Off-line Group Signatures with Smart Cards

Jean-Bernard Fischer and Emmanuel Prouff

Oberthur Card Systems,
71-73 rue des Hautes Pâtures 92 726 Nanterre Cedex France.
{jb.fischer,e.prouff}@oberthurcs.com

Abstract. Group signatures allow a group member to sign messages anonymously on behalf of the group and, if needed, a group authority is able to identify the signer. In most applications, groups are dynamic and the number of arrivals and departures is non-negligible. Group signature schemes must take into account this situation and deal with member revocation. Even if group signature schemes have been intensively investigated during the last decade, few are applicable in low resource context. Among them, the simplest and most efficient has been proposed by Canard and Girault at Cardis 2002 and involves the smart card as security proxy. This solution has many advantages; however, there is a need to be connected to a group authority in order to sign or to verify the signature. Clearly, this is a drawback in embedded security for mobile applications. Based on a re-assessment of the notion of group signature, we propose an improvement of the Canard-Girault scheme allowing to perform signature and verification both off-line. In particular, we introduce the notion of *risk-management* for group signature schemes, which leads us to a novel approach of the member revocation problem.

Keywords: Group signature, smart card, dynamic group, revocation, risk management.

1 Introduction

In 1991, Chaum and Heijst [17] introduced the concept of group signatures. Unlike ordinary signatures, group signatures provide *anonymity* to the signer. Moreover, in exceptional cases, any group signature can be opened by a designated *revocation manager* to reveal indisputably the identity of the signer. In other words, group signatures also provide *traceability.*

Group signature is a very interesting cryptographic functionality, as it allows individuals to perform signatures on behalf of a group (typically a company). Thanks to anonymity, the individual is protected in the group. This is of particular interest when the need is to link the signature with a function in the company and not with a person in particular. In many applications, there is also a strong need that the signer can be held accountable for his actions inside the group, meaning that his anonymity can be uncovered if for example some fraudulent transactions are discovered. As mentioned in [8], a number of projects have emerged nowadays that require the properties of group signatures (see for instance [1, 2, 11]).

In [17], Chaum and Heijst focus on *static groups* in which the number of group members and their identities are fixed and frozen in the setup phase. In real life applications, groups must be considered being *dynamic* as people join and leave the group (*e.g.* the company) all the time. So, since the original paper [17], most schemes have been designed to take into account the dynamicity of the group structure [3, 4, 13–15, 18, 26]. Adding a member to an existing group is usually quite simple: it consists in allowing him to access the secrets shared by all the group members. The situation is more complex when a member leaves the group. Indeed, he does so with all the knowledge to continue performing signatures. Thus, the so-called `LEAVE` protocol provided by the group signature scheme must remove the ability to sign for the leaving member in such a way that the signing ability of all the remaining members is unaffected.

Usually, group signature schemes are derived from personal signature schemes. In these cases, the computational cost of both signature and verification is very high. Moreover, providing good *coalition-resistance* and easy member revocation is difficult and adds further complexity. The fact that members have access to a lot of information about the system explains the complexity of the schemes. Indeed, various tricks, such as proofs of knowledge, must be used to verify that signatures have been honestly performed.

If one were to remove all knowledge of the secret group parameters from the members, yet still provide them with a means to perform the signatures, the schemes could be vastly simplified. So, it seems natural to have a trusted third party do the actual computation as a proxy for both the members (ensuring anonymous signature) and the group (ensuring traceability). In Cardis 2002, Canard and Girault [16] investigated this approach by having a tamper-resistant device (typically a smart card) carry out the group signature on behalf of each member. In this setting, the smart card is considered as a honest entity that performs correct signatures even if its possessor is dishonest. This allows new schemes mimicking closely the classic personal signatures with a very low computational cost. Moreover, using smart cards, several standing problems like coalition-resistance become trivial to solve and simple procedures can be used for the `LEAVE` protocol.

To check that a signature has been produced by a non-revoked member of a group, Canard and Girault propose two solutions. They both have the same drawback: the group membership has to be checked on-line, either by the signer or by the verifier. Clearly, being on-line to perform a signature goes against the fundamental idea that the smart card is a tool to perform security operations in a non-connected situation.

Based on the ideas developed in [16], we propose a new scheme with the following improvement: the group parameters remain constant throughout the lifetime of the group, the smart cards can sign off-line and the verification of a signature can be done off-line. This solution is made possible through a trade-off: the smart card is allowed to become de-synchronized, and we resolve the membership issue by introducing the concept of *risk management* on the verifier's side.

This paper is organized as follows. In Sect. 2, we provide background on group signature schemes and we point out the remaining problems. In Sect. 3, we focus on group signature schemes using smart cards and we discuss the proposal of Canard and Girault. Based on this analysis, we present in Sect. 4 a new group signature scheme allowing both the signer and the verifier to perform all the operations off-line.

2 Group Signature Schemes in the Literature

2.1 Introduction to the Notion of Group Signature

Desmedt was the first to notice in [19] that conventional and public key systems (in the sense of Diffie and Hellman [20]) are not adapted when individuals must act on behalf of a group. In particular, he introduced the notion of *group oriented cryptography* and investigated the different kinds of groups and their cryptographic needs. A few years later, group signatures were introduced by Chaum and Heijst [17] to allow individual members to sign messages on behalf of a group. The identity of the signer is secret (anonymity) except that a *group authority* can identify the signer if needed (traceability). Since [17], more security requirements were added to group signatures. Nowadays, such schemes must satisfy the following security properties:

Correctness : signatures formed by a group member must be accepted by the verifier.

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

Anonymity [17]: given a valid signature of one or several message(s), identifying the actual signer is computationally hard for everyone but the revocation manager.

Unlinkability: deciding whether two valid signatures were computed by the same group member is computationally hard.

Exculpability [5]: neither a group member nor the group authority can sign on behalf of other group members.

Traceability [17]: the group authority is always able to open a valid signature and identify the signer.

Coalition-resistance [5]: a colluding subset of group members (even if comprised of the entire group) cannot generate a valid signature that the group authority cannot link to at least one of the colluding group members.

2.2 Group Signature for Static Groups

In the original paper of Chaum and Heijst [17], the structure of the group is assumed to be *static*. In their setting, the number of group members and their identities are fixed and frozen in the setup phase. A trusted entity chooses not only all the group parameters (including the keys) and an *opening key* for the group authority, but also chooses the signing key of each group member.

Such a scheme requires an uncomfortably and unrealistically high degree of trust in the party performing setup. Indeed, the latter can frame any group member, since he knows the signing keys of all members. A second major drawback concerns the group signature key. In Chaum and Heijst's schemes, its length is at least linear in the size of the group, and therefore the running time of the verification algorithm depends on the number of group members. Moreover, the length of the signature itself and the running time of the signing algorithm depend on the group size.

Improvements or generalizations of the schemes of Chaum and Heijst were later presented by Chen and Pedersen [18], Camenisch [12], Petersen [25] and more recently by Boneh and Shacham [8], but they did not overcome all the disadvantages of Chaum and Heijst's schemes for practical implementations.

2.3 Group Signature for Dynamic Groups

Clearly, static groups limit the applicability of group signatures, since they do not allow the addition of members to the group over time. The limitations of the schemes derived from [17] were in fact recognized early in the development of the area, and the practical literature has from the start focused on the case where the group is dynamic. In this setting, neither the number nor the identity of group members are fixed or known in the setup phase, which now consists of the group authority choosing only a group public key and an opening key (for traceability). An entity can join the group and obtain a private signing key at any time by engaging in an appropriate joining protocol with the group authority. Reciprocally, a group member can leave the group at any moment. In this case, he loses his ability to sign on behalf of the group. The following definitions are borrowed from Ateniese *et al.* [3]: a group signature scheme in a dynamic context is a digital signature scheme comprised of the following procedures.

`SETUP:` an algorithm for generating the initial group public key pk_G and the corresponding private key sk_G.

`JOIN:` a protocol between the group authority (GA) and an user that results in the user becoming a new group member.

`G-SIGN:` an algorithm whereby a group signature is computed by a group member given a group private key and a message.

`G-VERIFY:` an algorithm for establishing the validity of a group signature given a group public key and a signed message.

`OPEN:` an algorithm that, given a signed message and a group private key, allows the revocation manager to determine the identity of the signer.

`LEAVE :` a protocol between the group authority (GA) and the members of the group that results in the user who leaves the group being no more able to sign as a group member.

Until 1997, dynamic signature schemes were designed as processes involving static group signature schemes together with dynamic group management.

Indeed, in these schemes, the group is re-created each time a member joins or leaves the group. The signature algorithm itself works as if it were intended for a static group. From this point of view, we can consider that the signature is not dedicated to a dynamic context.

Camenisch and Stadler [15] were the first to propose a signature scheme where the lengthes of the public keys and of the signatures are independent of the number of group members, and can therefore be used for large and dynamic groups. Furthermore, Camenisch and Stadler propose a `JOIN` procedure allowing the group authority to add new members without modification of the group public key. As a result, the identification of the group by its signature public key is now possible. Nowadays, the security of schemes derived from Camenisch and Stadler's solution has become well-established [6, 7], allowing them to be used in various applications such as electronic cash [26] or voting [24]. Moreover, the practical deployment of these schemes is investigated in [22] and different ways of co-operatively forming signatures are proposed. But despite all these efforts, problems still remain, among which the difficulty to achieve coalition-resistance and to deal with member revocation. For these reasons, the properties of these schemes do not match the properties of group signatures given by Ateniese *et al.*

2.4 The Problem of Member Revocation in Dynamic Groups

Bresson and Stern in [10] and Bresson in [9] proposed some ways of modifying the schemes of Camenisch and Stadler [15], Camenisch and Michels [14] and Ateniese *et al.* [3] to obtain a signature scheme with a `LEAVE` protocol. Nowadays, the proposal of Camenisch and Lysyanskaya [13] is preferred to transform the signature schemes [15] and [3] into group signatures schemes (with `LEAVE` protocols). However, the resulting solutions are still very complex and costly in memory space and computation time. Indeed, each member of the group has access to a part of the group-signature parameters and thus is able to use them for cheating. Numerous asymmetric algorithms (involving costly arithmetic computations) are used to counteract such internal attacks.

The use of a tamper proof device such as the smart card makes it easy to prevent a member from cheating, by letting his trusted device both store secretly the signature keys and control their legitimate usage.

3 Group Signature Schemes for Smart Cards

3.1 Smart Card Approach: a Step by Step Construction

General Idea The basic idea is to use the smart card as a deputy for signing, since the card is reputed honest. This means that the card will not perform operations out of the context it has been designed for: it will not allow the modification of the secrets, their disclosure or the performance of a computation with wrong data or execution errors. If a computation has been performed by a smart card, then the result can be considered as exactly what is expected: the honest behaviour of the signer (which is now the smart card) is not to be doubted, and no other proof of correctness is needed. As recalled in this section, this approach enables a simple and generic solution with very straightforward protocols.

Anonymity From the algorithmic point of view, we would start from an ordinary signature scheme (RSA or DSA for instance), with private key sk_G and with public key pk_G. The parameter sk_G is stored in a non-volatile memory inside the smart cards of the group members. In that way, any smart card can sign for the group, and a message is signed by two different smart cards in exactly the same way. This is a perfect anonymous signature. In order to ensure that the group authority cannot sign for a group member (and thus to satisfy the unforgeability property), we can distribute sk_G to the smart cards in such a way that the group authority is not able to use it for signing. A very simple way is to have a smart card play the role of the group authority; thus, secrets are perfectly safe and the group authority cannot sign for others since such a functionality is simply not implemented in this card.

Traceability Being totally anonymous, the scheme above does not satisfy yet all the properties of a group signature which have been recalled in Sect. 2.1. To allow traceability, the identity of the signer must be added to the message which is signed. Furthermore, this has to be done such that the anonymity of the signer is still enforced for everyone but the group authority: said identity should be enciphered in such a way that only the group authority can decipher it. Since smart cards are trusted to protect the secrets, the ciphering algorithm can be symmetric. In such a case, all the smart cards of the group members share a symmetric key K_G with the group authority's smart card, which is the only one having the ability to decipher a ciphered identity. To satisfy the unlinkability property, the encryption of the signer identity must be *probabilistic* (in our particular case, we can for instance design such an algorithm by modifying a deterministic encryption algorithm as described in [23], page 306).

The Scheme If Alice wants to sign as a member of the group, she has to hold a smart card containing her unique identifier, the group signature key sk_G and the parameter K_G to encipher her identity, along with the group signature algorithm. These are provided during the `JOIN` procedure by the group authority, either as a specific dedicated smart card, or as an application running on a multi-applicative smart card.

Let $\texttt{ENC}(x, K_G)$ denote the probabilistic encryption of a message x with a key K_G and let `DEC` denote the corresponding decryption algorithm. Let $\texttt{SIGN}(x, sk_G)$ denote the signature (RSA or DSA for instance) of a digest of x with an asymmetric private key sk_G and let `VERIFY` be the corresponding algorithm for establishing the validity of the signature given a group public key pk_G, a message m

and its signature s. We resume in the following the `G-SIGN` procedure performed by Alice's smart card:

Procedure 3.1 `G-SIGN` procedure for smart cards

INPUTS: a message m, the signer identity Id_A, the pair of keys (sk_G, K_G)
OUTPUT: a group signature of m

```
1. $c \longleftarrow \texttt{Enc}(Id_A, K_G)$
2. $s \longleftarrow \texttt{SIGN}(m||c, sk_G)$
3. Output $(m, c, s)$
```

The `G-VERIFY` algorithm associated with Procedure 3.1 is simple. The verifier obtains the message m, the cipher c and the signature s. Then, he simply verifies with the group public key pk_G that the signature corresponds to the concatenation of m and c. If needed, the group authority can apply the `OPEN` procedure which consists in decrypting c to recover the identity Id_A of the signer.

Security Analysis The algorithm depicted in Procedure 3.1 is very efficient, since a group signature is performed by computing only one encryption and one ordinary signature. Furthermore, if the involved algorithms `ENC` and `SIGN` are assumed to be secure, then the properties of a group signature are all provided thanks to the tamper-resistance of the smart card:

– Unforgeability: only group members have sk_G in their smart card and sk_G is never made public to anyone (including the smart card holder), so that no one outside of the group can sign with this key.
– Anonymity: the signature key is the same for all the group members and the identity is ciphered.
– Unlinkability: the cryptosystem is semantically secure.
– Exculpability: the identity of the signer is unique and is added by the smart card, which prevents any fraud.
– Traceability: the card always encrypts the identity of the member in a way that can be recovered by the group authority.
– Coalition-resistance: smart cards being tamper-resistant, group members cannot access to any information about the group keys sk_G and K_G (in that setting, coalition-resistance is equivalent to unforgeability).

The group authority can add new members very easily by loading onto their smart cards a unique identifier Id together with the pair (sk_G, K_G). However, it is not simple to tackle the issue of member revocation: if a member leaves the group with his smart card still empowered with the group signature's service, how can the group authority prevent him from signing for the group?

3.2 Group Signature Schemes of Canard and Girault

In [16], Canard and Girault propose a group signature scheme along the lines of what has been described in Sect. 3.1. To provide a revocation process, they suggest two different approaches, either based on an additional group-shared signature key, or based on a *black listing* scheme. We analyze in the following the two approaches and we show that they are impractical in a typical smart-card setting, where transactions are performed off-line and where the participants have no access to central servers.

First Approach It simply consists in generating a second signature on the concatenation of the message and the group signature.

Procedure 3.2 `G-SIGN` procedure for smart cards

```
INPUTS: a message $m$, the signer identity $Id_A$, a triplet of keys $(sk_G, K_G, dsk_G)$
OUTPUT: a group signature of $m$

1. $(m, c, s') \longleftarrow$ Procedure 3.1$(m, Id_A, sk_G, K_G)$
2. $s \longleftarrow$ SIGN$(m||s', dsk_G)$
3. Output $(m, c, s, s')$
```

The signature computed with dsk_G at the second step of Procedure 3.2 is dynamic: whenever a member leaves the group, all the smart cards of the remaining members are provided with a new key dsk_G. Since the revoked member does not know this new key, he cannot continue to sign as a group member. Let us denote by dpk_G the public key corresponding with dsk_G. We recall in the following the verification procedure associated with Procedure 3.2:

Procedure 3.3 `G-VERIFY` procedure for smart cards

```
INPUTS: an uplet $(m, c, s, s')$, a group identity $Id_G$
OUTPUT: ACCEPT or REJECT

1. $(dpk_G, pk_G) \longleftarrow$ Ask for the pair of public keys associated with $Id_G$
2. Check1 $\longleftarrow$ VERIFY$(m||c, s', pk_G)$
3. Check2 $\longleftarrow$ VERIFY$(m||s, dpk_G)$
4. if Check1 = ACCEPT and Check2 = ACCEPT
      Output ACCEPT
5. else output REJECT
```

A group signature scheme involving Procedures 3.2 and 3.3 has several drawbacks.

Firstly, each time a verifier checks a signature, he has to access a list in order to retrieve the corresponding key. This imposes on the verifier to be on-line in order to access such a list. Furthermore, each key in the list has to be associated with a validity period, from the time of distribution to the time of revocation, so that the verifier can decide which key corresponds to the time of the signature. This also implies that the verifier must have a way to know when the signature has been performed.

Secondly, the distribution of the dynamic group keys to the smart cards is problematic. The whole process is cumbersome, since the authors sketch a scheme based on several group authorities sharing a discrete logarithm as a private key. Synchronizing all the smart cards is also going to be difficult: being in essence a portable device, a smart card is more likely to be in the pocket of the member than connected to the group authority. Thus, a member may issue invalid signatures in good faith, simply because the group authority has issued a new key and his smart card is de-synchronized.

Finally, this approach works only if all the smart cards are on-line permanently, or at least have a way to synchronize the group key with the group authority before actually performing the signature. It becomes quickly inefficient when the number of group members is large (distribution) and when revocations occur often (synchronization), as for a thousand employees strong company.

Second Approach It consists in using the so-called *revocation lists*. It is the most convenient of the two solutions proposed by Canard and Girault, since it has the advantage of involving only a static pair of group keys (sk_G, pk_G). The group authority maintains a list of revoked members and, before every signature, the smart card simply looks up the list in order to confirm its own belonging to the group. If the member identity is not on the list, it signs with the fixed group key as described in Procedure 3.1. Otherwise, the member is revoked: its smart card cancels its group belonging and will not sign for this group anymore.

The obvious drawback of this approach is that the smart card has to be on-line in order to sign, since verification of the revocation list is a prerequisite for signature. This goes against the fundamental idea of the smart card being a mobile device that permits operations off-line.

4 Our Proposal

With a personal signature, time is mostly irrelevant. Indeed, a signature, either handwritten or electronic, is proof of a commitment by a physical person; it does not change over time and it is similarly binding whether the signature has been performed yesterday or several years ago (ignoring issues of key ageing or obsolescence). As we argue in the following section, time plays a central role in group signatures, where the membership question must be resolved.

4.1 Group Signature: a Question of Timing

A group is an abstract notion, an intellectual construction which partitions entities into two sets, those belonging to the group and those who do not. When it comes to partitioning people, a straightforward and usual way is to give a differentiator to the group members, setting them apart from the others. In the case of group signatures, the group authority gives to the members the means to sign on behalf of the group. In that sense the group can be defined as those who can sign. Thus, the fact that someone can produce a valid signature is often considered in itself as a proof of membership. But, however appealing, this approach precludes from performing the signature off-line. Indeed, the group authority has to be involved to resolve the question of membership.

Let us now re-discuss the membership notion. Clearly, in a dynamic group, membership in essence is a question of time: the period of time during which someone is a member of a group is clearly defined. Consequently, there are two critical moments in the relationship between a physical person and an abstract group: joining the group and leaving the group. So, if the verifier wants to link the signature to the group, two things have to be verified: firstly that the signature is valid, and secondly that the signer was member of the group at the time of signature. When designing a group signature scheme, the main difficulty is to resolve the membership question, and the following two threats must be considered:

Threat 1: a member can sign a document ante-dated to a time before him joining the group.

Threat 2: a member can sign a document post-dated to a time after he will have left the group. In particular, the revocation issue, when a member leaves the group with the means to sign, can be considered thus.

In any case, the validity of the group signature is in jeopardy if the verifier cannot determine whether the signature has been issued by a member at a time when he was part of the group. Clearly, any group signature should provide an easy way to check that the signer belonged to the group at the instant of signature.

The classical approach to solve the revocation issue is to rebuild the group after each departure, distributing new means to sign to the remaining members. This restores the basic partition between those in the know and the others (including now the leaving member). It amounts to rebuilding a static group each time (as it is done by Canard and Girault in their first approach described in 3.2). This mandates an evidence included in the signature that is clearly visible to the verifier and is not forgeable by a former member of the group. It is usually done by changing the public key of the group each time a member leaves the group. Meaning that, in order for the verifier to know which key has been used at the time of signature, he has to get it from the group authority in an authenticated manner; and the deciding parameter is the time of the signature, which solves postdating. Ante-dating is usually not considered, as it would imply changing the group key also at each joining.

Our idea consists in incorporating in the signature all the relevant elements of date, in such a way that the verifier can easily make an informed decision on the fact that the member belongs to the group at the moment of signing. In that way, we are able to prevent Threats 1 and 2. Implementing this idea in smart cards, we show in the following that we are also able to perform operation with the signer and the verifier being off-line.

4.2 Off-line Group Signatures and Risk Management

Let us assume that, from time to time, Alice's smart card verifies on-line with the group authority that it belongs to the group. Let us denote by Md_A the *Membership date*, which corresponds to the date of the last successful membership test with the group authority. We present our new group signature algorithm in the following:

Procedure 4.1 New `G-SIGN` procedure for smart card

INPUTS: a message m, the signer's identity Id_A, the pair of keys (sk_G, K_G), the signing date d and the membership date Md_A
OUTPUT: a group signature of m

```
1. $c \longrightarrow$ ENC($Id_A, K_G$)
2. $s \longrightarrow$ SIGN($m||d||Md_A||c, sk_G$)
3. Output $(m||d||Md_A||c, Id_G, s)$
```

When the verifier gets a message signed by Alice, he first checks the correctness of the classical signature. Then, he follows what we called a *risk management procedure*. It consists in comparing the two dates d and Md_A, so that the verifier can decide whether the time interval between the last connection of the card to the group authority and the date of the actual signature is acceptable in view of its security policy. If the verifier deems the dates d and Md_A to be too far apart for him to confidently accept the signature, then the off-line part of the verification procedure rejects the signature. When this occurs, the verifier has two options: to stop the procedure and reject the signature, or to start the on-line part of the verification procedure by connecting to the group authority.

We resume all these operations in the following `G-VERIFY` procedure, associated with Procedure 4.1:

Procedure 4.2 New `G-VERIFY` procedure for smart card

INPUTS: an uplet $(m||d||Md_A||c, s)$ and a group identifier Id_G
OUTPUT: `ACCEPT` or `REJECT`

```
1. if VERIFY($m||d||Md_A||c, s, pk_G$) = REJECT
     Output REJECT
2. if RISK-MANAGEMENT($m, d, Md_A, Id_G$) = REJECT
     if ONLINE-CONFIRM ($m||d||Md_A||c, s, Id_G$) = REJECT
         Output REJECT
3. Output ACCEPT
```

We give in the following an instance of `RISK-MANAGEMENT` procedure:

Procedure 4.3 `RISK-MANAGEMENT` procedure

INPUTS: an uplet (m, d, Md_A) and a group identifier Id_G

Output: `ACCEPT` or `REJECT`

```
1. Compute time difference between $d$ and $Md_A$
2. Make decision according to security policy for the group $Id_G$ and/or ask human verifier
3. Output ACCEPT or REJECT
```

In order to check on-line the belonging of the signer to the group, the verifier has to send the signature information to the group authority. As an example, we used the following protocol in our implementation:

Procedure 4.4 `ONLINE-CONFIRM` procedure

Inputs: a pair $(m||d||Md_A||c, s)$ and a group identifier Id_G
Output: `ACCEPT` or `REJECT`

```
1. Connect in authenticated manner to group authority (GA) of group $Id_G$
2. Send $(m||d||Md_A||c, s)$ to GA
3. GA verifies signature and computes $Id_A = \texttt{DEC}(c, K_G)$
4. if $Id_A$ belonged to the group $Id_G$ at the date $d$
      Output ACCEPT
5. else output REJECT
```

Let us recapitulate the different procedures for our signature protocol.

`Setup:` the group authority is in possession of a GA smart card. He chooses the group parameter Id_G and provides it to the GA smart card. The smart card generates the signature key set (sk_G, pk_G) and the symmetric key K_G and outputs pk_G. The group authority publishes pk_G for all the verifiers to know.

`Join:` the new member A is issued a smart card personalised with a unique identity Id_A. The smart card contacts the group authority and is issued the keys sk_G and K_G in a secure and confidential manner from smart card to smart card. Then, the member card makes a first connection to the group authority to update (actually initialise) the membership date.

`G-Sign:` see Procedure 4.1

`G-Verify:` see Procedure 4.2

`Open:` the group authority deciphers c and returns the signer identity.

`Leave:` if the card connects to the group authority after the user has left the group, it will be issued a command to stop signing on behalf of that group. From there on, the former member will not be able to sign anymore with his card. Otherwise, if the card does never connect again to the group authority, then the value Md_G shall never be updated and the signature will quickly loose any credibility.

Remark 1. Whatever the signature scheme, a former member can always sign a document ante-dated to a time when he was part of the group, there is no way to prevent this. However, in our protocol, the window of time in which he can ante-date a signature is restricted to the time between his last membership update and the time of his revocation.

4.3 Discussion About the New Proposal

The protocol described in the previous section allows both the signer and the verifier to interact off-line. This is a crucial advantage for the smart card, as it is in essence mobile. Without this feature, the communication between the verifier and the group authority is a major obstacle which completely prevents the completion of the signature process.

This paper does not aim to discuss the details of the risk management procedure. It is up to the verifier to define it in compliance with its security policy. It can for example take into account the security importance of the document, the level of confidence in the signer, the endorsement leeway of the verifier, etc.

In the on-line mode, our protocol is similar to the second approach of Canard and Girault (depicted in Sect. 3.2). The two solutions essentially differ with respect to the entity (signer or verifier) which checks the membership on-line with the group authority. As for hand-written signatures, our protocol has been defined such that the group-membership test is the verifier's responsibility. By involving a risk management process, we give him the means to decide wether or not such an on-line test is necessary.

5 Conclusion

In this paper, we recalled why the use of a smart card is natural when we want to design an efficient signature scheme for dynamic groups. After having pointed out the difficulty of dealing with member revocation, we designed a new group signature improving the proposals of Canard and Girault. Our scheme allows to perform signature and verification both off-line, which was not possible with the previous solutions. We succeeded in resolving the membership issue by introducing the concept of risk management on the verifier's side. In this manner, the group signature we propose is very close to a hand-written signature: it is always possible to sign on behalf of the group if one has the means for doing it, and the verifier checks both the signature and some additional information on the signer in order to quantify the risk of a falsified signature.

References

1. IEEE P1556 working group, VSC project. Dedicated short range communications (DSRC), 2003.
2. Trusted computing group. Trusted Computing Platform Alliance (TCPA) Main Specification, 2003.
3. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 255–270. Springer-Verlag, 2000.
4. G. Ateniese, D. X. Song, and G. Tsudik. Quasi-efficient revocation in group signatures. In M. Blaze, editor, *Financial Cryptography 2002*, volume 2357 of *Lecture Notes in Computer Science*, pages 183–197. Springer, 2002.

5. G. Ateniese and G. Tsudik. Some open issues and new directions in group signatures. In Kalisky Jr. [21], pages 196–211.
6. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: formal definition, simplified requirements and a construction based on trapdoor permutations. In E. Biham, editor, *Advances in cryptology - EUROCRYPT 2003, proceedings of the internarional conference on the theory and application of cryptographic techniques*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629, Warsaw, Poland, May 2003. Springer-Verlag.
7. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. Cryptology ePrint Archive, Report 2004/077, 2004. `http://eprint.iacr.org/`.
8. D. Boneh and H. Shacham. Group signatures with verifier-local revocation. Manuscrit, 2004.
9. E. Bresson. *Protocoles cryptographiques pour l'authentification et l'anonymat dans les groupes.* PhD thesis, École polytechnique, Oct. 2002.
10. E. Bresson and J. Stern. Efficient revocation in group signatures. In K. Kim, editor, *Public Key Cryptography – PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 190–206. Springer-Verlag, 2001.
11. E. Brickell. An efficient protocol for anonymousy providing assurance of the container of a private key. Submitted to the Trusted Computing Group, april 2003.
12. J. Camenisch. Efficient and generalized group signatures. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 465–479. Springer-Verlag, 1997.
13. J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In M. Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer-Verlag, 2002.
14. J. Camenisch and M. Michels. A group signature scheme with improved efficiency. In K. Ohta and D. Pei, editors, *ASIACRYPT*, volume 1514 of *Lecture Notes in Computer Science*, pages 160–174. Springer, 1998.
15. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups. In Kalisky Jr. [21], pages 410–424.
16. S. Canard and M. Girault. Implementing group signature schemes with smart cards. In *CARDIS 2002*, pages 1–10. USENIX, 2002.
17. D. Chaum and E. van Heyst. Group signatures. In J. Feigenbaum, editor, *Advances in Cryptology – EUROCRYPT '91*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer-Verlag, 1991.
18. L. Chen and T. Pedersen. New group signatures schemes. In A. D. Santis, editor, *Advances in Cryptology – EUROCRYPT '94*, volume 950 of *Lecture Notes in Computer Science*, pages 171–181. Springer-Verlag, 1994.
19. Y. Desmedt. Society and group oriented cryptography: A new concept. In C. Pomerance, editor, *CRYPTO'87*, volume 293 of *Lecture Notes in Computer Science*, pages 120–127. Springer, 1987.
20. W. Diffie and M. Hellman. New Directions in Cryptography. *IEEE Transaction on Information Theory*, 22(6):644–654, november 1976.
21. B. Kalisky Jr., editor. *Advances in Cryptology – CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*. Springer-Verlag, 1997.
22. G. Maitland and C. Boyd. Co-operatively formed group signatures. In B. Preneel, editor, *CT-RSA*, volume 2271 of *Lecture Notes in Computer Science*, pages 218–235. Springer-Verlag, 2002.

23. A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1997. Electronic version available at `http://www.cacr.math.uwaterloo.ca/hac/`.
24. K. Q. Nguyen and J. Traoré. An online public auction protocol protecting bidder privacy. In E. Dawson, A. Clark, and C. Boyd, editors, *ACISP*, volume 1841 of *Lecture Notes in Computer Science*, pages 427–442. Springer, 2000.
25. H. Petersen. How to convert any digital signature scheme into a group signature. In *Security Protocols Workshop*, 1997.
26. J. Traoré. Group signatures and their relevance to privacy-protecting off-line electronic cash systems. In J. Pieprzyk, R. Safavi-Naini, and J. Seberry, editors, *ACISP'99*, volume 1587 of *Lecture Notes in Computer Science*, pages 228–243. Springer, 1999.