$, it is hard to decide whether $ID(j)$ or $ID(j')$ is the preimage of $F(ID(j))$.
- Verifiability: for each $ID(j)$, there exists a *witness*, $w(ID(j))$ with the property that $ID(j)$ and $F(ID(j))$ are easily computable from $w(ID(j))$.
- Independence: The anonymity condition still holds, even when one is also given a set of pairs $\{(ID(i), w(ID(i))) \mid i \neq j, j'\}$, where the i's are chosen at random from J.

The independence condition is meant to protect against the case where an enemy knows the identity of some registered individuals. The condition says that this does not help him to find other identities. Note, however, that since we assume that the given identities are randomly distributed in J, the condition does not cover the case where an enemy can choose freely individuals for which he would like to see corresponding F-values (c.f. known plaintext versus chosen plaintext attacks on a crypto system).

The verifiability condition assigns to each individual a unique witness, which can thought of as a certificate of the connection between corresponding ID and F-values. This allows an individual to prove to anyone that he is or is not identical to a given person registered in C. More details can be found in Section 5.

The anonymity condition is as restrictive as possible: it says that even when given that an unknown person registered in C is identical to one out of two individuals, it is still hard to tell which one. This and the independence condition means that some of the more obvious solutions will not work:

Consider for example using as F a publicly known one way function. This means at least that one cannot compute j from $F(ID(j))$. But since it is trivial to test from $ID(j')$ and $F(ID(j))$ whether $j = j'$, the anonymity condition is violated. One way to repair this could be to use a function depending on some secret parameter, like a pseudo random function [GGM] or a conventional cipher, i.e. setting $F = f_K$, where K is secret. This may satisfy the anonymity condition, but the only way we can get verifiability is by setting $w(ID(j)) = K$ for all j, which clearly violates the independence condition.

The solution we suggest can be informally described as follows: Select a trapdoor one way permutation f and a one way function g with the same domain as f. By redefining ID, we make sure that $ID(j) \in domain(f)$ for all j.

We describe one way of doing this in the following: To be specific, let $ID(j)$ consist of a number of fields, such as *firstname*(j), *secondname*(j), *street*(j), *city*(j), etc., where *firstname*(j) belongs to some set *FIRSTNAMES*, and similarly for the other fields. This makes $ID(j)$ an element of

$$J = FIRSTNAMES \times SECONDNAMES \times STREET \times \cdots$$

considered as a concatenation of ASCII characters. The set J has a certain redundancy, and using an ideal encoding rule $c : J \rightarrow \{0,1\}^k$, which is nearly a bijection for

$$k = \log_2(|FIRSTNAMES|) + \log_2(|SECONDNAMES|) + \cdots$$

we may represent the set of possible ID 's as binary strings of length k. The parameter k should be chosen such that $domain(f) = \{0,1\}^k$. In practice, k will be a security parameter, and the number of fields in ID must be chosen accordingly. Also, we must of course admit that the cardinality of $domain(f)$ will not in general be an exact 2-power, so we have to content ourselves with approximations in practice.

With this scheme, choosing a random person in J and applying c produces an (almost) uniformly distributed element in $domain(f)$. Moreover, it is a reasonable assumption that choosing a random set of strings corresponding to persons registered in the data base gives a good approximation to a uniform choice from all of J, where "good" is defined relative to the behavior of polynomial time algorithms using the strings as input. More specifically, we are assuming that no feasible algorithm is able to exploit the fact that the individuals in C are not really uniformly chosen, but are selected by some specific (incredibly complicated) random process.

We then set $F(ID(j)) = g(f^{-1}(ID(j)))$ and $w(ID(j)) = f^{-1}(ID(j))$.

Actually this definition is a bit too restrictive. It is clearly sufficient that both $ID(j)$ and $F(ID(j))$ are easily computable from $w(ID(j))$, and with some choices of f and g, there are other ways to meet this condition.

Theorem 3.1
With F, w and ID defined as above, the verifiability and independence conditions are satisfied.

Proof.
Given $w(ID(j))$, one can directly compute $F(ID(j)) = g(w(ID(j)))$. Thus the verifiability condition is satisfied. With the definition of ID given above, we may assume that selection of a random individual i will produce an element $ID(i)$ uniformly distributed in the domain of f. Therefore a randomly chosen set $\{(ID(i), w(ID(i))\}$ can always be produced without knowing the identity of any individual, just by starting with a set of randomly chosen witnesses and computing f on each of them. Therefore an algorithm which would break the anonymity condition given a set of corresponding identities and witnesses can easily be modified to do without this just by producing the required set from schratch as above. $\square$

It is much harder to say something conclusive about the anonymity condition. It is clearly a necessary condition that it is hard to compute x from $F(x)$ and vice versa. But it is not necessarily true that in order to solve the *testing* problem, i.e. find out whether $x = x'$ given x and $F(x')$, one must be able to actually compute F or F^{-1}. For example, it is proved below that if both f and g are independently selected trapdoor permutations, then F is in fact hard to compute "both ways". Suppose now that there exists a large class of trapdoor one way permutations which commute for all choices of trapdoor, which sounds, if not likely, then at least conceivable. Then if f and g are chosen from this class, it is trivial to see that testing is always easy. Another trivial necessary condition is therefore that f and g do not commute.

One could of course try to prove that testing is equivalent to computing function values for all functions. But there is little hope of this: in [BoLa] it is proved that this is equivalent to a long standing, and hard problem about separation of complexity classes. Indeed if the problem was settled such that testing was equivalent to computing for ALL functions, then functions like discrete log and squaring modulo a composite would not be one way!

Thus, for the concrete constructions we propose, all we can say is that the necessary conditions are satisfied, and that independent choice of f and g does seem to be sufficient to ensure anonymity in those cases.

It remains an open problem, however, to formulate precisely what kind of "independence" one needs between f and g to get anonymity in general.

As a final observation about the anonymity condition, consider the obvious attack starting with a randomly chosen witness w and computing $f(w)$, which will be $ID(j)$ for some j, and $g(w)$, which is equal to $F(ID(j))$. If j happens to be registered in C, we have broken the anonymity of j. This attack will not work, however, because we have assumed that the number of individuals actually registered is negligible compared to the number of possible individuals in J. Thus there is only a negligible probability that this attack will result in a known identity for any "useful" individual. This does not exclude that there could be some way to cleverly *choose* w in a way that would ensure that $f(w)$ was in fact ID of somebody in C, corresponding to what one tries to do in an attack on a signature scheme with redundancy build into the messages. Note, however, that when such redundancy schemes can be cracked, it is always because there exists some simple algebraic description of the set of valid messages. This description, together with for example the multiplicative property of RSA, can then be used to breake the system. It seems extremely unlikely, though, that such a description would exist for the set of individuals registered in C at some random point of time. Unfortunately, for precisely the same reason, it seems to be very hard to actually prove something about this question!

But at least, we can prove that with right choice of f and g, F is hard to compute in "both directions":

Theorem 3.2
Suppose F is constructed using randomly and independently chosen trapdoor permutations f and g. Suppose also that it is infeasible to compute f^{-1} and g^{-1} for more than a negligible fraction of the possible choices of f and g. Then both $F = gf^{-1}$ and $F^{-1} = fg^{-1}$ are infeasible to compute for more than a negligible fraction of the possible choices of pairs (f,g).

Proof.
Suppose we have an efficient algorithm for computing F. Then this algorithm can be used to compute f^{-1} for a randomly chosen f with unknown trapdoor as follows: select a g with known trapdoor at random, and run the algorithm on F constructed from f and g. By assumption, the algorithm can compute F-images with nonnegligible probability, and for each x for which it tells us what $F(x)$ is, we can use the trapdoor for g to compute $f^{-1}(x) = g^{-1}F(x)$. The case with F^{-1} is symmetric. □

There is a price to pay in order to be able to prove that F and F^{-1} have the claimed properties, namely the assumption that g is trapdoor, which introduces the risk of having the trapdoor revealed to an enemy. One can do away with this by developing systems, where g, and therefore F is a one way function with no (known) trapdoor. This would mean that even organisations with maximal information on the system would be unable to ''decrypt'' randomly chosen identities in C, although knowledge of the trapdoor for f would enable them to test given identities against F-values. This would be of little use to an enemy, however, if C was only willing to release data on an individual to D_i, if D_i had previously provided data on that individual. This could be implemented by including a protocol by which any D_i could indentify itself to C before getting access to any data.

One way to implement the system in practice is to assume a trusted center which selects f and g together with the trapdoor information for f, computes and sends secretly $f^{-1}(ID(j))$ to each j, then forgets the trapdoor information and stops functioning. Alternatively the center can be made permanent if new persons have to enter the system later. The individuals can verify that they have correct information from the center, can compute their own F-value, and later convince each D_i that this value is correct. This can be done simply by showing $w(ID(j))$ to D_i. In any case, no w-values have to remembered by the D_i's. This solution protects optimally against the D_i's reading data they should not have access to: each D_i can find data about individual j, precisely if j has given $F(ID(j))$ to D_i. For all other individuals, D_i is in exactly the same position as an outside enemy, by the independence condition.

Another way is to make the trapdoor for f known to all D_i's, but not to C. Then the D_i's can have their information stored in clear, and compute F-values as needed when they communicate with C. This removes the need for a trusted center, but on the other hand all D_i's are now faced with the security problem of safeguarding the trapdoor of f. Also the protection against the D_i's themselves is reduced: since knowledge of the trapdoor for f implies ability to compute F-values, the D_i's can check if a given individual is identical to a person registered in C, but they are not able to find the identity of a randomly chosen person in C, by the one way property of g.

At this point we must address the ultimate disaster for the proposed model: the disclosure of both trapdoors to an enemy. Obviously, the enemy may then calculate $ID(j)$ from $F(ID(j))$ and vice versa, and the entire database is seriously compromised. It therefore seems natural to introduce some messure that would make this impossible. One scheme is to apply a one-way funtion h to $ID(j)$ and then use the above model on $h(ID(j))$. If h is truely one-way this makes it impossible for anyone to get from $F(h(ID(j)))$ to $ID(j)$ except by exhaustive search which, by the very nature of the problem, we can never prevent if the trapdoors are revealed. There are many choices for practical implementations of h. It could be a hash function from a set of long ID's to a much smaller set of binary strings. Here one should take care to ensure injectivity on the set of actual ID's.

4. Concrete Constructions

1) $F(x) = (\sqrt{x} \bmod n)^3 \bmod n'$.
The function F can be constructed from

$$f(x) = x^2 \bmod n \quad \text{and} \quad g(x) = x^3 \bmod n',$$

where n and n' are products of two large and strong primes, chosen independently of each other. Moreover n and n' must be of compatible size (to prevent $F(x) = x$!). Also f in only injective on the elements of odd order in Z_n^*, which, as mentioned earlier is compensated for through the definition of ID.

Obviously, f and g do not commute and Theorem 3.2 indicates that F and F^{-1} are infeasible to compute for a non vanishing fraction of choices of n and n'. Note that if the factorization of n' is known, $\sqrt{x} \bmod n$ and hence probably x can be computed from $F(x)$. But as mentioned earlier, the trapdoor for g is never used in an application, so the factorization of n' can be deleted immediately after choosing n'.

Note that using squaring for both f and g will not work: given a consistent pair $(ID, F(ID))$, the witness can be computed using the Chinese Remainder Theorem and without knowledge of the factorizations! The generalization of this attack by Hastad [Ha] does not seem to work with our choice of exponents, since there is only 2 equations involving the witness, and this is insufficient to make the attack work. The number of equations needed to compute the witness becomes much larger, when the exponents get large, and therefore better security may be achieved by choosing random RSA-exponents in stead of 2 and 3.

2) $F(x) = \alpha^{\sqrt{x} \bmod n} \bmod n'$.
F can also be constructed from

$$f(x) = x^2 \bmod n \quad \text{and } g(x) = \alpha^x \bmod n'$$

where n is chosen as above. n' can be chosen as n or as a large prime, it is important that α is chosen such that it generates a large subgroup of $Z_{n'}^*$, whence discrete log's base α is (presumably) hard to compute. The same remarks as those relevant to case 1) applies here, except the fact that g is not trapdoor in this case. This means that Theorem 3.2 does not apply, on the other hand there is no risk of accidental release of a trapdoor for g.

For convenience, it might even be reasonable to choose $n = n'$, except for the fact that f and g will then not be independently chosen.

3) $F(x) = x^{\sqrt{x} \bmod n} \bmod n$.
Here, it is not so transparent how to choose f and g. However if we set

$$\hat{f}(x) = x^x \bmod n \quad \text{and} \quad g(x) = x^2 \bmod n$$

then

$$F(x) = x^{\sqrt{x} \bmod n} \bmod n = \sqrt{x}^{2(\sqrt{x} \bmod n)} \bmod n = g\hat{f}g^{-1}(x).$$

So F is conjugate to $\hat{f}$ under the action of the symmetric group on the elements of odd order in Z_n^* - on which g is a bijection.

The function $\hat{f}$ is not one to one. In fact it has some of the properties one would expect from a "typical" random function from Z_n^* to Z_n^*. Indeed, as is well known:

Lemma 4.1
Consider the set of functions from a set A into itself, where A has cardinality n. Then the average size of $Im(f)$ is

$$(1 - e^{-1})n \approx 0.63n \quad \square$$

From practical experiments, this seems to hold for $\hat{f}$. Consequently, it is reasonable to assume that $\hat{f}$ is one to one on very small subsets of its domain - like the set of existing ID's, for example. We then define

$$f = g\hat{f}^{-1}$$

to obtain $F(x) = gf^{-1}(x)$. In this setup, however, we cannot define $w(x) = f^{-1}(x)$ as in the previous section, since x would then not be computable from $w(x)$. In stead we simply define $w(x) = \sqrt{x} \bmod n$, from which both x and $F(x)$ can be easily computed, as required in the verifiability condition.

5. A Solution Using Bit Commitments

A *bit commitment scheme* is a method by which A can "encrypt" a bit in such a way that

(1) No one else can guess from the encryption which bit it encrypts.

(2) After releasing the encryption, A is *committed* to her choice of the bit, i.e. she can convince everyboby about her original choice - typically by releasing some more information - but she cannot change her mind about the choice.

The encryption is computed using a random input which is also chosen by A. For a bit string s, we will let $BC(s,r)$ denote a string of encryptions, one for each bit in s, computed using the binary string r as random input. We will talk about this as a *bit commitment* to s.

Such bit commitment schemes exist relative to many of the widely accepted intractability assumptions, such as the hardness of factoring, discrete log, graph isomorphism, etc. More details about bit commitments can be found in [Da] or [BrCr].

A very simple idea to solve our basic problem is now to let

$$F(ID(j)) = BC(ID(j),r), \text{ and put } \quad w(ID(j)) = r.$$

$F(ID(j))$ can be computed by j himself, and j can prove the correctness of $F(ID(j))$ to D_i, simply by showing $w(ID(j))$ to D_i.

By property 1 above, this solution satisfies both the anonymity condition and the independence condition, even in a strict information theoretic sense, if the bit commitment scheme is chosen correctly. Property 2 prevents cheating by individuals, such as having several identities represented by the same F-value. Unfortunately, there is still one problem left: the verifiability condition is not satisfied, because the witness is not a function of the identity, but is independently chosen, and therefore $ID(j)$ is not computable from $w(ID(j))$.

To see what this means in practice, consider the difference to the earlier described solutions: there, it is possible for j to prove that $ID(j)$ is NOT connected to $F(ID(j'))$ without having to reveal $F(ID(j))$, i.e. give up his own anonymity. This can be done by setting up a boolean circuit doing the following computation: it takes as input $w(ID(j))$, and is given $ID(j)$ and $F(ID(j'))$ as constants. It checks $w(ID(j))$ by computing $ID(j)$ from it, then computes $F(ID(j))$ and compares with $F(ID(j'))$. The output is two bits,

b_1, which is 1 precisely if the witness is correct, and

b_2, which is 1 precisely if $F(ID(j)) = F(ID(j'))$.

Using this circuit, j can convince anyone in minimum knowledge that he knows how to choose input for it that gives output $b_1 = 1$ and $b_2 = 0$. This is clearly equivalent to proving that he is not identical to the individual registered under

$F(ID(j'))$. The proof can be executed using for example the general protocol from [BrChCr].

With the solution from this section, the above protocol does not work, simply because it is not possible to check the correctness of a witness, and without this check, the protocol does not prove anything.

The only way to repair this is to ensure that j is committed, also to his choice of $w(ID(j))$. This can be done by introducing a public directory, containing entries for all individuals. For person j, the entry is $BC(w(ID(j)),r')$. This entry can be computed and proven correct by j himself initially. We can now make the above protocol work once again, since a witness can now be checked by testing whether the appropriate entry in the public file contains a commitment to the witness in question.

Thus this solution is of theoretical interest because it shows the existence of systems that provably satisfy the anonymity condition, but it is not of great practical importance, because we must introduce additional complications to get a complete solution.

Conclusion.

We have shown a practical solution to anonymous and verifiable registration in databases, and we have pointed out 3 basic conditions that such a solution should satisfy. We have also shown the existence of solutions that satisfy all 3 conditions.

References.

[BrChCr] G.Brassard, D.Chaum and C.Crepeau: "Minimum Disclosure Proofs of Knowledge", tech. report PM-R8710, CWI, Amsterdam 1987.

[BrCr] G.Brassard and C.Crepeau: "Non-Transitive Transfer of Confidence: a *perfect* zero-knowledge Protocol for SAT and beyond", Proc. of FOCS 86, pp.188-195.

[Ch] D.Chaum: "Security Without Identification: Transaction Systems to make Big Brother Obsolete", *CACM*, vol 28, 1985.

[Da] I. Damgård: "The Application of Clawfree Functions in Cryptography; Unconditional Protection in Cryptographic Protocols", Ph.D-thesis, Aarhus University, 1988.

[Ha] J.Hastad: "On Using RSA with Low Exponent in a Public Key Network", Proceedings of Crypto 85, Springer.

[BoLa] M.Boppana and L.Lagarias: "One Way Functions and Circuit Complexity", *Information and Computation*, vol 74, pp.226-240, 1987.