Implementing Group Signature Schemes With Smart Cards[*]

Abstract

Group signature schemes allow a group member to sign messages on behalf of the group. Such signatures must be anonymous and unlinkable but, whenever needed, a designated group manager can reveal the identity of the signer. During the last decade group signatures have been playing an important role in cryptographic research; many solutions have been proposed and some of them are quite efficient, with constant size of signatures and keys ([1], [6], [7] and [13]). However, some problems still remain among which the large number of computations during the signature protocol and the difficulty to achieve coalition-resistance and to deal with member revocation. In this paper we investigate the use of a tamper-resistant device (typically a smart card) to efficiently solve those problems.

1 Introduction

In 1991, D. Chaum and E. van Heijst [8] introduced the concept of group signature schemes. A group signature scheme allows members to sign a document on behalf of the group in such a way that signatures remain anonymous and unlinkable for everybody but a group manager (GM), who can recover the identity of the signer whenever needed (the latter procedure is called "signature opening"). Numerous group signature schemes have been published and some of them are quite efficient ([1], [6], [7] and [13]). In more recent ones, signatures and public keys are constant-size and security is well established, allowing them to be used in various applications such as electronic cash ([13]), voting or bidding systems ([11]). However some problems still remain among which the high computation cost of the signature, the coalition-resistance and member revocation (that we will consider more precisely in section 2.2).

In this paper, we investigate a completely different approach for carrying out group signature schemes, namely the usage of a tamper-resistant device - typically a smart card. This allows a very low cost during the signature phase. In fact, the signer only has to compute two or three modular exponentiations (in contrast with roughly a dozen in the scheme from [1] for example). Moreover, the coalition-resistance problem is very easy to solve when using smart cards and more simple procedures can be used for member revocation.

The use of a smart card allows to prevent an (untrusted) member from cheating, by letting his (trusted) device both secretly store the signature keys and control their legitimate usage. Using smart cards allows to provide solutions for member revocation

[*] USENIX Publisher, CARDIS 2002, pp. 1-10, 2002

that are generic (i.e. work with any group signature scheme) and efficient, in that the signatures are short and constant-size, and the number of computations (for the signer and the verifier) is constant. Moreover the work during the revocation protocol is constant. Since smart cards are more and more used in real-life applications, our solutions can be implemented at a negligible extra-cost.

This paper is organized as follows. The following section provides background on group signature schemes and points out remaining problems. Section 3 presents our group signature scheme and shows that it is coalition-resistant. Section 4 presents various solutions for providing member revocation. Finally, we conclude in section 5.

2 Group Signature Schemes

This section presents the state of the art in the group signature area. It briefly introduces the security properties and then the related works.

2.1 Definition

Definition 1. *A group signature scheme is a signature scheme which satisfies the following properties:*

(i) Correctness: a signature produced by a group member is always valid.

(ii) Unforgeability: only group members are able to sign messages on behalf of the group.

(iii) Anonymity: given a valid group signature, it is infeasible for everyone but the group manager to identify the actual signer.

(iv) Unlinkability: deciding whether two different valid signatures were computed by the same group member is infeasible.

(v) Exculpability: neither a group member nor the group manager can sign on behalf of other group members.

(vi) Traceability: the group manager is always able to open a valid signature, i.e. to identify the actual signer.

(vii) Coalition-Resistance: a colluding subset of group members should not be able to generate a valid signature that the group manager cannot link to one of the colluding group members.

2.2 Related Works: Group Signature Schemes

Since the paper of Camenisch and Stadler [7], the same method has always been used to set up a group signature scheme. It is based on a difficult problem implying two or more values. Alice is a member of the group if and only if she knows a solution of this difficult problem.

If Alice wants to become a group member, she interacts with GM (who holds a secret key) in order to obtain in a blind manner her private key and her membership certificate. This latter value allows GM to establish the link between a signature and a group member.

During the signature protocol, Alice encrypts her membership certificate, then “proves” that she knows a solution of the difficult problem and that she has correctly encrypted her certificate. As a consequence, this protocol involves numerous modular exponentiations. Someone who wants to verify the signature only has to verify the whole proof, also known as a signature of knowledge. The group manager can open the signature by decrypting Alice's certificate.

Coalition-resistance has often be defeated ([7]) and was an unsolved problem until [1] and [6]. In these two articles, the authors propose new group signature schemes based on the strong RSA assumption ([3] and [9]) and prove that they are resistant to coalitions.

2.3 Related Works: Member Revocation

At any time a member can decide to leave the group. In this case, we can reasonably think that he will not try to cheat in the future, but it is far from sure. Furthermore if a member is revoked from the group against his will, it is very plausible that he will try to keep on signing even if he has not the right to anymore. In both cases, it is necessary to set up a mechanism which prevents this type of fraud.

The paper of E. Bresson and J. Stern [4] proposed the most intuitive solution which consists for the signer in proving that he is different from any revoked member. But this method obviously generates a signature whose size linearly increases according to the number of revoked members.

In a recent paper, Song [12] proposed two revocation methods that are relatively similar and provide constant-length signatures and a constant work for the group manager. But the work of the verifier is also linear in the number of revoked members. Moreover, the solution is not very practical since it deals with a group with a limited life-expectancy.

Ateniese, Song and Tsudik [2] proposed a modification of the Ateniese et al. scheme [1] to improve member revocation, which also provides a constant size of signature. But works during the revocation phase and the verification one are linear in the number of revoked members. Finally, the cost of the signature is very expensive and consequently it is an overall unpractical solution.

Very recently, Camenisch and Lysyanskaya [5] proposed the first practical method for member revocation. It is also based on the scheme of Ateniese et al. [1] and therefore is not really generic (i.e. cannot be easily applied to any other group signature scheme). Moreover the signer has to make (possibly off-line) a number of modular exponentiations which is proportional to the number of modifications in the group (addition or deletion) until his last signature. Finally, this solution implies additional proofs of knowledge and, consequently, many other modular exponentiations.

3 Group Signature Schemes and Smart Cards

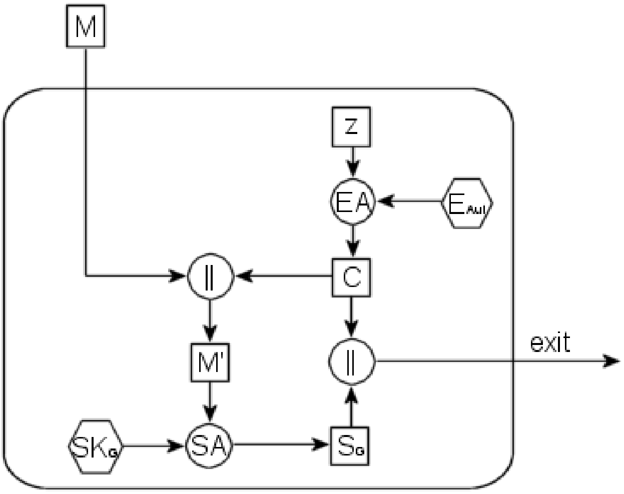
In this paper, we propose to build a group signature scheme relying on (typically) a smart card. It enables us to obtain straightforwardly the integrity of the (public or secret) data and of the program implemented in this tamper-resistant device. Moreover the confidentiality of keys and data is in the same way easily well-preserved. As a consequence, a solution simpler than previously proposed ones ([1] or [6]) can be introduced.

3.1 Shared Private Key and Smart Card

Our solution consists in using a smart card and a group-shared private key. First of all, we must choose an ordinary signature scheme (keys SK_G and PK_G) and a semantically secure cryptosystem (keys D_{Aut} and E_{Aut}). Then, the group manager computes keys in such a way that he can keep secret private ones (D_{Aut}) or distribute them (SK_G) to members without knowing them (for example, several group managers can share a discrete logarithm as the private key). He publishes public keys (PK_G and E_{Aut}).

If Alice wants to become a new group member, she firstly has to hold a smart card. Then, she has to obtain from the group manager an identifier z (which is unique and that identifies her) and the shared private key SK_G (which is common to all group members). Alice's smart card also has access to all parameters so as to use the cryptosystem (among which E_{Aut}) and the signature scheme defined above. The group manager has to keep in mind the link between the identifier (i.e. z) and the identity of the group member (i.e. Alice).

When Alice wants to sign a message as a group member (see Figure 1), she has to use her smart card. First, the identifier z is encrypted (algorithm EA) with the group manager's public key E_{Aut} (so that the group manager is the only one who can decrypt). Then the message M is concatenated with this encrypted value C and the whole is signed with the help of (algorithm SA and) the shared private key SK_G. As a consequence, only group members can sign a message and everybody is able to verify the signature with the associated public key PK_G.



M = Message
z = Member's identifier
EA = Encryption algorithm
E_{Aut} = GM's encryption key
C = Encryption of the identifier

$||$ = Concatenation algorithm
M' = Concatenation of M and C
SA = Signature algorithm
S_G = Signature of the message
SK_G = Group-shared signature private key

Figure 1: Shared Private Key and Smart Card

The verifier obtains the encrypted value C, the message M, and the signature S_G of the whole. He only has to verify the signature to be sure that the message is sent by a group member (because only group members possess the group-shared private key used to compute the signature). The group manager can open the signature by decrypting the identifier (with the key D_{Aut}).

This approach makes possible a very fast signature, since there is only one encryption and one ordinary signature to compute. It is important to note that the encryption scheme can either be symmetric or asymmetric. Nevertheless, it must be semantically secure. On the contrary, it is necessary to use an (asymmetric) signature scheme for obvious reasons.

3.2 Coalition-Resistance

The problem of coalition-resistance is easily solved when using tamper-resistant devices. In fact, it is impossible for two members to create a new card because they cannot access to protected data. In particular, they have no knowledge about the group-shared secret key SK_G (only their cards have).

3.3 Security Arguments

Theorem 1. *Under the assumption that a smart card is tamper-resistant, the group signature scheme proposed in section 3.1 is secure.*

Proof. (sketch of)
We have to show that our scheme satisfies all the security properties that are listed in Definition 1.
(i) Correctness: by construction.
(ii) Unforgeability: only group members can have the private group-shared key in their smart card (due to their interaction with the group manager) and consequently can sign on behalf of the group.
(iii) Anonymity: everybody has the same private signature key and the identifier of the signer is encrypted. As a consequence, a verifier cannot identify the signer because each group member can potentially compute the same signature and he cannot learn anything from the encrypted value (see semantically secure cryptosystem).
(iv) Unlinkability: group members have a shared key and the cryptosystem is semantically secure. It is then infeasible to link two different signatures.
(v) Exculpability: this is due to the fact that the identifier of a signer is embedded in his group signature and that the smart card is tamper-resistant.
(vi) Traceability: the card always encrypts the identifier of the group member. As a consequence, the group manager can always decrypt it and then open the signature.
(vii) Coalition-Resistance: see the remark in section 3.2. □

4 Revocation in Group Signature Schemes

We suggest two approaches for dealing with member revocation. The first one is based on a group-shared private key and, as in section 3, relies on the confidentiality of this key (even w.r.t. the card-holder). The second one is based on "black lists" and relies on the integrity of the "black list" membership program executed by the card.

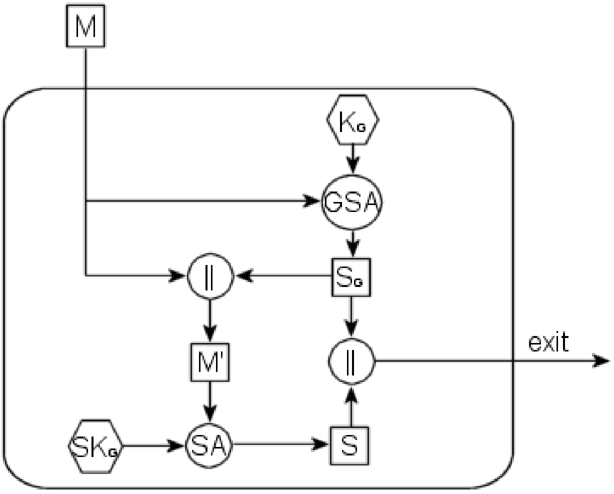
4.1 First Approach

4.1.1 General Principle

Our approach consists in generating an additional signature computed with a group-shared private key SK_G. We denote by PK_G the associated public key. SK_G is communicated by the group manager to each non revoked member, by the means of a group key distribution scheme (for example [14]). As a consequence, the revocation problem is reduced to a group key distribution problem, for which solutions already exist. Moreover, it happens that, in our case, these solutions are easier to use.

When a new member wants to integrate the group, the group manager securely sends him, among other elements, the group-shared key SK_G. And when a member is revoked, the group manager sets up a mechanism of member revocation, which implies the renewal of the group-shared key. It is impossible for the revoked member to learn anything about the new shared key and consequently he cannot sign anymore. The group manager has to publish data in order to make possible for other members to get the new key.

After that, if a member wants to sign on behalf of the group a message M (see Figure 2), he computes his group signature as usual (using [1], [6] or the solution described in section 3 for example) to obtain a couple (M, S_G) which he is going to sign by means of SK_G. The receiver can then verify the latter signature with PK_G and the value S_G as a signature of a group member.



M = Message
K_G = Group (private/secret) key(s)
GSA = Group signature algorithm
S_G = M's group signature
$\|$ = Concatenation algorithm
M' = Concatenation of M and S_G
SK_G = Group-shared signature private key
SA = Signature algorithm
S = Signature of the message

Figure 2: First Approach - Signature Protocol

4.1.2 Group Key Distribution

The most simple solution to manage group key distribution for our proposal is to share a secret key with each group member and to encrypt the new group-shared key with each secret key. Each valid member can decrypt one of the encrypted values to obtain the new group-shared key.
The identifier of the group member can be appended to each encrypted value. The group member only has to test if it is his own identifier and to decrypt the corresponding value if it is the case (see Figure 3).

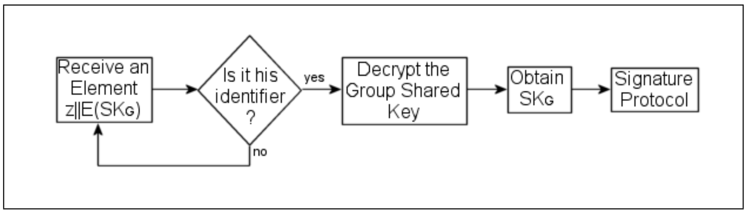


Figure 3: First Approach - Getting the Key

There exist some other solutions in the literature that are more interesting than this simple one. For example, Wong et al. [14] propose a solution based on a tree, where each leaf corresponds to a group member and where each node corresponds to a secret key. Each group member shares with the group manager all keys that are in the path between their leaf and the root. As every member knows the key root, this latter is chosen as the group-shared key. Consequently, for a particular revocation phase, the GM only has a limited number of values to encrypt, instead of many in the naive method.

4.1.3 Security and Efficiency Considerations

There is no way for the revoked member to learn anything about the new group-shared key. Then, the key contained in his smart card is no longer valid. As a consequence, the second signature will never be correct anymore. Finally, the group manager can efficiently and securely revoke group members.
The size of the signature is constant and the group signature is only increased by a single classical signature. Moreover, this method can be applied to any group signature scheme (including the one of section 3) and there is no extra work for the verifier (the cost is constant). The revocation protocol depends on the group key distribution scheme which is used. In particular, its cost will be at most linear in the number of group members.

4.1.4 Shared Private Key and Smart Card : Dynamic Case

Section 3 presents a new group signature scheme based on a shared secret key and a smart card. Section 4.1 presents a solution to the problem of revocation that adds to the general group signature an ordinary signature that depends on a group-shared key. If one wants to apply this revocation method to this group signature, each signer will have a priori to compute two different signatures. But the two signatures can easily be merged into a single one, since they both use a group-shared secret key. This leads to a very attractive method which allows revocation while generating only one signature. More precisely, only one signature is necessary because it is possible to replace the (fixed) group-shared key of section 3 with a dynamic group-shared key, as explained in section 4.1. The group-shared key used in the group signature scheme only needs to be modified by the group manager after each revocation (see. section 4.1.2) and the rest is unchanged. Figure 1 shows the mechanism carried out by the smart card during the signature phase to which must be added the key updating phase illustrated in Figure 3.

4.2 Second Approach

4.2.1 General Principle

Generally speaking, the simplest idea to deal with revocation problem is to maintain a revocation list (or a black list). The signer reveals a personal value and the verifier is then able to say, by matching the received value against each entry of the black list, if the person is revoked or not. Unfortunately, in the context of group signatures, it is not possible to reveal a personal value since it would compromise the anonymity of the signer. Using a smart card allows to give a simple solution to this problem. In a few words, each member owning a personal value (an identifier), the smart card will get the revocation list from the group manager database (or any database where the black list stands, e.g. the verifier device) and will check if one value of the list and its personal value match. If the card reaches the end of the list, it will accept to sign as a group member; and if its personal value lies in the list, then the card will refuse to sign and make itself out of order. Figure 4 shows the general principle of this approach.

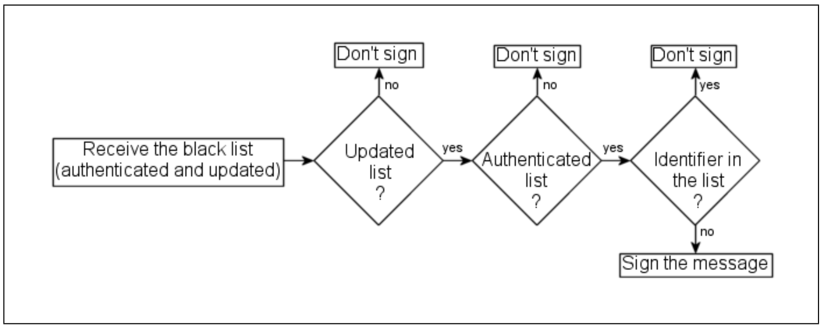


Figure 4: Second Approach - General Principle

4.2.2 First Solution

Description.

The first solution is straightforward and Figure 5 shows its principle. It consists in having the whole black list signed by the GM. Assuming that the underlying hash function of the signature scheme is iterative (most of them are so), it is possible for the smart card to verify the signature of a large message without needing to keep the entire message in his memory.

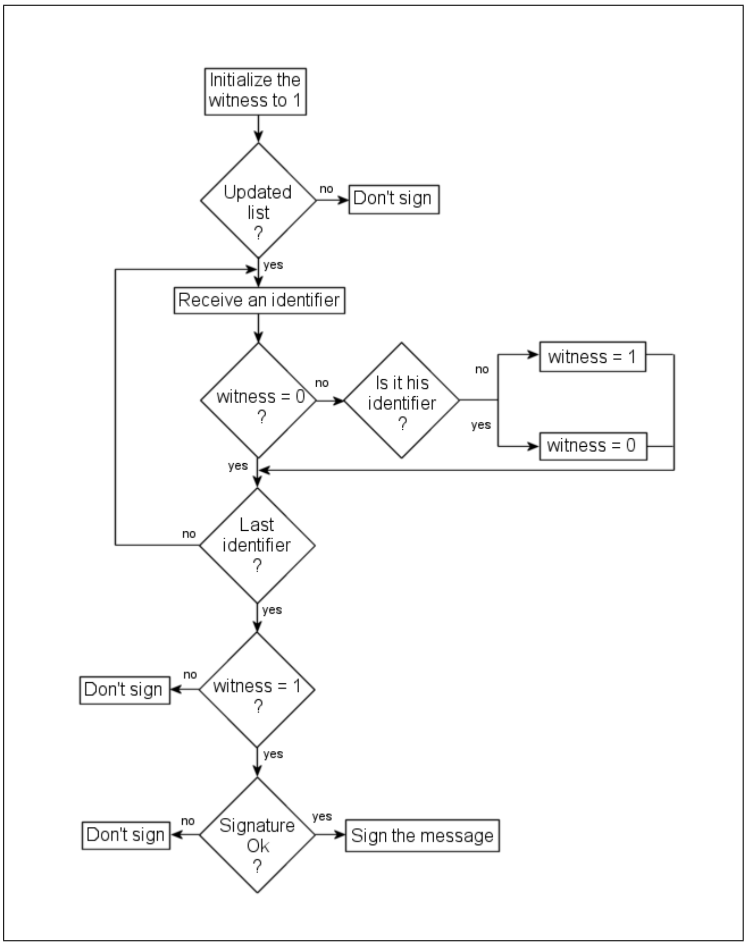


Figure 5: Second Approach - First Solution

Note that it is possible to use this method in a context of "white list" (that is a list which contains the identifiers of all members). In this case the card accepts to sign only if its identifier is in the list. It can be useful if the group has few members but a lot of revocations. We do not treat this case in this paper as it is an easy adaptation of the "black list" case.

Security.
The mechanism is secure under the assumption that the card is tamper-resistant. In fact, an attacker who wants to add some more values in the revocation list cannot do it because he cannot falsify the group manager signature. Then, it is impossible to substitute a value for another one because the signature would then be incorrect. Moreover removing a value from the revocation list would generate a card error because the final test on the signature verification would be wrong. Finally, replaying indefinitely the same revocation list would imply the rejection of the signature by the verifier because he could compare the date of the updating by GM (D_{GM}) with the

date of the last signature by the smart card (D_C). In fact, if D_C is different from D_{GM} he can think that the signer has wanted to cheat. For example the revocation list can be updated every day. Another solution is the use of an on-line verification (even if it is an "extreme" case). We can then conclude that the previous mechanism is secure under the assumption that the card is secure.

Efficiency Considerations.
This is a generic solution with a constant size of signature. In fact, the size of the signature is the same as that of the underlying signature scheme. From a computational point of view, there is a number of equality tests that is proportional to the number of revoked members, which can be considered as negligible, and the verification of only one signature. Another advantage of this solution is that the verifier does not have any extra computation to do. His work is no greater than that of the verifier in the underlying signature scheme. The work during the revocation phase is also constant. The group manager only has to add a value in the revocation list and to modify the resulting signature.

4.2.3 Second Solution

Description.
The second solution is also straightforward (see Figure 6). It consists in sending to the card all elements of the black list one by one, each of them signed by the group manager. It is yet necessary to add a *revocation number* (a sequence number: number 1 corresponds to the first revoked member, etc.) to prevent some attacks (for example addition or substitution of some identifiers). In addition, GM signs the date of his updating of the "black list" D_{GM} and the number of revoked members.

Security.
The mechanism is secure under the assumption that the card is tamper-resistant. In fact an attacker cannot add some more values in the revocation list because he cannot afterwards compute the related signature. He cannot substitute a value for another one because the corresponding signature would then be incorrect. Removing a value from the revocation list would generate a card error because the final test on the signed number of revoked members would be wrong. Finally, as for the first solution (see section 4.2.2), there is no way to replay indefinitely the same list.

Efficiency Considerations.
This is a generic solution with a constant size of signature. Once again, the size of the signature is the same as that of the underlying signature scheme. However, the signer has to check the validity of GM signatures for each revoked member which makes his work linear in the number of revoked members. The work of the group manager is constant-size since he only has to add a new value and to compute two signatures at each revocation. The verifier also has a constant-size work. Note that this method can also be used in a context of "white list".

An Improvement.
At first glance, this solution seems to be less attractive than the first one. Indeed, the number of signatures to be verified is large if there are many revoked members. But a

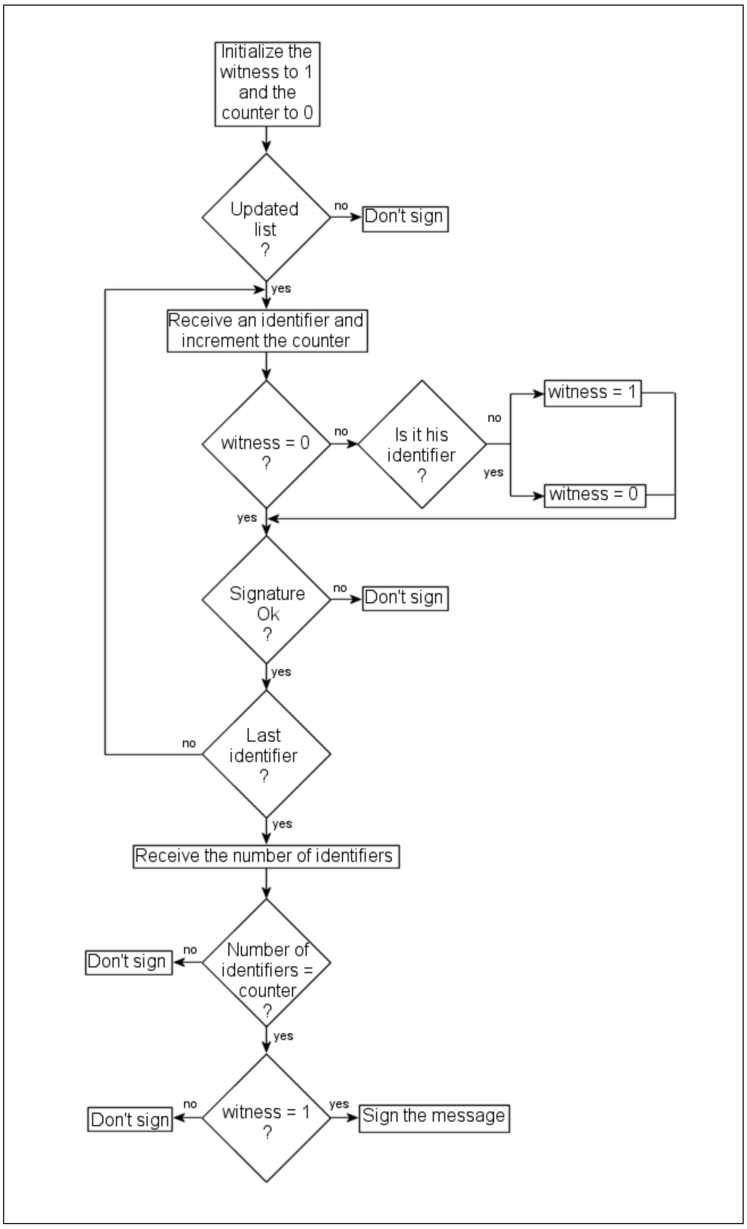


Figure 6: Second Approach - Second Solution

modification can be done so as to improve it.
Actually, we can argue that nobody can see nor modify the data exchanged between the smart card and the card reader. This is a plausible assumption if we consider that each member of the group has got a personal card reader that is always linked to his proper computer.
We therefore can improve the solution by putting on a new value in the smart card memory that corresponds to the number of values that the card has already verified in the group manager database. Indeed, the card does not need to test twice the same values. Consequently, it can inform the card reader of the number of values it has already tested and as a consequence the card reader will only send to the card the new values since the last signature of that card (plus the signature of the updating date and of the number of revoked members). As a result, the card will only have a limited number of GM signatures to verify before producing a signature.

4.2.4 Third Solution

A variant of the first solution consists in replacing the black list by a much shorter digest, so that the verification step becomes in average much faster. If the output of this step is "no", then we are sure that the member is not revoked and the card accepts to sign. However, if the output is "yes" then we cannot definitely conclude and the whole black list should be requested for a complete verification. We now briefly describe in the following subsection a possible way of achieving a compression of this kind.

An Example of Representation.
The mechanism named "Superimposed coding" [10] allows to store a set of data of variable size into a bit-string of fixed size. It is then possible, with a simple test, to estimate the probability that an element is in the set of data (which depends on the size of the result bit-string and on the number of data). This probability is equal to 0 if the output of the test is "no".
More precisely, the result is an m-bit string named B. We note $B = b_{m-1}b_{m-2}\dots b_1b_0$ where each $b_i \in \{0,1\}$. Initially, B is set to $00\dots0$. We have then k elements $y_1,\dots,y_k$ of various size and we note the set of data $Y = \{y_1,\dots,y_k\}$. Moreover, let us define q hash functions $h_1,\dots,h_q$ where each $h_i : \{0,1\}^* \longrightarrow \{0,1\}^c$ with $m = 2^c$.
For $j = 1..k$ we compute $h_1(y_j),\dots,h_q(y_j)$ and for every $l = 1..q$ we put to 1 the bit b_i where $i = h_l(y_j)$.
To know if the element y_R is in the set of data $Y = \{y_1,\dots,y_k\}$, we compute for every $l = 1..q$ $Y_l = h_l(y_R)$ and if there is an element $l_0 \in \{1,\dots,q\}$ such as $b_{Y_{l_0}} = 0$ then $y_R \notin Y$. If not, then $y_R \in Y$ with an error probability of about $\left(1 - e^{\frac{-kq}{m}}\right)^q$.

Description.
The group manager uses the "Superimposed coding" to transform the set of all personal keys of each revoked member into the m-bit string B. He then signs the latter value. A smart card is going to receive this signed bit-string, then treat it so as to verify the signature and to learn if its holder is revoked or not.
According to the size of the group and more particularly to the number of revoked members, the size of the result bit-string and the number of packets will vary in order to obtain good trade-offs (negligible error probability and m of reasonable size). For

example, for $q = 8$ and $k = 10000$ (i.e. at most 10000 revoked members), the error probability is 2.3×10^{-5} for a result bit-string of size 2^{18} (i.e. 32 Kbytes).

Efficiency Considerations.
This method is very interesting as the size of the signature and the number of computations remain constant and the resulting scheme is completely generic. Moreover, the size of verification work is constant. During the revocation protocol, computations are very simple and relatively independent from the number of revoked members, as the revocation manager only has to modify the resulting chain and to compute the new linked signature. The only drawback is the probability of mistake, but since it can be made negligible, this third solution seems to be the more attractive one.

5 Conclusion

We have introduced a new way of designing group signature schemes by using a tamper-resistant device (as a smart card). First we showed how to build a (coalition-resistant) group signature scheme starting from any (ordinary) signature scheme and any (semantically secure) encryption scheme. Such group signatures can be computed very efficiently (typically only one or two exponentiation(s)) and are constant-size. Then we addressed the member revocation problem and solved it by using two approaches: in the first one, the group signature is completed with a signature involving a group-shared key which is renewed at each revocation; in the second one, the card checks it does not lie in a "black list" before computing a group signature. As a result, smart cards allow to design group signature schemes which are simple, generic, efficient and secure at the same time.

References

[1] G. Ateniese, J. Camenisch, M. Joye, G. Tsudik. A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In L. Bellare, editor, Advances in Cryptology-CRYPTO'2000, volume 1880 of LNCS, pages 255-270. Springer-Verlag, 2000.

[2] G. Ateniese, D. Song and G. Tsudik. Quasi-Efficient Revocation of Group Signatures. In Financial Cryptography 2002, Southampton, Bermuda, March 11-14, 2002.

[3] N. Barić, B. Pfitzmann. Collision-Free Accumulators and Fail-Stop Signature Schemes Without Trees. In W. Fumy editor, Advances in Cryptology-EUROCRYPT'97, volume 1233 of LNCS, pages 480-484. Springer-Verlag, 1997.

[4] E. Bresson, J. Stern. Efficient Revocation in Group Signatures. In K. Kim, editor, Public Key Cryptography-PKC2001, volume 1992 of LNCS, pages 190-206. Springer-Verlag, 2001.

[5] J. Camenisch, A. Lysyanskaya. Efficient Revocation of Anonymous Group Membership Certificates and Anonymous Credentials. CRYPTO'2002, to appear.

[6] J. Camenisch, M. Michels. A Group Signature Scheme based on an RSA-variant. Technical Report RS-98-27, BRICS, Dept. of Comp. Sci., University of Arhus, preliminary version in Advances in Cryptology-EUROCRYPT'98, volume 1514 of LNCS.

[7] J. Camenisch, M. Stadler. Efficient Group Signature Schemes for Large Groups. In B. Kaliski, editor, Advances in Cryptology-CRYPTO'97, volume 1296 of LNCS, pages 410-424. Springer-Verlag, 1997.

[8] D. Chaum, E. van Heyst. Group Signatures. In D. W. Davies, editor, Advances in Cryptology-EUROCRYPT'91, volume 547 of LNCS, pages 257-265. Springer-Verlag, 1991.

[9] E. Fujisaki, T. Okamoto. Statistical Zero-Knowledge Protocols Solution to Identification and Signature Problems. In A.M. Odlyzko, editor, Advances in Cryptology-Crypto'97, volume 1294 of LNCS, pages 16-30. Springer-Verlag, 1997.

[10] D. E. Knuth. The Art of Computer Programming, Volume 3 / Sorting and Searching. Addisson-Wesley Publishing Compagny. pages 559-563. 1973.

[11] K.Q. Nguyen, J. Traoré. An Online Public Auction Protocol Protecting Bidder Privacy. Information Security and Privacy, 5th Australasian Conference-ACISP 2000, pages 427-442. Springer-Verlag, 2000.

[12] D. Song. Practical Forward Secure Group Signature Schemes. ACM on Computer and Communications Security. 2001.

[13] J. Traoré. Group Signatures and Their Relevance to Privacy-Protecting Off-Line Electronic Cash Systems. In J. Pieprzyk, R. Safavi-Naini, J. Seberry, editors, Information Security and Privacy, 4th Australasian Conference-ACISP'99, volume 1587 of LNCS, pages 228-243. Springer-Verlag, 1999.

[14] C. K. Wong, M. G. Gouda, S. S. Lam. Secure Group Communications Using Key Graph. Technical Report TR-97-23, July 28, 1997, revised version in IEEE/ACM Transactions on Networking, Feb 2000.