One Time Anonymous Certificate: X.509 Supporting Anonymity*

Aymen Abed[1] and Sébastien Canard[2]

[1] Logica IT Service - 17 place des Reflets - 92097 Paris la Défense Cedex - France
[2] Orange Labs - 42 rue des Coutures - BP6234 - F-14066 Caen Cedex - France

Abstract. It is widely admitted that group signatures are today one of the most important cryptographic tool regarding privacy enhancing technologies. As evidence, the ISO organization has began a subject on authentication mechanisms supporting anonymity, in which group signatures are largely studied. However, it seems difficult to embed group signatures into other standards designed for classical authentication and signature mechanisms, such as the PKI X.509 certification. In fact, X.509 public key certificates are today widely used but not designed to support anonymity. One attempt has been done by Benjumea *et al.* but with the drawback that (i) the solution loses the principle of one certification per signer, (ii) revocation cannot be performed efficiently and (iii) the proposed architecture can not be applied to anonymous credentials, a concept close to group signature and today implemented by IBM or Microsoft. This paper presents a new approach which permits to use the X.509 standard to group signature schemes and anonymous credentials in a more standard and efficient way than related work.

1 Introduction

Anonymity is today considered as one possible base of individual privacy. In this case, the customer is anonymous when she is accessing a specific service. A lot of theoretical work has been done to design new signature schemes which provide such anonymity of the signer. Among them, one of the most popular is the concept of group signature scheme, introduced by Chaum and van Heyst [17] and currently under discussion to be standardized at the ISO organization.

Several constructions of group signature schemes today exist, either secure in the random oracle model [1, 5, 18] or in the standard model [21]. Many variants have also been proposed, such as Direct Anonymous Attestation [16, 7, 15, 30] or anonymous credential systems [12, 2].

The latter, a.k.a. "need to know approach", is an emerging concept which is currently under development in concrete systems such as IBM Idemix [23] or

* This work has been financially supported by the French Agence Nationale de la Recherche under the PACE project and by the European Commission's Seventh Framework Programme (FP7) under contract number ICT-2007-216676 ECRYPT II, while 1st author was working at Orange Labs.

Appeared in S.-H. Heng, R. N. Wright, B.-M. Goi (Eds.): CANS 2010, volume 6467 of LNCS, pp. 334-353, 2010.

Microsoft Credentica UProve [25]. This concept permits users to access services that are conditioned to some attributes such as the age, the address or the nationality, while revealing the minimum of information about them. In fact, this is not necessary to reveal the information on all the attributes (age, nationality, etc.) or indeed the attribute itself. For example, one needs not to reveal her date of birth to prove that she is more than 65 years old.

However, there is not enough work to apply these theoretical concepts in current existing infrastructures. As one of the main example, X.509 public key certificates are widely used but not designed to support anonymity. A Public Key Infrastructure (PKI) serves to identify the holder of a private key in a standard fashion and has been used in many types of transactions and communications over Internet. It seems however that it is not possible to use a X.509 certificate with e.g. group signatures, for the following reasons.

1. In the *Subject Name* field of a X.509 certificate, one can find the true identity of the user to whom it was issued.
2. In the *Subject Public Key Information* field, the Certification Authority (CA) puts the public key of the signer.
3. A certificate is published by the CA, for example in a directory system, which may be widely accessible.

Recently at CANS 2007, Benjumea, Choi, Lopez and Yung [4] have proposed Anonymity 2.0 as a way to use X.509 certificates in the group signature scheme setting (as well as in the case of traceable and ring signatures). In a nutshell, they consider a unique X.509 certificate for the whole group, which one is valuable to all group members. In the proposed X.509 certificate, the *Serial Number* and the *Validity Period* are fixed, the *Subject Name* field is the identity Id_G of the whole group and the *Subject Public Key Information* field contains the public key gpk of the whole group.

However, there are three main issues to solve regarding the use of X.509 certificates for group signatures.

1. Using Anonymity 2.0 [4], the "real identity" of the user is lost since the X.509 certificate is no more related to one unique signer.
2. X.509 standard includes the way to revoke one particular signer. Regarding Anonymity 2.0, either each revocation in the group modifies the group public key gpk and thus the corresponding X.509 certificate, which is a very expensive solution, or the X.509 certificate is not used for the revocation purpose, which makes Anonymity 2.0 not in accordance with the X.509 standard.
3. Anonymity 2.0 can not be applied in the context of anonymous credentials.

It also exist some work on the way to use X.509 certificates while protecting the privacy of the owner of the certificate. For example, the RFC 5636 on traceable anonymous certificate [28] defines a practical architecture and protocols such that the X.509 certificate contains a pseudonym, while still retaining the ability to map such a certificate to the real user who requested it. The architecture separates the authority responsible for the verification of the ownership of a private key and the one who validate the contents of a certificate. In [29], Persiano and Visconti propose the concept of Secure and Private Socket Layer protocol, that is an extension of the SSL/TLS protocol that allows the server to present a list of certificates and the client to prove that she owns the private key related to at least one of those certificates, without revealing which one. But none of these two approaches focuses on group signatures nor anonymous credentials, which is the case in this paper.

In fact, we propose a new approach, called One-Time Anonymous Certificate (OTAC), which permits to better use group signatures with X.509 certification. Our solution also solve all the above issues and, this way, permits the user to become the center of the whole system.

The paper is organized as follows. Section 2 presents the concept of group signature schemes, and describes some existing constructions. In Section 3, we recall X.509 certification and give some words on Anonymity 2.0. Section 4 describes our work and compares it with other proposed solutions to make X.509 supporting anonymity. The Section 5 finally study the application of OTAC to anonymous credentials.

2 Group Signature Schemes

We first focus on group signature schemes which have been introduced in [17]. In such scheme, any member of the group can sign messages on behalf of the group. Such signature remains anonymous and unlinkable for anyone except a designated authority who has the ability to identify the signer. This entity is sometimes called the Opening Manager. The group is typically controlled by a Group Manager $\mathcal{GM}$ that handles enrollment of members.

2.1 Concept of Group Signatures

In the following, we consider a group which is publicly identified by a unique identifier denoted Id_G. This group can be dynamic in the sense that group members can enter or leave the group at any time during the life cycle of this group. Following [3], a group signature scheme is a digital signature scheme comprised of the following procedures.

- Setup is a probabilistic algorithm which on input a security parameter 1^λ, outputs the initial group public key gpk, the secret key gmsk for the group manager and the opening secret key osk.
- UserKg is a user key generation which on input a user i and the group public key gpk, outputs a personal public and private key pair $(\mathsf{upk}[i], \mathsf{usk}[i])$. We assume that $\mathsf{upk}[i]$ is public.
- Join is a protocol between the group manager, on input gmsk and a user i on input $(\mathsf{upk}[i], \mathsf{usk}[i])$ that results in the user becoming a new group member. The user's output is a membership secret $\mathsf{gsk}[i]$, which may include $(\mathsf{upk}[i], \mathsf{usk}[i])$. $\mathcal{GM}$ makes an entry $\mathsf{reg}[i]$ in the registration table which is included into gpk.

- Sign is a probabilistic algorithm that on input a group public key gpk, a membership secret gsk$[i]$ and a message m outputs group signature σ of m.
- Verify is an algorithm for establishing the validity of a group signature σ of a message m with respect to the group public key gpk.
- Open is an algorithm that, given a message m, a valid group signature σ on it, the group public key gpk and an opening secret key osk, determines the identity i of the signer, together with a proof τ that the opening has been correctly done. This algorithm output $i = 0$ in case no group member has produced this signature.
- Judge takes on input the group public key gpk, an integer $j \geq 1$, the public key upk$[j]$ of the user j, a message m, a valid signature σ of m, and a proof τ. Its aim is to check that τ is a proof that user j has truly produced σ.

As described in [3], a secure group signature scheme must satisfy the following properties. Note that we do not give the formal definitions since our aim in this paper is not to study in detail the security properties of some schemes.

- **Correctness**: a signature which is produced by the group member i using Sign must be accepted by Verify. Moreover, in case of opening, the Open procedure should output i together with a proof τ which is accepted by the Judge algorithm.
- **Anonymity**: an adversary should not be able to decide whether one given signature has been produce by two known group members of her choice. The adversary has chosen both members and may know their respective secret keys. The adversary has also access to both group manager's secret keys gmsk and can ask for the opening of group signatures (except for the given signature).
- **Traceability**: this property describes that an adversary is not able to produce a signature such that either the honest opener declares herself unable to identify the origin of the signature (that is Open outputs $i = 0$), or she has find the signer but is unable to produce a correct proof τ of its claim.
- **Non-frameability**: the adversary should not be able to create a proof, accepted by the Judge algorithm, that an honest user produced a certain valid signature unless this user really did produce this signature.

2.2 Constructions of Group Signature Schemes

Group signatures have been the subject of many research papers over the past years. Currently proposed group signatures have been proved secure in the random oracle model [1, 5, 18], or in the standard one [21].

Zero-knowledge proof of knowledge. Roughly speaking, a zero knowledge proof of knowledge is an interactive protocol during which an entity proves to a verifier that he knows a set of secret values $\alpha_1, \ldots, \alpha_q$ verifying a given relation R without revealing anything else. These protocols are also used to prove that some public values are well-formed from secret ones known by the prover.

In the sequel, we denote by $\textsc{Sok}(\alpha_1, \ldots, \alpha_q : \mathsf{R}(\alpha_1, \ldots, \alpha_q))(m)$ a signature of knowledge based on a proof of knowledge of the secrets $\alpha_1, \ldots, \alpha_q$ verifying the relation R, and where m is the message to be signed. As shown in [14, 9], it is today possible to prove discrete-logarithm based predicates (e.g. representation, equality of discrete logarithms, belonging to a public interval).

Constructions in the random oracle model. Group signature scheme constructions that are secure in the random oracle model [1, 8, 5, 20, 18] are based on the use of signatures of knowledge [13], that is zero-knowledge proofs of knowledge transformed into signatures using the Fiat-Shamir heuristic [19].

In a nutshell, such group signature schemes are based on the same structure. Namely, during the Join procedure, the new group member obtains from the group manager a (Camenisch-Lysyanskaya [11] type) signature on one secret computed by both entities but only known by the member. The group signature is next the proof of knowledge of the signature from $\mathcal{GM}$ on the secret, without revealing the signature nor the secret, to ensure anonymity. We recall below the ACJT group signature scheme [1] and give some words on the BBS one [5].

The ACJT group signature scheme. In this scheme[3], the underlying signature scheme [11] is based on the Flexible RSA assumption (a.k.a. Strong RSA assumption). The secret key, denoted x is signed by the group manager and the resulting signature is the couple (A, e) such that $A^e = a_0 a^x \pmod n$ where n is a safe RSA modulus and a_0 and a are publicly known random elements of $QR(n)$, the group of quadratic residues modulo n. The secret key of the group manager is the factorization of n, which permits to choose at random e and to compute the corresponding A in the signature.

The group signature next consists in first encrypting one part of the obtained signature using the public key related to the opening secret one. In the ACJT case, the encryption is done by using the El Gamal encryption scheme, as

$$T_1 = Ay^w, T_2 = g^w, T_3 = g^e h^w$$

where y is the public key corresponding to the opening secret key θ (that is, $y = g^\theta$). The couple (T_1, T_2) is the El Gamal encryption of A while T_3 corresponds to a commitment on e. Note that these values, denoted in the following $K_g = (T_1, T_2, T_3)$, do not depend on the message m to be signed. The second part of the group signature, denoted S_m in the following, is the signature of knowledge on the message m, which can be written as

$$U = \textsc{Sok}\big[x, e, w, ew : a_0 = T_1^e/(a^x y^w e) \wedge T_2 = g^w \wedge 1 = T_2^e/g^{we} \wedge T_3 = g^e h^w\big](m)$$

with the message m on input. The group signature σ on m is finally (T_1, T_2, T_3, U), which correspond to the couple (K_g, S_m).

[3] One better solution in terms of efficiency and security has afterward been proposed by Camenisch and Groth in [8] but we do not need to detailed it in this paper, since both solutions can be used in our result and the ACJT description is easier to give.

The BBS group signature scheme. The BBS group signature scheme [5] and its variants [20, 18] are based on the q-SDH assumption. For example in [20, 18], the secret key, denoted y, is signed by the group manager and the resulting signature is the couple (A, x) such that $A^{\gamma+x} = g_0 h^y$ in a group of prime order. γ is the secret key of the group manager and g_0 and h are publicly known generators.

Next, the group signature is composed of a ciphertext and a signature of knowledge. For example, in the XSGS variant [18] of the BBS group signature scheme, the encryption is done by using the double El Gamal encryption scheme. Next, the first part of the group signature scheme consists in

$$T_1 = Ay_1^\alpha, T_2 = g^\alpha, T_3 = Ay_2^\beta, T_4 = g^\beta,$$

where y_1 and y_2 are the public keys corresponding to the opening secret keys ζ_1 (that is, $y_1 = g^{\zeta_1}$) and ζ_2 (that is, $y_2 = g^{\zeta_2}$). The tuple (T_1, T_2, T_3, T_4) is the double El Gamal encryption of A and does not depend on the message m. The second part of the group signature is the signature of knowledge on the message m, which can be written as

$$U = \textsc{Sok}\big[\alpha, \beta, x, z : T_2 = g^\alpha \wedge T_4 = g^\beta \wedge T_1/T_3 = y_1^\alpha / y_2^\beta \wedge \\ e(T_1, g_2)^x e(h, w)^{-\alpha} e(h, g_2)^z = e(g_1, g_2)/e(T_1, w)\big](m)$$

where g_1 and g_2 are random generators, $z = x\alpha + y$, $w = g_2^\gamma$ and with the message m on input. The resulting group signature is composed of the (double) El Gamal or linear encryption $K_g = (T_1, T_2, T_3, T_4)$ and the signature of knowledge $S_m = U$ on the message m. The group signature is again a couple of the form (K_g, S_m) where K_g does not depend on the message, while S_m does.

Constructions in the standard model. Groth Sahai NIWI proofs [22] permit to prove to a third party that some given values are well-formed (they lie in the correct given language) but do not permit to prove the knowledge of these values. As it is sometimes necessary to prove the knowledge of some secret values related to the group membership (e.g. in group signature schemes), the use of Groth-Sahai technique is not enough.

In [21], Groth proposes to use certified signatures. During the Join protocol, each group member obtains a signature v_i on a user secret key (x_i, a_i, b_i). During the signature procedure, the signer chooses at random a new key pair $(\mathsf{vk}_{sots}, \mathsf{sk}_{sots})$ for a one-time signature, produces a certificate σ and two intermediary values (a, b) (one a which is revealed and the other, b, which is kept secret by the signer) of the corresponding public key using her certified secret key. The Groth signature also encrypts the value v_i to open the group signature if needed. She next produces proofs (π, ψ) that all the above values are well-formed. Finally, she signs the message m and the values $\mathsf{vk}_{sots}, a, \pi, y, \psi)$, using the one-time secret key, obtaining σ_{sots}. The final group signature is $(\mathsf{vk}_{sots}, a, \pi, y, \psi, \sigma_{sots})$, which is again of the form (K_g, S_m) with $K_g = (\mathsf{vk}_{sots}, a, \pi, y, \psi)$ (not depending on m) and $S_m = \sigma_{sots}$ (depending on m).

2.3 Group Member Revocation

It may be necessary, in some cases, to handle the situation in which a group member wants to leave a group or is excluded by the Group Manager. In both cases, it is necessary to set up a mechanism to prevent the possibility for a revoked member to produce a valid group signature. It exists today two different methods to revoke a group member in a group signature scheme, that is, make it infeasible for a membership secret $\mathsf{gsk}[i]$ to be used to produce a group signature which will be accepted by the VERIFY algorithm: the use of accumulators or the verifier-local revocation approach.

It is moreover necessary to modify the above security model. In fact, it currently exists two different models dealing with group member revocation, one for each underlying solution: accumulator technique [10], or verifier local revocation [6, 24]. We do not detail these models as it is not really necessary to the understanding of our paper.

The accumulator technique. This technique has been introduced by Camenisch and Lysyanskaya [10] and is based on the use of dynamic accumulators [10, 27]. In a nutshell, $\mathcal{GM}$ publishes a single value v which accumulates a group member secret key per valid group member. The group signature should next include a zero-knowledge proof that the member knows a secret value and a corresponding witness that this value is truly accumulated in v. It should be hard for users outside the group to forge such proof (by finding an appropriate witness). The accumulator is next updated each time a new user becomes a group member (a new value is added into the accumulator) and after each revocation (the value of the revoked group member is deleted from the public accumulator).

The main problem with this method is that it implies for each group member to update her secret data (more precisely a witness that her value is truly accumulated) after each modification (addition and deletion) in the group.

The Verifier Local Revocation (VLR) technique. This technique has been proposed by Boneh and Shacham [6]. The idea behind is to manage a revocation list with a data for each revoked group member. During the SIGN process, the group member produces an extra value which is used by the verifier who run through the revocation list to test, for each entry, if this is related to the value used to produce the signature.

This gives a solution without any update for group members. However, the group public key gpk needs to be regularly updated to avoid backward linkability (see *e.g.* [26]). More importantly, the time complexity from the verifier's side is linear in the number of entries in the revocation list.

Using one of these solutions, a revocation mechanism can thus be used for group signatures with the X.509 principles. The side effect is that each group member needs to produce two different group signatures, which makes our solution less efficient than the Anonymity 2.0 one [4]. However, our solution is more efficient considering the revocation mechanism since the use of Anonymity 2.0 implies the creation of a new certificate at each modification within the group.

2.4 Anonymous credential

In the context of anonymous credentials, users have to show a kind of tokens to prove statements about themselves. For this purpose, each user has one or several credentials which are issued by some organizations that ascertain the authenticity of the information and can be provided to check things on demand. The certified attributes can be *e.g.* a name, an address, an age, etc. Users may also be required to prove a predicate on the attributes encoded in their credentials, such as for example that her age is greater than a fixed value, revealing neither their age nor other attributes.

The main requirements an anonymous credential systems are (i) *unforgeability* which states that the user can not prove the validity of forged credentials or predicates that are encoded on her issued credential and (ii) *privacy* which states that the verifier should not be able to learn any information about the user's credentials (e.g. other attributes) beyond what can be logically inferred from the status of the proven predicate.

It is possible to construct an anonymous credential system using the same techniques as group signature schemes. For example [12], it is possible to construct such system based on the ACJT group signature scheme as follows. The credential (A, e) is of the form $\mathsf{A}^{\mathsf{e}} = \mathsf{a}_0 \mathsf{a}_1^{c_1} \cdots \mathsf{a}_\ell^{c_\ell} \mathsf{b}^{\mathsf{x}} \pmod{\mathsf{n}}$ where n is a safe RSA modulus, $\mathsf{b}, \mathsf{a}_0, \cdots, \mathsf{a}_\ell$ are publicly known random elements of $QR(\mathsf{n})$, the c_i's are the certified attributes (for example c_1 represents the nationality, c_2 the address, c_3 the date of birth, etc.) and x is a secret value only known by the user, but jointly computed.

The proof of possession of a credential is done similarly as for a group signature. For example, if one user wants to prove that her first attribute (her nationality) is the value c_1, she has first to compute

$$\mathsf{T}_1 = \mathsf{A}\mathsf{y}^{\mathsf{w}}, \mathsf{T}_2 = \mathsf{g}^{\mathsf{w}}, \mathsf{T}_3 = \mathsf{g}^{\mathsf{e}}\mathsf{h}^{\mathsf{w}}$$

where w is a random value and next to produce the signature of knowledge

$$\begin{aligned}\mathsf{U} = \textsc{Sok}\big[&\mathsf{x}, \mathsf{e}, \mathsf{w}, \mathsf{ew}, c_2, \cdots, c_\ell : \mathsf{a}_0 \mathsf{a}_1^{c_1} = \mathsf{T}_1^{\mathsf{e}} / (\mathsf{a}_2^{c_2} \cdots \mathsf{a}_\ell^{c_\ell} \mathsf{b}^{\mathsf{x}} \mathsf{y}^{\mathsf{we}}) \wedge \\ &\mathsf{T}_2 = \mathsf{g}^{\mathsf{w}} \wedge 1 = \mathsf{T}_2^{\mathsf{e}} / \mathsf{g}^{\mathsf{we}} \wedge \mathsf{T}_3 = \mathsf{g}^{\mathsf{e}}\mathsf{h}^{\mathsf{w}}\big](m)\end{aligned}$$

with the message m on input. The user sends $(c_1, \mathsf{T}_1, \mathsf{T}_2, \mathsf{T}_3, \mathsf{U})$ to the verifier who verifies the signature of knowledge to be convinced that the credential embed the certified attribute c_1, as expected.

3 X.509 Certification and Anonymity

In this section, we recall X.509 certification principles and we describe the paper from Benjumea *et al.* [4], which is the first to propose the use of X.509 certification for signature schemes with anonymity (group, traceable and ring signatures).

3.1 X.509 Certification

X.509 is an ITU-T standard for a public key infrastructure (PKI). Its aim is to make the link between a public key and an entity, since public key cryptography, for example in the signature setting, only permits to know that one signature has been produced by this particular public key but not by this particular entity. In a nutshell, a X.509 certificate is the certification by a trusted authority called the Certification Authority (CA) that the public key in the certificate belongs to the identity in this certificate.

Thus, when sending a signed message, one has to give the message m, the signature σ produced using her secret key, and the X.509 certificate on the corresponding public key. The verification step consists next in verifying the validity of the certificate, extracting the verification public key and using it to verify σ on m. More precisely, all X.509 certificates have the following data, in addition to the signature from the CA on all these fields.

- *Version*: this identifies which version of the X.509 standard applies to this certificate, which affects what information can be specified in it. We do not detail this field as it is not really important in our study, except that we should use X.509 Version 3, which is the most recent one.
- *Serial Number*: the entity that created the certificate is responsible for assigning it a serial number to distinguish it from other certificates it issues. This information is used in numerous ways, for example when a certificate is revoked its serial number is placed in a Certificate Revocation List (CRL).
- *Signature Algorithm Identifier*: this identifies the algorithm used by the CA to sign the certificate.
- *Issuer Name*: the X.500 name of the entity that signed the certificate. This is normally a CA. Using this certificate implies trusting the entity that signed this certificate[4].
- *Validity Period*: each certificate is valid only for a limited amount of time. This period is described by a start date and time and an end date and time, and can be as short as a few seconds or almost as long as a century. The chosen validity period depends on a number of factors, such as the strength of the private key used to sign the certificate or the amount one is willing to pay for a certificate. This is the expected period that entities can rely on the public value, if the associated private key has not been compromised.
- *Subject Name*: the name of the entity whose public key is embedded into the certificate. This name uses the X.500 standard, so it is intended to be unique across the Internet. This is the Distinguished Name (DN) of the entity.
- *Subject Public Key Information*: this is the public key of the entity being named, together with an algorithm identifier which specifies which public key cryptosystem this key belongs to and any associated key parameters.
- *Extensions*: there are today some common extensions in use. *KeyUsage* limits the use of the keys to particular purposes such as "signing-only", *AlternativeNames* allows other identities to also be associated with this public key,

[4] Note that in some cases, such as root or top-level CA certificates, the issuer signs its own certificate.

e.g. DNS names, Email addresses, IP addresses. Extensions can be marked critical to indicate that the extension should be checked and enforced/used. For example, if a certificate has the KeyUsage extension marked critical and set to "keyCertSign" then if this certificate is presented during SSL communication, it should be rejected, as the certificate extension indicates that the associated private key should only be used for signing certificates and not for SSL use.

3.2 Anonymity 2.0

X.509 public key certificates were designed to support the concept of one public key, corresponding to a unique private key, for one identity, referring to the one who has the private key. As a consequence, this does not support the anonymity of the signer, by construction. If one want to use such X.509 certificate structure for group signature, one has to face to the problem that the group member key $\mathsf{reg}[i]$ (see group signatures above) can not be transcript in the certificate, as it reveals the identity of the group member.

Benjmumea *et al.* propose in [4] to consider one single standard X.509 certificate for the whole group. The elegant model which is given in [4] is based on adding some semantic extensions, keeping the same X.509 structure. More precisely, they define a X.509 public key certificate with extended semantic where the public key is not bound to a single entity but it is bound to a concept (see the Appendix A in [4] for the specification of extension fields in ASN.1). Considering the above structure of an X.509 certificate, they propose the following modifications.

1. Fix the *Serial Number*.
2. Fix the *Validity Period*, that is the start date/time and the end date/time.
3. Put the identity of the group Id_G in the *Subject Name* field.
4. Put the public key of the group gpk in the *Subject Public Key Information* field.

In fact, this concept stated that members belonging to the same group possess the same X.509 certificate. Thanks to their solution we get the desired anonymity and their solution can next be applied for group signatures, but also for traceable and ring signatures, as shown in [4].

3.3 Remaining Issues

However, there remains three main issues to solve regarding the use of X.509 certificates for group signatures.

1. One aim of X.509 certificates is to make the link between one public key and one identity. However, using Anonymity 2.0 [4], the "real identity" binding the X.509 certificate is lost since it is no more related to one unique signer, but to the whole group. From this point of view, Anonymity 2.0 is not totally in accordance with the X.509 standard principles.

2. X.509 standard includes the way to revoke one particular signer. Using for example a revocation list, this is possible to put the certificate corresponding to a revoked signing secret key onto this revocation list so that, in case of fraud, a signature produced by this private key is no more accepted. Thus, in this case, each time a signature and a certificate is received, the verifier needs to verify whether this certificate belongs to the revocation list. Regarding group signature schemes, this is not completely possible using Anonymity 2.0. In fact, there are two ways to consider revocation using Anonymity 2.0.
   (a) Each revocation in the group modifies the group public key gpk and thus the corresponding X.509 certificate. As a consequence, if a revoked member uses the wrong group public key to be accepted, then the X.509 will be refused and the whole group signature rejected. This is thus necessary to put each time the previous X.509 certificate onto the revocation list and next to recreate a new one for the whole group. This is therefore a very expensive solution.
   (b) The X.509 certificate is not used for the revocation purpose. This way, the revocation is only done using the simple group signature and usual techniques (see Section 2.3). The X.509 certificate is here only used to make the link between the group public key and the group in which belong the signer. With such solution, Anonymity 2.0 is again not in accordance with the X.509 standard.
3. Anonymity 2.0 can not be applied in the context of anonymous credentials (see Section 2.4), which can be described as a way to manage several groups (group of people who live in the same town, group of people being more than 65 years old, group of people being a student, etc.) in an efficient and compact way. Using Anonymity 2.0 in such context, having one X.509 certificate per group, implies as many certificates as there are possibilities of attributes, which makes it unusable in practice.

In the following, we design a new way to consider X.509 certificates in the context of group signature schemes and anonymous credential systems in such a way that the proposed solution is totally in accordance with the X.509 standard, without the multiplication of revoked and/or issued certificates.

4 One Time Anonymous Certificate

In this section, we present our result on X.509 certification for group signature schemes. We name it OTAC for One-Time Anonymous Certificate and we here give a general overview and next detail our system.

4.1 Overview of Our Solution

We want to add to a message m and its group signature σ a X.509 certificate which

1. can be used by the verifier to verify the group signature, as a standard X.509 certificate;

2. does not compromise the security aspects of group signature schemes (see Section 2.1);
3. is unique for a given group member and
4. directly permits the management of revocation of group members.

The idea is to put on the *Subject Public Key Information* field a cryptographic key which is at the same time unique for a given group member and different from one signature to another (to obtain the anonymity property). This "public key" should be generated by the group member as many time as she wants and should not depend on the message to be signed (since it is included into the certificate).

If we examine different group signature schemes (See Section 2.2), we remark that for each of them, the final group signature is divided into two parts. The first one, denoted K_g does not depend on the message to be signed and is related to the identity of the group member. The second one, denoted S_m, depends on the message m. Moreover, the first part K_g includes the way for the Opening Manager to open the group signature (see (T_1, T_2) for the ACJT group signature [1] or y for the Groth one [21]). In Figure 1, we resume the way current group signatures are divided in that way.

Group signature	K_g	S_m
ACJT [1]	T_1, T_2, T_3	U
BBS [5, 18]	T_1, T_2, T_3, T_4	U
Groth [21]	$vk_{sots}, a, \pi, y, \psi$	σ_{sots}

Fig. 1. The K_g and S_m variables for some group signatures

As a consequence, regarding the X.509 certificate as described in Section 3.1, we can consider that the key K_g corresponds to the *Subject Public Key Information*, while S_m is a true signature (of knowledge) on the message m, using, in some senses, the "public key" K_g. We thus obtain our "one-time" X.509 certificate.

Focusing on the X.509 certificate signature, it is obvious that we cannot ask for the certification authority to perform this task each time a group member produces a group signature, for two obvious reasons. First, this implies that this authority is always on-line, which comes against the principle of certification. Second, this goes against the anonymity of the user.

In this paper, we adopt a new approach which consists in delegating to each group member the power to sign a certificate. As it is necessary for the signer to be anonymous, we introduce the concept of "major group" in which each user is thus able to produce a major group signature on the above one-time X.509 certificate. Consequently, each group member lies into two (or more) groups, the "major" one for signing a X.509 certificate and the "minor" one(s) for the purpose of the initial group.

4.2 Detailed Description

In the following, we consider a group, called the *minor group*, identified by Id_G, and related to a group signature scheme. This minor group is managed by a Group Manager and an Opening Manager and is composed of several group members. Each of them has a membership secret $\mathsf{gsk}[i]$ for this minor group and is thus capable of producing group signatures σ on messages m, on behalf of this minor group.

We now describe how to create a One Time Anonymous Certificate (OTAC) for group signatures.

Creation of an OTAC. On input a message m, the group member executes $\textsc{Sign}(\mathsf{gpk}, \mathsf{gsk}[i], m)$ and obtains the group signature σ. We here remember that we use one of the group signature that has been described in Section 2.2. Thus, the obtained signature is of the form (K_g, S_m) (see Section 2.2 and Figure 1). The user can now create the one-time certificate, based on the X.509 structure, with the following fields.

- *Serial Number*: as we consider that this certificate is one-time, and according to the standard X.509 certificate, the serial number should be different from one OTAC to another. It may for example be the (collision-resistant) hash value of K_g;
- *Signature Algorithm Identifier*: the signature algorithmID must be redefined to be the major group signature. We detail this step just below;
- *Issuer Name*: the issuer name will be the CA;
- *Validity Period*: according to [4] the validity period needs to be fixed by *e.g.* the CA and is consequently common to all group members;
- *Subject Name*: the subject name should not give any information about the identity of the group member. It may for example be the hash value of $K_g \| Id_G$;
- *Subject Public Key Information*: this field is defined by the user to be the value K_g outputted by the execution of $\textsc{Sign}(\mathsf{gpk}, \mathsf{gsk}[i], m) = \sigma = (K_g, S_m)$.

Remark 1. As this is the user herself who fill in these fields, one can argue that the verifier may not be convinced that those values have been correctly computed. In fact, for most of them (serial number and subject name), it is in the user's interest to fill in them as described above. For the other ones (Signature Algorithm ID, Issuer Name, Validity Period), it is possible for the verifier to check them using a parent certificate.

Note that this certificate is necessarily one-time since it includes the value K_g, which is specific to this signature, for unlinkability purpose. It now remains to sign this certificate. As said above, this is not possible to ask the certification authority to sign this message, since it goes against the essence of X.509 certification (the CA should be off-line) and compromises the user anonymity.

Signature of the OTAC. To cope with the above problem, our solution consists in enabling the user to sign certificates on behalf of the CA. As the user should be anonymous, we thus use again a group signature scheme. For this purpose, the user should also belong to a *major group*, managed by the Certification Authority himself. Thus, each above user should interact with the CA using the JOIN protocol of the major group signature scheme, to obtain a membership secret here denoted $\mathsf{GSK}[i]$. She can now produce a group signature Σ on some message M.

After having creating a certificate and fulfilled the different fields as described above, the generated certificate should be hashed in order to get m_{cert} that plays the role of a generic message m. The group member next generates a "major" group signature Σ which play the role of the signature of the X.509 certificate (performed by the CA in the standard certification process).

Finally, the complete certificate OTAC is composed of the above fields, including the subject public key with K_m, and the complete major group signature Σ.

General scenario. After having created the certificate, the user can send all the information to the verifier, that is (i) the message m, (ii) the signature S_m from the minor group signature and (iii) the above signed OTAC. The verifier next first verifies the validity of the certificate by using the VERIFY algorithm for the major group on input the signature Σ and the message m_{cert} (that is, the hash value of all the fields in the X.509 certificate). Next, the verifier gets back from the OTAC the value K_g and uses it with the signature S_m on the message m on input of the VERIFY procedure for the minor group signature. If both verifications succeed, then the signature is accepted.

Remark 2. In the standard PKI setting, a fraudulent user may give her X.509 certificate and the corresponding secret key to another user so that the verification is falsely said to be correct. We have the same problem with our above description since one fraudulent user can give her major group membership secret key to another user (which is not necessarily a group member in this major group). There are two ways to deal with this problem. First, we can assume that the Group Manager for the minor group does not accept a user which is not a group member in the major group. This also implies that this Group Manager and the Certificate Authority communicate one to each other in case of revocation of a user. The second solution consists in checking that the two valid group signatures $\sigma = (K_g, S_m)$ and Σ are based on the same initial secret. In fact, this is easily possible in the group signature schemes described in Section 2.2, as shown in Section 5.4.

4.3 Conclusion on OTAC for Group Signature Schemes

As a first conclusion, our OTAC proposal for group signature schemes solves all the issues considered in Section 3.3, except the case of anonymous credentials which is considered in the next section. More precisely, we have the following points.

Unicity. With OTAC, it is clear that the resulting X.509 certificate is unique for a given group member and is related to the identity of her. In fact, the certificate is related to K_g which includes, as described in Section 2.2, the encryption of a value which is used by the Opening Manager to open the group signature. OTAC is here totally in accordance with the X.509 standard.

Revocation. In Section 2.3, we have described the two ways to deal with the revocation of a group member in group signature schemes. It is relatively obvious that our OTAC solution can easily include both mechanisms. In fact, we only need to add a revocation mechanism to the minor group (even if this is also possible for the major one), which makes the revocation totally in accordance with the revocation principles in the X.509 standard.

1. **Accumulator**: this revocation solution includes the modification of the group signature, that is the addition of some T_i's (see *e.g.* [10] for details in the ACJT case) which are included into K_g and the modification of the signature of knowledge U accordingly. If a group member is revoked, she will not be able to produce a valid U.
2. **VLR**: similarly, this mechanism add some values to K_g and implies the modification of the signature of knowledge U, accordingly. This time, the revocation is next based on the principle of a certificate revocation list (CRL) which is used by the verifier to check whether this certificate is related to a revoked user. This way, OTAC is in accordance with the X.509 standard.

In some cases, the Group Manager can inform the Certification Authority to revoke a particular user in the major group.

Security considerations. We should be careful that the addition of an OTAC does not compromise the security of the initial group signature scheme. As a consequence, the whole protocol (including the OTAC) should verify the correctness, anonymity, traceability and non-frameability properties, as described in Section 2 and in [3]. We here do not give a formal security proof but assuming that both chosen group signature schemes (minor and major) are secure in the BSZ sense [3] and according to the fact that the filled fields in OTAC does not compromise any security property, then it is clear that the global scheme is also secure in the BSZ sense. The latter argument can be validated by the fact that the fields depending on K_g (*Serial Number*, *Subject Name* and *Subject Public Key Information*) do not reveal any extra information that is not initially revealed. Finally, the other OTAC fields do not give any useful information to the adversary.

5 The Case of Anonymous Credentials

In this section, we show how our OTAC solution can be applied to anonymous credentials [2]. As described in Section 2.4, anonymous credentials permits users to give evidence to service providers that they have the right attribute (age, address, nationality, etc.) while revealing the minimum of information on the certified attributes. We here first recall the two ways to treat attributes with the X.509 standard. (X.509 Attribute Certificate or the addition of an extension to X.509 traditional certificates) and we next give our solution.

5.1 X.509 Attribute Certificate

An attribute certificate is a digital document that describes a written permission from the issuer to use a service or a resource that the issuer controls or has access to use. It is also known as an authorization certificate. The most recent ITU-T X.509 recommendation standardizes this concept of attribute certificate and describes how the attribute certificate should be used with a standard PKI X.509 certificate.

As shown in Figure 2, the Attribute Certificate is close to the identity certificate described in Section 3.1. The main differences come from the field *Holder* which contains the serial number of the related PKI X.509 certificate, and the *Attribute* one which contains the attributes of the related user. This certificate is signed by an authority designated to be the Attribute Authority (AA). The revocation process is defined using the concept of Attribute Certificate Revocation List, which is used in the same way as a Certification Revocation Lists in the PKI case.

Version Number	Holder
Serial Number	Attributes
Signature Algorithm	Issuer Unique Identifier
Issuer	Extensions
Validity Period	AA Signature

Fig. 2. The Attribute Certificate

5.2 An Alternative to X.509 Attribute Certificate

Most of recent applications do not use attribute certificate to carry permission or authorization information. User attributes (date of birth, address, nationality, etc.) are rather put inside a X.509 PKI certificate, using an extension field which contain a sequence of one or more attributes. This extension field, namely *Subject Directory Attributes*, is defined in ASN.1 as follows.

```
subjectDirectoryAttributes Extension::=  {
   SYNTAX AttributesSyntax
   IDENTIFIED BY id-ce-subjectDirectoryAttributes }
AttributesSyntax ::= SEQUENCE SIZE (1..MAX) OF Attribute
```

5.3 Using OTAC with the Extension Field

As described in Section 2.4, we suppose that a user obtains from an organization a credential which contains several user certified attributes $c_1, \cdots, c_\ell$. As we use the extension field, we can consider that the credential system is the *minor group* in our OTAC solution. Thus, following the description given in Section 2.4, the proof of possession of a credential is composed of a first part $K_g = (\mathsf{T}_1, \mathsf{T}_2, \mathsf{T}_3)$ and a second part $S_m = \mathsf{U}$. We can now apply our technique described in Section 4. As a first consequence, this user (as any other in the same case) belongs to the major group managed by the Certification Authority (CA). She can thus produce group signatures Σ on some message M, using her membership secret $\mathsf{GSK}[i]$.

When this user wants to prove that her first attribute c_1 is for example equal to the correct nationality (see our example in Section 2.4), she has to create a one-time certificate OTAC as described in Section 4, with the following modified fields:

- *Subject Public Key Information*: the public key contains $K_g = (\mathsf{T}_1, \mathsf{T}_2, \mathsf{T}_3)$;
- *Subject Directory Attributes*: this extension field contains the value of the revealed attribute c_1.

The signature of the OTAC is done by the user as a member of the major group, and is thus equal to Σ_M, that is the group signature on the hash of all the fields of the OTAC. The verification is done similarly as for the simple OTAC described in Section 4 and is not repeated again.

5.4 Using OTAC with the Attribute Certificate

We now describe how to use the attribute certificate with our OTAC solution. As described above, the Attribute Certificate is sent together with a standard X.509 PKI certificate. In our case, the idea is to replace the latter by our OTAC during the proof of possession. Before describing our solution, we first focus on the link between a group signature and an anonymous credential, based on Remark 2 (see Section 4.2).

Link between a group signature and an anonymous credential. In the following, we need to make a link between a group membership secret and a certified credential. For this purpose, we take the example of the ACJT based construction but, in fact, we need to slightly modify both. According to Section 2 and based on the result in [11], we remark that the group membership secret can also be taken as (A, e, v, x) such that $A^e = a_0 a^v b^x \pmod{n}$, where x is only known by the group member but computed by both her and the Group Manager, while v is randomly chosen by the group member. Moreover, this group member may have a credential on her attributes $(c_1, \cdots, c_\ell)$ of the form $(\mathsf{A}, \mathsf{e}, v, \mathsf{x})$ such that $\mathsf{A}^{\mathsf{e}} = \mathsf{a}_0 \mathsf{a}_1^{c_1} \cdots \mathsf{a}_\ell^{c_\ell} \mathsf{a}^v \mathsf{b}^{\mathsf{x}} \pmod{\mathsf{n}}$, with the same v, which does not compromise the security, as show in [11].

From these two related tuples, (A, e, v, x) and $(\mathsf{A}, \mathsf{e}, v, \mathsf{x})$, this becomes possible for the group member to provide a proof that both tuples are truly related. Thus, from

$$T_1 = Ay^w, T_2 = g^w, T_3 = g^e h^w and \mathsf{T}_1 = \mathsf{A}\mathsf{y}^{\mathsf{w}}, \mathsf{T}_2 = \mathsf{g}^{\mathsf{w}}, \mathsf{T}_3 = \mathsf{g}^{\mathsf{e}}\mathsf{h}^{\mathsf{w}},$$

which are computed at each group signature and proof of possession (see Sections 2.2 and 2.4), the group member can provide the following signature of knowledge

$$V = \textsc{SoK}\big[v, x, e, w, we, \mathsf{x}, \mathsf{e}, \mathsf{w}, \mathsf{we}, c_2, \cdots, c_\ell : a_0 = T_1^e/(a^v b^x y^{we}) \wedge \\ \mathsf{a}_0 \mathsf{a}_1^{c_1} = \mathsf{T}_1^{\mathsf{e}}/(\mathsf{a}_2^{c_2} \cdots \mathsf{a}_\ell^{c_\ell} \mathsf{a}^v \mathsf{b}^{\mathsf{x}} \mathsf{y}^{\mathsf{we}})\big](m),$$

which in particular proves that the same v is used in both equations. Given that, we are now able to describe our solution.

Our solution. We consider that a user is the member of a (minor) group, identified by Id_G, and can therefore produce group signatures on behalf of this group. Again, this user also belongs to a major group which permits her to sign X.509 based group signature OTAC on behalf of the Certification Authority, as explained in Section 4. Finally, the organization in the anonymous credential plays the role of the Attribute Authority AA.

According to the X.509 Attribute Certificate principle (see Section 5.1), the user has to send a message, a signature, a PKI X.509 certificate and a X.509 attribute certificate. In our solution, the PKI X.509 certificate is the OTAC as described in Section 4. Most of the fields (e.g. *Serial Number*, *Validity Period*) are unchanged from the initial description of OTAC, which includes the two following ones.

- *Subject Public Key Information*: this field is defined by the user to be the value $K_g = (T_1, T_2, T_3)$ outputted by the execution of $\textsc{Sign}(\mathsf{gpk}, \mathsf{gsk}[i], m) = \sigma = (K_g, S_m)$.
- *OTAC Signature*: this field is the "major" group signature Σ, using the membership secret $\mathsf{GSK}[i]$.

The Attribute Certificate is next constructed as follows.

- *Holder*: we put in this field the signature of knowledge V described above, which permits to make the link between the key embed in the OTAC and the anonymous credential.
- *Attribute*: this field contains the revealed attribute c_1. In case no attribute needs to be given (if the user has for example to prove that she is more than 65 years old), this field is set to the empty string.

– *AA Signature*: the Attribute Certificate signature is the whole proof of possession of an anonymous credential $(\mathsf{T}_1, \mathsf{T}_2, \mathsf{T}_3, \mathsf{U})$ as described in Section 2.4.

Remark 3. Another solution can be to have only one group which permits to sign both the message and the OTAC. This is more efficient but implies the delegation of the group management to a single authority.

References

1. Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In M. Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 255–270. Springer, 2000.
2. Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In *CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2009.
3. Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA 2005*, volume 3376 of *LNCS*, pages 136–153. Springer, 2005.
4. Vicente Benjumea, Seung Geol Choi, Javier Lopez, and Moti Yung. Anonymity 2.0 - x.509 extensions supporting privacy-friendly authentication. In *CANS 2007*, volume 4856 of *Lecture Notes in Computer Science*, pages 265–281. Springer, 2007.
5. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, 2004.
6. Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In *ACM Conference on Computer and Communications Security 2004*, pages 168–177. ACM, 2004.
7. Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *ACM Conference on Computer and Communications Security 2004*, pages 132–145. ACM, 2004.
8. Jan Camenisch and Jens Groth. Group signatures: Better efficiency and new theoretical aspects. In *SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 120–133. Springer, 2004.
9. Jan Camenisch, Aggelos Kiayias, and Moti Yung. On the portability of generalized schnorr proofs. In *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 425–442. Springer, 2009.
10. Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2002.
11. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN 2002*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer, 2002.
12. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO 2004*, volume 3152 of *LNCS*, pages 56–72. Springer, 2004.
13. Jan Camenisch and Markus Stadler. Efficient group signature schemes for large groups (extended abstract). In *CRYPTO 97*, volume 1294 of *LNCS*, pages 410–424. Springer, 1997.

14. Sébastien Canard, Iwen Coisel, and Jacques Traoré. Complex zero-knowledge proofs of knowledge are easy to use. In *ProvSec 2007*, volume 4784 of *LNCS*, pages 122–137. Springer, 2007.
15. Sébastien Canard, Berry Schoenmakers, Martijn Stam, and Jacques Traoré. List signature schemes. *Discrete Applied Mathematics*, 154(2):189–201, 2006.
16. Sébastien Canard and Jacques Traoré. List signature schemes and application to electronic voting. *Proceedings of Workshop on Coding and Cryptography (WCC'03)*, pages 81–90, 2003.
17. David Chaum and Eugene van Heyst. Group signatures. In *EUROCRYPT'91*, pages 257–265, 1991.
18. Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In *VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2006.
19. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, LNCS, pages 186–194. Springer, 1986.
20. Jun Furukawa and Hideki Imai. An efficient group signature scheme from bilinear maps. In *ACISP 2005*, Lecture Notes in Computer Science, pages 455–467. Springer, 2005.
21. Jens Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT 2007*, pages 164–180, 2007.
22. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432. Springer, 2008.
23. IBM. Idemix - identity mixer. http://www.zurich.ibm.com/security/idemix/, 2004.
24. Benoît Libert and Damien Vergnaud. Group signatures with verifier-local revocation and backward unlinkability in the standard model. In *CANS 2009*, volume 5888 of *Lecture Notes in Computer Science*, pages 498–517. Springer, 2009.
25. Microsoft. Microsoft U-Prove CTP. https://connect.microsoft.com/content/content.aspx?contentid=12505& siteid=642, 2010.
26. Toru Nakanishi and Nobuo Funabiki. Verifier-local revocation group signature schemes with backward unlinkability from bilinear maps. In *ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 533–548. Springer, 2005.
27. Lan Nguyen. Accumulators from bilinear pairings and applications. In *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 275–292. Springer, 2005.
28. S. Park, H. Park, Y. Won, J. Lee, and S. Kent. Traceable Anonymous Certificate. RFC 5636 (Experimental), August 2009.
29. Pino Persiano and Ivan Visconti. User privacy issues regarding certificates and the tls protocol: the design and implementation of the spsl protocol. In *ACM Conference on Computer and Communications Security*, pages 53–62, 2000.
30. Trusted Computing Group. Direct anonymous attestation. http://www.zurich.ibm.com/security/daa/, 2004.