An Efficient and Usable Multi-Show Non-Transferable Anonymous Credential System*

Giuseppe Persiano[1] and Ivan Visconti**[2]

[1] Dipartimento di Informatica ed Applicazioni.
Università di Salerno
via S. Allende - 84081 Baronissi (SA), Italy
`giuper@dia.unisa.it`

[2] Département d'Informatique
École Normale Supérieure
rue d'Ulm - 75230 Paris Cedex 05, France
`ivan.visconti@ens.fr`

Abstract. In an anonymous credential system a user can prove anonymously the possession of credentials to a service provider. Multi-show and non-transferability are two important properties of such systems. More precisely, in a *multi-show* system the same credential can be used more than once without threatening anonymity, moreover, lending of non-transferable credentials is *inconvenient.* In this paper we give a construction for multi-show non-transferable credentials for which the owner can prove that the credentials satisfy access control policies expressed by means of linear boolean formulae.

1 Introduction

Privacy enhancing technologies focus on the deployment of infrastructures that protect user privacy in the digital world. Currently many services that are fundamental for the worldwide economy are available on the Internet and many others are going to be supported. Several services are not accessible anonymously from everybody, but a form of access control is performed in order to distinguish qualified users (i.e., the ones that have enough rights to access the service) and unqualified ones. Current standard technologies implement such access control policies by requiring user identification (e.g., by using username and password or X509 [1] digital certificates), by retrieving user's credentials from a local database and then by checking that user's credentials satisfy a given access control policy. Such a typical scheme is secure, efficient, practical and guarantees that unqualified users do not get access to a restricted service. However such a scheme does not deal at all with user privacy protection since in each transaction between a user and a service provider the user reveals his identity, the transaction can

* Work partially supported by NoE ECRYPT

** Work done while Ivan Visconti was a temporary researcher at the Dipartimento di Informatica ed Applicazioni of the Università di Salerno.

be logged and an accurate dossier of all activities performed by the user can eventually be extracted from the logs.

In an *anonymous credential system* we have three types of players: organizations, users and service providers. Typically, a user gets a credential certificate encoding the user credentials from an organization and uses the credentials to access services. Each service provider specifies which users are qualified to access the service as a function of the credentials of the user. Thus, before accessing the service the user and service provider engage in a protocol in which the user proves possession of (a credential certificate encoding) credentials sufficient for the service in question. In an anonymous credential systems it is desired that the service provider cannot link a request for the service with a specific user or with other past requests. Roughly, we now list the properties that an anonymous credential system should enjoy.

1. **Security:** it is hard for a coalition of users to get access to a service without having the requested credentials.
2. **Multi-show privacy:** a user during a transaction can prove possession of credentials and, at the same time, the service provider does not obtain any private user information. This holds even if the user interacts using the same credential certificate several times with the same (or other) service provider.
3. **Usability:** a user that possesses a credential certificate should be able to prove *general statements* (in our case the satisfaction of linear Boolean formulae) over the credentials while preserving multi-show privacy.
4. **Non-Transferability:** it should be inconvenient for a user to lend his credentials to another user.
5. **Efficiency:** the overhead in terms of communication and computation imposed by the anonymous credential system to users and service providers must not heavily affect their performance.

Related Work. An anonymous credential system can be based on the concept of proofs in which a user shows possession of some piece of information (the credentials) that satisfies some given conditions (e.g., the access control policy). A first implementation of these proofs, for the case in which the conditions are expressed by a monotone Boolean formula, can be traced back to the general results on statistical zero-knowledge proof systems by [2] with efficiency communication improvements given by [3]. In [4, 5] these techniques are further explored and their applicability to real-life scenarios shown. In particular, Brands [4] presented a Public Key Infrastructure in which a user can prove in zero knowledge that the credentials encoded by his certificate satisfy a given linear Boolean formula. Brands' constructions are efficient since only a few modular exponentiations (i.e., the number of modular exponentiations is linear in the number of encoded credentials) have to be performed in order to prove that the credentials encoded in the certificate satisfy a given linear Boolean formula. The main drawback of Brands' certificates is that they are *one-show* in the sense that using the same certificate in two distinct transactions links the two transactions as performed by the same user. As a consequence of this weakness, a user needs to obtain from the Certification Authority an impractically large batch of certificates so that no certificate is used twice.

In our construction we will be interested in *multi-show* certificates that can be used several times still guaranteeing unlinkability. We will base our construction on some techniques proposed in [4, 5] in order to achieve proofs of possession of credentials that satisfy a given linear Boolean formula and we extend such schemes in order to achieve the *multi-show* property.

The first implementation of anonymous credentials has been presented in [6] where a third party is necessary in order to achieve unlinkability. Lysyanskaya et al. [7] proposed a general credential system that, however, is impractical being based on general one-way functions and zero-knowledge proofs. Moreover, they also presented an efficient one-show construction. The same drawback affects the solution proposed in [8]. The work of Camenisch and Lysyanskaya [9] has improved these pioneering works. In [9] the authors proposed an anonymous credential system that is based on the strong RSA assumption and the DDH assumption. In the system of [9] it is possible to unlinkably prove possession of a credential supporting the multi-show property and the entities that release credentials can independently choose their cryptographic keys. More recently, Verheul [10] proposed a more efficient solution for multi-show credentials. The result is based on the assumption that for some groups the Decisional version of the Diffie-Hellman (DDH) problem is easy while its Computational version (CDH) is hard and on an additional *ad-hoc* assumption.

The property of Brands' constructions, by which general statements of the values encoded by a certificate can be proved, is not enjoyed by the works of [9, 10]. Indeed, in such systems it is necessary to have one ad-hoc credential for each specific access control policy otherwise more information than the minimal one needed in each transaction is disclosed.

Finally, in [11], Persiano and Visconti presented a non-transferable anonymous credential system that is multi-show and for which it is possible to prove properties (encoded by a linear Boolean formula) of the credentials. Unfortunately, their proof system is not efficient since the step in which a user proves possession of credentials (that needs a number of modular exponentiations that is linear in the number of credentials) must be repeated k times (where k is the security parameter) in order to obtain a satisfying soundness.

Thus, the state of the art presents efficient solutions that either require one credential certificate for each service [9, 10] but each can be used as many times as needed; or require one credential certificate for each time the user needs to access a service [4] but a certificate can be used for any service.

Our contribution. In this paper, we present an anonymous credential system that is secure, multi-show, usable, non-transferable and efficient. In our system a user needs to obtain only *one* certificate to be used as many times as necessary with any service provider since our system enjoys the multi-show property. Moreover, in our construction we show how to enforce the non-transferability property. The security property of our system is based on new computational assumptions derived from the ones used in [12–14].

2 The model

Our model consists of three types of players:

1. The *organizations*, that release *credential certificates* to users.
2. The *users*, each with a set of credentials. A user receives a credential certificate encoding his credentials from an organization that he will then use to access services.
3. The *service providers*, that offer services and have access control policies for their services. We assume that the access control policy for each resource of each service is represented by a linear Boolean formula Φ over the required credentials.

A *credential system* $\mathcal{C}$ consists of the following five algorithms

$$(\texttt{SetUp}, \texttt{Enroll}, \texttt{IssueCred}, \texttt{ProveCred}, \texttt{VerifyCred}).$$

Next, we summarize how these algorithms interact and which of the parties executes each algorithm. In Section 4 we will present an implementation of such procedures.

1. **System set-up**: this step is performed only once by each organization in order to establish publicly verifiable parameters `Pub` (that will be used by the other procedures) and consists in executing algorithm `SetUp` on input the security parameter. The organization also obtains the private information `Priv` which corresponds to the public information `Pub` that she will use to release credential certificates.

   At the end of this phase, the organization is ready to release credential certificates.
2. **User enrollment**: this step is performed jointly by the user and by an organization and consists in executing the pair of algorithms (`Enroll`,`IssueCred`). The user has as input her credentials encoded by an m-tuple $(x_1, \cdots, x_m)$ and the public information published by the organization during the set up of the system.

   The organization verifies the credentials and then releases the credential certificate.
3. **Proving possession of credentials**: this step is performed by a user executing algorithm `ProveCred` interacting with a service provider executing `VerifyCred` in order to gain access to a service which is restricted to legitimate users.

   The user has as input her credential certificate, the credentials, the public information and the formula Φ that encodes the access control policy.

Our anonymous credential system guarantees the following properties:

Usability: once a user has obtained a credential certificate `CredCert` for credentials $(x_1, \cdots, x_m)$ from the organization then `CredCert` can be used (as input to algorithm `ProveCred`) to successfully access any service with access control formula Φ such that $\Phi(x_1, \cdots, x_m) = 1$. In other words, the user has to interact only once with the organization in order to get her credential certificate. From then on, the *same* credential certificate can be used to access any service (provided that the credentials satisfy the access control formula).

Security: it is computationally infeasible for a coalition of users to access the system without having a credential certificate; moreover, it is not possible to generate a new credential certificate even knowing a polynomial number of other credential certificates.

We formalize the security property in the following way.

Definition 1. *Let $\mathcal{C}$=(*`SetUp`*,* `Enroll`*,* `IssueCred`*,* `ProveCred`*,* `VerifyCred`*) be a credential system. We say that $\mathcal{C}$ is* secure *if for all probabilistic polynomial time algorithms A and for any formula Φ the probability that A on input the public information* `Pub` *and a sequence of credential certificates corresponding to credentials not satisfying Φ successfully interacts with algorithm* `VerifyCred` *on Φ is negligible.*

Multi-show privacy: no information about the credentials owned by a user is leaked by the protocol executed to prove possession of credentials other than the fact that the credentials satisfy the access control formula Φ. Moreover, two uses of the same credential certificate cannot be linked.

We formalize the property of multi-show privacy in the following way.

Definition 2. *Let $\mathcal{C}$=(*`SetUp`*,* `Enroll`*,* `IssueCred`*,* `ProveCred`*,* `VerifyCred`*) be a credential system. We say that $\mathcal{C}$ is* multi-show private *if for any triple of adversarial algorithms* ($\texttt{SetUp}^*$*,* $\texttt{IssueCred}^*$*,* $\texttt{VerifyCred}^*$) *there exists an algorithm S running in expected polynomial-time such that for any linear Boolean formula Φ, for all $(x_1, \cdots, x_m)$ for which $\Phi(x_1, \cdots, x_m) = 1$ it holds that*

$$S(\texttt{Pub}) = (\texttt{ProveCred}(x_1, \cdots, x_m, \texttt{CredCert}, \texttt{Pub}, \Phi) \leftrightarrow \texttt{VerifyCred}^*(\texttt{Priv}, \texttt{Pub}, \Phi))$$

where $(\texttt{Priv}, \texttt{Pub}) \leftarrow \texttt{SetUp}^*(1^k)$ *and* $\texttt{CredCert} \leftarrow (\texttt{Enroll}(x_1, \cdots, x_m, \texttt{Pub}) \leftrightarrow \texttt{IssueCred}^*(\texttt{Priv}, \texttt{Pub}))$.

Roughly, the formalization above states that the interaction between a user possessing a certificate for credentials $(x_1, \ldots, x_m)$ and a service provider (that could be the same organization that releases credential certificates and thus take as input `Priv`) can be efficiently simulated by an algorithm that has as input the public information and black-box access to the service provider.

Moreover (formal definition omitted from this abstract) in our construction the multi-show privacy holds even if the same credential certificate is used a polynomial number of times for a polynomial number of different Boolean formulae.

Our construction also enjoys the *non-transferability* property: it is "inconvenient" for a legitimate user to share her credentials with other users.

Finally, our construction is particularly *efficient* in the amount of information to be stored and in the amount computation to be performed (see Section 4.6) when a large number of credentials is considered used.

3 Background

In this section we summarize the main cryptographic techniques that we use in our constructions. For further details see [15, 4, 13].

We start by reviewing the notions of discrete logarithm and discrete logarithm representation.

Definition 3. *Given a group G of order n, and two elements g and y of G, the* discrete logarithm of y to the base g*, if it exists, is the integer* $0 \leq x \leq n-2$ *such that* $y = g^x$*.*

We stress that the discrete logarithm of y to the base g exists if and only if y belongs to the subgroup generated by g. This is the case, for example, if G is a finite cyclic group and g is a generator of G; in this case computing the discrete logarithm is considered hard. The discrete logarithm is also considered hard in the group Z_n^* where n is a composite integer. Indeed, if the discrete logarithm problem in Z_n^* can be solved in polynomial time, then n can be factored in expected polynomial time.

Definition 4. *Let G be a group of order n and let $y, g_1, \ldots, g_m \neq 1$ be elements of G. A* $(G, g_1, \ldots, g_m)$-DL representation *of y is a tuple $(x_1, \ldots, x_m)$ such that $0 \leq x_i \leq n-1$ for $i = 1, \ldots, m$ and $y = g_1^{x_1} \ldots g_m^{x_m}$.*

Moreover, for $i = 1, \ldots, m$, we call x_i the g_i-part *of the $(G, g_1, \ldots, g_m)$-DL representation $(x_1, \ldots, x_m)$ of y.*

Definition 5. *Let e be an element of Z_n^* co-prime with $\phi(n)$. A* (Z_n^*, e)-root of $y \in Z_n^*$ *is an element $x \in Z_n^*$ such that* $x^e \equiv y \pmod{n}$*.*

Note that if the factorization of n is unknown then computing e-th roots in Z_n^* is assumed to be infeasible (this is the RSA assumption).

We introduce the notion of RSA representation that has also been used in some of the constructions of [4, 5].

Definition 6. *Let $e \in Z_n^*$ be co-prime with $\phi(n)$ and let $y, g_1, \cdots, g_m \neq 1$ be elements of Z_n^*. A $(Z_n^*, e, g_1, \cdots, g_m)$-RSA* representation *(RSAREP) of y is a tuple $(x_1, \ldots, x_m, x)$ such that* $y \equiv g_1^{x_1} \cdots g_m^{x_m} x^e \pmod{n}$*, $0 \leq x_i < e$ for $i = 1, \ldots, m$ and $x \in Z_n^*$.*

In [4] it is shown how to choose the parameters so that the RSA representation problem can be reduced to the RSA problem.

3.1 Proofs of knowledge

We summarize the proofs of knowledge (PoKs) that will be used in our construction, for details see [13, 4, 16].

PoK of a discrete logarithm. On input (the description of) a cyclic group G of order n, a generator g of G, and an element $y \in G$, the prover P proves knowledge to the verifier V of x, the discrete logarithm of y to the base g.

PoK of a $(G, g_1, \ldots, g_m)$-DL representation. On input the description of a cyclic group G of order n and elements $y, g_1, \cdots, g_m$ of G, the prover P proves knowledge of a $(G, g_1, \ldots, g_m)$-DL representation $(x_1, \ldots, x_m)$ of y.

PoK of a $(Z_n^*, e, g_1, \ldots, g_m)$-RSA representation. On input $n, e, g_1, \ldots, g_m$ where (n, e) is an RSA public key and $g_1, \ldots, g_m$ are randomly chosen element of Z_n^*, the prover proves knowledge of a $(Z_n^*, e, g_1, \ldots, g_m)$-RSA representation $(x_1, \ldots, x_m, x)$ of $y \in Z_n^*$.

In such proofs of knowledge if the challenge of the verifier is in $\{0,1\}^k$ and $k = O(\log\log n)$, then the proofs are statistical zero-knowledge proofs of knowledge with soundness probability 2^{-k}. This probability can be reduced by sequentially repeating the protocols. If instead $k = \Omega(\log n)$, then the protocols are statistical witness indistinguishable. In particular by using such proofs, at the cost of one exponentiation per base it is possible to prove in a witness indistinguishable manner knowledge of a discrete logarithm or of a DL-representation or of an RSA-representation and still have negligible error probability.

PoK of an e-th root of the h_1-part of a (G, h_1, h_2)-DL representation. In [13], when e is a small integer, an efficient PoK by which it is possible to prove knowledge of a root of a part of a DL representation is presented. More precisely, we have the following theorem.

Theorem 7. *[13] Let G be a cyclic group of order n where n is the product of two safe primes. There exists a statistical zero-knowledge proof of knowledge for the polynomial-time relation $((G, h_1, h_2, e, y), (x_1, x_2, x))$, where h_1, h_2 are two elements of G, e is co-prime with $\phi(n)$, $y \in G$, (x_1, x_2) is a (G, h_1, h_2)-DL representation of y and $x \in Z_n^*$ is the e-th root of $x_1 \in Z_n^*$.*

The PoK of the previous theorem requires a number of modular exponentiations which is linear in e and thus is particularly efficient when $e = 3$ (as in our construction).

Recently, in [17] such a PoK has been improved by requiring only a number of modular exponentiations that is logarithmic in e.

PoK of the discrete logarithm to a based g of the h_1-part of a (G, h_1, h_2)-DL representation. On input (the description of) a cyclic group G of order n, where n is the product of two safe primes, two elements h_1, h_2 of G, a large order element g of Z_n^* and an element $y = h_1^{g^{x_1}} {h_2}^{x_2} \in G$, the prover P proves knowledge to the verifier V of the discrete logarithm x_1 to the base g of the h_1-part of the (G, h_1, h_2)-DL representation of y to the bases h_1, h_2.

Such a PoK can be given by repeating the following steps:

1. P computes $t = h_1^{g^{r_1}} h_2^{r_2}$ with randomly chosen r_1, r_2 and sends t to V;
2. V sends a bit b to P;
3. if $b = 0$ then P computes $s_1 = r_1$ and $s_2 = r_2$, otherwise he computes $s_1 = r_1 - x_1$ and $s_2 = r_2 - x_2 g^{s_1}$ and sends s_1, s_2 to V;
4. V accepts if $t = h_1^{g^{s_1}} h_2^{s_2}$ and $b = 0$ or $t = y^{g^{s_1}} h_2^{s_2}$ otherwise.

Theorem 8. *Given a cyclic group G of order n where n is the product of two safe primes, two elements h_1, h_2 of G and an element $g \in Z_n^*$, the previous protocol is an honest-verifier zero-knowledge proof of knowledge of the discrete logarithm x_1 to the base g of the h_1-part of a (G, h_1, h_2)-DL representation (g^{x_1}, x_2) of an element $y \in G$.*

Proving knowledge of a special representation. In [4, 5], Brands presents PoKs by which the prover can prove knowledge of a DL representation that satisfies a given linear Boolean formula Φ, i.e., a Boolean formula where each atomic proposition is a relation that is linear in the values of the DL representation (e.g., $((x_1 + 3x_2 = 4)$ OR (NOT $(5x_1 + 4x_3 = 6)))$ AND $(8x_2 + 3x_3 = 12))$.

The protocol is based on the following technique. For a finite cyclic group G and for any linear Boolean formula Φ, knowledge of a $(G, g_1, \ldots, g_m)$-DL representation $(x_1, \ldots, x_m)$ of y for which $\Phi(x_1, \ldots, x_m) = 1$ is equivalent to knowledge of a $(G, g'_1, \ldots, g'_m)$-DL representation $(x'_1, \cdots, x'_m)$ of $y' \in G$. The values $g'_1, \cdots, g'_m$ and y' only depend on the formula Φ and on $g_1, \ldots, g_m$ and y while $(x'_1, \cdots, x'_m)$ can be computed from $(x_1, \cdots, x_m)$. Thus, the prover and the verifier construct the auxiliary instance of the DL representation problem and run the protocol described above for proving knowledge of a DL representation. This implies that the above efficiency considerations regarding the PoK of a DL-representation apply also to the PoK in which knowledge of a DL-representation satisfying Φ is proved.

Theorem 9. *[4, 5] Let G be a finite cyclic group of order n and Φ be a linear Boolean formula, then there exists a statistical zero-knowledge proof of knowledge and a constant-round statistical witness indistinguishable proof of knowledge for the polynomial time relation $((G, u, g_1, \cdots, g_m, \Phi), (x_1, \cdots, x_m))$ such that $(x_1, \cdots, x_m)$ is a $(G, g_1, \cdots, g_m)$-DL representation of $u \in G$ and $\Phi(x_1, \cdots, x_m) = 1$.*

The same result can be extended also to the case in which we need to prove knowledge of an RSA representation and we have the following result.

Theorem 10. *[4, 5] Let Φ be a linear Boolean formula, (n, e) be an RSA public key, then there exists a statistical zero-knowledge proof of knowledge and a constant-round statistical witness indistinguishable proof of knowledge for the polynomial time relation $((Z_n^*, e, u, g_1, \cdots, g_m, \Phi), (x_1, \cdots, x_m, x))$ and it holds that $(x_1, \cdots, x_m, x)$ is a $(Z_n^*, e, g_1, \cdots, g_m)$-RSA representation of $u \in Z_n^*$ and $\Phi(x_1, \cdots, x_m) = 1$.*

4 Our implementation of credential certificates

We first introduce a computational assumption on which we base our construction of credential certificates.

4.1 Computational assumption

The security of a group signature scheme presented in [12] (based on the one presented in [13]) is based on the assumption that, on input an integer $n = p_1 p_2$ where p_1 and p_2 are primes of the same length, an integer e such that $\gcd(e, \phi(n)) = 1$ and $a, c \in Z_n^*$, it is hard to find in probabilistic polynomial time a pair (v, x) such that $v^e = a^x + c \pmod{n}$. Moreover, in [12] it is assumed that such a pair (v, x) is hard to find even if several other pairs are known. This property is used in order to prove the unforgeability of their scheme even with respect to coalitions of users. The computational assumption on which we are going to the base the security of our construction is a generalization of the assumption of [12].

We start with the following definition.

Definition 11. *Given an RSA public key (n, e) and elements $g, c, g_1, \ldots, g_l \in Z_n^*$ such that $g_1, \ldots, g_l \in \langle g \rangle$, we say that a tuple $(x_1, \ldots, x_l, x, v, z, y)$ such that, for $i = 1 \ldots, l$, $0 \leq x_i < e$, $0 \leq y < e$, $v, x \in Z_n^*$ and is a* good *tuple with respect to $(n, e, g, c, g_1, \cdots, g_l)$ if $v^3 \equiv g^y g_1^{x_1} \cdots g_l^{x_l} x^e + c g^z \pmod{n}$. Moreover, if $(x_1, \ldots, x_l, x, v, z, y)$ is a good tuple then we say that $(x_1, \ldots, x_{l-1})$ is its prefix.*

Consider the following game in which a probabilistic polynomial-time algorithm $\mathcal{A}$ receives as input:

1. an integer n such that $n = p_1 p_2$ where $p_1 = 2q_1 + 1, p_2 = 2q_2 + 1$, q_1, q_2 are primes of length k, $3 \not| (p_1 - 1)(p_2 - 1)$;
2. $e \in Z_n^*$ such that $3 \not| e$, $e \not| (p_1 - 1)(p_2 - 1)$;
3. large order elements $g, c \in Z_n^*$ and $g_1, \ldots, g_l$ for $l \geq 2$ that are elements of $\langle g \rangle$;
4. s such that $g \equiv s^3 \pmod{n}$;

and has access to an oracle $\mathcal{O}$ that, on input the sequence $(x_1, \ldots, x_{l-1})$ outputs a random good tuple $(x_1, \ldots, x_l, x, v, 0, 0)$ with prefix $(x_1, \ldots, x_{l-1})$ such that x_l is co-prime with both 3 and e.

We denote by $\mathrm{Succ}^{\mathcal{A}}(k)$ the probability that algorithm $\mathcal{A}$, running on the input described above, where n has length k, and having oracle access to $\mathcal{O}$ outputs a good tuple $(x'_1, \ldots, x'_l, x', v', z', y')$ whose prefix $(x'_1, \ldots, x'_{l-1})$ is different from any of the queries made to the oracle by $\mathcal{A}$.

We have the following assumption.

Assumption 12. *For all efficient algorithms $\mathcal{A}$, for all constants η and for all sufficiently large k*

$$\mathrm{Succ}^{\mathcal{A}}(k) \leq k^{-\eta}.$$

Given a good tuple $(x_1, \ldots, x_l, x, v, 0, 0)$, the tuple $(x_1, \ldots, x_l, x, vs^y, y, y)$ is also good for any value of $0 \leq y < e$. Indeed we will use exactly this property in order to achieve the multi-show privacy. However, we stress that in order to break our assumption it is necessary to produce a new tuple in which the prefix $(x_1, \ldots, x_{l-1})$ (which in our system encodes the credentials) is different from that of each original sub-tuple.

4.2 Our implementation

For the sake of simplifying the presentation, we now describe our system only for the case in which a credential certificate carries two credentials. We stress that modifying the system in order to support more than two credentials is straightforward.

System set-up. We now describe algorithm `SetUp` performed by an organization O.

Algorithm `SetUp`(1^k)

1. randomly pick two k-bit safe primes $p_1 = 2q_1 + 1, p_2 = 2q_2 + 1$ such that $\gcd(3, \phi(p_1 p_2)) = 1$, q_1, q_2 are primes and sets $n = p_1 p_2$;
2. randomly pick $e \in Z_n^*$ such that $\gcd(e, \phi(n)) = 1$ and $\gcd(3, e) = 1$;
3. compute $d \in Z_n^*$ such that $3d \equiv 1 \pmod{\phi(n)}$;
4. select element $g, c \in Z_n^*$ of large order;
5. randomly pick elements v_1, v_2, v_3 and set $g_1 \equiv g^{v_1}, g_2 \equiv g^{v_2}, g_3 \equiv g^{v_3} \pmod{n}$;
6. compute $s \equiv g^d \pmod{n}$;
7. compute a cyclic group G of order n in which computing the discrete logarithm is infeasible (e.g., G can be computed as a subgroup of Z_q^* for a prime q such that $n|(q-1)$), along with six elements $h_1, \ldots, h_6 \neq 1$ of G;
8. output public information `Pub` $= (n, e, g, s, c, g_1, g_2, g_3, G, h_1, \ldots, h_6)$ and private information `Priv` $= (q_1, q_2, d, v_1, v_2, v_3)$.

The bases g_1 and g_2 are used to encode the two credentials of a certificate, c, g_3 for the security of the system, g and s are used in order to achieve multi-show privacy, while $h_1, \ldots, h_6$ are used in order to compute commitments.

User enrollment. In this phase a user asks the organization for a credential certificate with encoded values x_1, x_2 of the two credentials. We assume that $0 \leq x_1, x_2 < e$. The organization returns a good tuple with prefix (x_1, x_2).

Protocol `Enroll`$(x_1, x_2, $`Pub`$) \leftrightarrow$ `IssueCred`$($`Pub`$,$`Priv`$)$

1. the organization verifies the credentials (x_1, x_2) submitted by the user in observance to its policy;
2. the organization randomly chooses x_3 such that $0 \leq x_3 < e$, x_3 is co-prime with e and not multiple of 3, and $x \in Z_n^*$;
3. the organization sets $a \equiv g_1^{x_1} g_2^{x_2} g_3^{x_3} x^e \pmod{n}$, $b \equiv c \pmod{n}$, and computes $v \equiv (a+b)^d \pmod{n}$;

4. the organization sends the good tuple $(x_1, x_2, x_3, x, v, 0, 0)$ to the user;
5. the user verifies that the tuple received is a good tuple;
6. the user and the organization engage in zero-knowledge proofs in which the organization proves that n is the product of two safe primes and that g is a large-order element of Z_n^* (this is done using the protocols of [18]), that g_1, g_2, g_3, x are elements of $\langle g \rangle$ (this is done by proving knowledge of the discrete logarithms of g_1, g_2, g_3 to the base g).
   Such proofs have to be performed only once for each user, independently of the number of credential certificates that he receives.

Showing possession of credentials. In this phase a user proves to a service provider the possession of a good tuple with prefix (x_1, x_2) satisfying the linear Boolean formula Φ encoding the access control policy. More precisely, the following steps are performed by the user and the service provider.

Protocol `ProveCred` $\leftrightarrow$ `VerifyCred`

Common input to user and service provider: the public information `Pub` $= (n, e, g, s, c, g_1, g_2, g_3, G, h_1, h_2, h_3, h_4, h_5, h_6)$ and a linear Boolean formula $\Phi(\cdot, \cdot)$ encoding the access control policy.

Private input to user:

1. a good tuple $(x_1, x_2, x_3, x, v, 0, 0)$ such that $\Phi(x_1, x_2) = 1$.

Instructions for user and service provider:

1. the user sets $a \equiv g_1^{x_1} g_2^{x_2} g_3^{x_3} x^e \pmod{n}$, $b \equiv c \pmod{n}$, $m \equiv a + b \pmod{n}$;
2. the user picks a random y such that $0 \leq y < e$;
3. set $\hat{m} \equiv g^y m \pmod{n}$, $\hat{v} \equiv s^y v \pmod{n}$, $\hat{a} \equiv g^y a \pmod{n}$ and $\hat{b} \equiv g^y b \pmod{n}$; thus we have $\hat{m} - \hat{a} - \hat{b} \equiv 0 \pmod{n}$;
4. the user computes commitments to $\hat{m}$, $\hat{a}$ and $\hat{b}$ by randomly choosing $r_1, r_2, r_3 \in Z_n$ and computing the following values (operations are performed in G):
   4.1. $\mathrm{Com}(\hat{m}) = h_1^{\hat{m}} h_2^{r_1}$;
   4.2. $\mathrm{Com}(\hat{a}) = h_3^{\hat{a}} h_4^{r_2}$;
   4.3. $\mathrm{Com}(\hat{b}) = h_5^{\hat{b}} h_6^{r_3}$;
5. the user sends to the service provider $\mathrm{Com}(\hat{m}), \mathrm{Com}(\hat{a}), \mathrm{Com}(\hat{b})$ and $\hat{a}$;
6. let $\mathrm{Com}(c) = \mathrm{Com}(\hat{m})\mathrm{Com}(\hat{a})\mathrm{Com}(\hat{b})$. Parties engage in proofs of knowledge:
   6.1. The user proves knowledge of a $(Z_n^*, e, g, g_1, g_2, g_3)$-RSA representation (u_g, u_1, u_2, u_3, u) of $\hat{a}$ such that $\Phi(u_1, u_2) = 1$. This is achieved by means of the PoK of Theorem 10 with (y, x_1, x_2, x_3, x) as witness.
   6.2. The user proves knowledge of $(u_1, \ldots, u_6)$, a $(G, h_1, \ldots, h_6)$-DL representation of $\mathrm{Com}(c)$ such that $(u_1 - u_3 - u_5 = 0) \wedge (u_3 = \hat{a})$. This is achieved by means of the PoK of Theorem 9 and with $(\hat{m}, r_1, \hat{a}, r_2, \hat{b}, r_3)$ as witness.

6.3. The user proves knowledge of the $(Z_n^*, 3)$-root of the h_1-part of the (G, h_1, h_2)-DL representation of $\mathrm{Com}(\hat{m})$; that is, the user proves to know the third root $\hat{v}$ of $\hat{m}$. This is achieved by using the PoK of Theorem 7.
6.4. The user proves knowledge of the discrete logarithm with base g of the h_5-part of the (G, h_5, h_6)-DL representation of $\mathrm{Com}(\hat{b})^{c^{-1}}$. Since $\hat{b} \equiv g^y b \equiv cg^y \pmod{n}$, then $y \pmod{n}$ can be used as witness for the PoK of Theorem 8.

4.3 Properties of our implementation

The usability property is obvious. For the proof of the security property we will use the following theorem.

Theorem 13. *The pair of algorithms* (`ProveCred`, `VerifyCred`) *is a PoK for the polynomial time relation of pairs* $((n, e, g, c, g_1, g_2, g_3), (x_1, x_2, x_3, x, v, z, y))$ *for which* $(x_1, x_2, x_3, x, v, z, y)$ *is a good tuple with respect to* $(n, e, g, c, g_1, g_2, g_3)$.

Proof. For the soundness, consider a knowledge extractor $\mathcal{E}$ that has black-box access to algorithm `ProveCred`* that interacts successfully with algorithm `VerifyCred`. Extractor $\mathcal{E}$ engages in the five statistical zero-knowledge proofs of knowledge of step 6 and for each proof of knowledge $\mathcal{E}$ extracts the witness used in the proof. Thus, $\mathcal{E}$ obtains

1. the $(Z_n^*, e, g, g_1, g_2, g_3)$-RSA representation (y, x_1, x_2, x_3, x) of $\hat{a}$ such that $\Phi(x_1, x_2) = 1$;
2. the values $\hat{m}, r_1, \hat{a}, r_2, \hat{b}, r_3$;
3. the third root $v = \hat{v}$ of $\hat{m}$;
4. the discrete logarithm z of $c^{-1}\hat{b}$ to base g;

Using the values extracted, $\mathcal{E}$ can compute a good tuple $(x_1, x_2, x_3, x, v, z, y)$ whose prefix satisfies Φ. □

The theorem above thus shows that if an adversary successfully interacts with algorithm `VerifyCred` then she possesses (with very high probability) a good tuple whose prefix satisfies the access control formula. On the other hand, by our assumption, it is not computationally feasible for a polynomial-time adversary to construct a good tuple without the help of the organization (even if the adversary has access to a polynomial number of good tuples that do not satisfy Φ). Therefore we can conclude that the proposed credential system is secure.

For the multi-show privacy we consider a simulator S that has black-box access to a malicious service provider (that possibly has access to the organization private values). The simulator S receives as input the public information output by the organization and, for all (x_1, x_2) for which $\Phi(x_1, x_2) = 1$, outputs a view that is statistically close to the view of the interaction of a legitimate user that possesses credentials (x_1, x_2) with the service provider. Essentially the same simulator can be used to simulate any number of interactions with a service provider each interaction with a (possibly) different formula Φ.

The simulator starts by randomly choosing y such that $0 \leq y < e$, x_3 such that $0 \leq x_3 < e$, x_3 is co-prime with e and 3, $x \in Z_n^*$ and x_1', x_2' such that $0 \leq x_1', x_2' < e$, and $\Phi(x_1', x_2') = 1$[3]. Then the simulator computes the values $\hat{a} \equiv g^y g_1^{x_1} g_2^{x_2} g_3^{x_3} x^e \pmod{n}$, $\hat{b} \equiv cg^y \pmod{n}$ and $\hat{m} \equiv \hat{a} + \hat{b} \pmod{n}$ and commitments by randomly choosing $r_1, r_2, r_3 \in Z_n^*$ and setting $\mathrm{Com}(\hat{m}) = {h_1}^{\hat{m}} {h_2}^{r_1}, \mathrm{Com}(\hat{a}) = {h_3}^{\hat{a}} {h_4}^{r_2}, \mathrm{Com}(\hat{b}) = {h_5}^{\hat{b}} {h_6}^{r_3}$.

The simulator sends $\mathrm{Com}(\hat{m}), \mathrm{Com}(\hat{a}), \mathrm{Com}(\hat{b})$ and $\hat{a}$ to the verifier. Observe that the values computed by the simulator have the same distribution of the values sent by the prover (i.e., a legitimate user) in a real execution of the protocol.

For the proofs of knowledge of 6.1 to 6.4, we observe that the simulator has the witnesses for successfully performing proofs 6.1, 6.2, 6.4. However, since the proofs are statistical zero-knowledge (and hence statistical witness indistinguishable) the output of the simulator for this proofs is statistical indistinguishable from a proof performed by a legitimate user with credentials (x_1, x_2). Instead for proof 6.3, the simulator resorts to the simulator of the statistical zero-knowledge proof systems and thus again the output produced by the simulator is statistically close to the view of the verifier in a real execution.

We thus have the following theorem

Theorem 14. *The tuple of algorithms (*`SetUp`*,* `Enroll`*,* `IssueCred`*,* `ProveCred`*,* `VerifyCred`*) is a usable, secure and multi-show private credential system.*

4.4 Non-transferability

When a user shares his credential certificate with other users he has to give them the corresponding good tuple. In order to discourage such a sharing it is possible to add some credentials that typically are not shared by their owners. For example another base $g^* \in \langle g \rangle$ could be considered in order to encode a credit card number or a private key in a credential certificate. Using this mechanism each user that tries to use such a credential certificate needs to know the owner's credit card number or private key in order to successfully perform protocol `ProveCred` while interacting with the service provider running algorithm `VerifyCred`.

More precisely, a credential certificate can be lent by a user only if all credentials that are in the credential certificate are also released thus making the lending of credential certificates inconvenient.

4.5 Efficiency improvement

The step 6.4 of protocol `ProveCred` $\leftrightarrow$ `VerifyCred` is the most expensive one of our protocol. We briefly suggest two possible improvements, details are omitted from this extended abstract.

[3] For the sake of ease of exposition we are not addressing the problem of finding a tuple (x_1', x_2') that satisfies $\Phi(\cdot, \cdot)$. We will give details in the full version of the paper.

1. Step 6.4 could be replaced by a proof of knowledge of a third root of a part of a representation, in this case the constant c has to be replaced by a value chosen by O that gives to the user the corresponding third root. In this case the complexity assumption need to be strengthened. One possible workaround is to add another (efficient) proof of knowledge on the value $\hat{a}$, for instance by setting $a = cg_0^w$ for a new base g_0, then the user should also prove knowledge of a square root of w and knowledge of the third roots of z_1, z_2 such that $\hat{a} = c_1 z_1 + c_2 z_2$ where c_1, c_2 are new constants. This can be achieved by considering a proof of knowledge of the square root of a committed value [16], since $\hat{a} = cg^y g_0^w$.
2. By using the subgroup of quadratic residues of Z_n^*, it is possible to extend both the system set-up and user enrollment phases by integrating the construction of [14] (obviously, the ability of the group manager of opening signatures must be removed). In such a case, each user obtains a triplet A^*, e^*, x^* such that $A^{*e^*} = g^{*x} g'$ and it is hard (strong RSA assumption) to generate a new valid triplet. By linking such a triplet to the value $g_1^{x_1} g_2^{x_2} g_3^{x_3} x^e$, then the user should prove knowledge of both a valid triplet and an RSA representation.

4.6 Efficiency of our scheme

In our proposed scheme, a user only needs one certificate to access any (polynomial) number of times any (polynomial) number of different services each with her access control policy encoded by a linear boolean formula. Previous schemes either required one certificate for each access and each service (e.g., [4, 5]) or one certificate for each type of service (e.g., [9, 10]). In Table 1 we show a general case in which a user is described by β attributes and wants to unlinkably access α different services each for δ times. Each service in general has a different access control policy (encoded via a linear boolean formula) that is based on the β credentials. Each row contains data for an anonymous credential system. The

Table 1. Performance comparison.

Scheme	# Certificates	# Workload
This paper	1	$O(\beta)$
[10]	α	$O(\beta)$
[9]	α	$O(\beta)$
[4]	$\alpha\delta$	$O(\beta)$

first column specifies the schemes. The second column specifies the number of certificates required to unlinkably access services. The last column specifies the number of modular exponentiations to be performed for each access.

The scheme of Brands (see [4, 5]) requires $\alpha\delta$ certificates since Brands' certificates are one-show and thus accesses that use the same certificate are linkable. Accessing a service require $O(\beta)$ modular exponentiations and these are due to a proof of knowledge of a discrete logarithm representation. The constant hidden by the O notation is small.

The schemes by Camenisch and Lysyanskaya [9] and the one by Verheul [10] are multi-show and thus the same certificate can be used more than once for gaining access to the same service. However, each service (with its own access control policy described by different formulae) requires a different certificate. This is due to the fact that the schemes of [9] and [10] do not allow to prove general statements on the credentials encoded by a certificate and thus the certificates themselves have to encode the access control policy. This has an important consequence. In the system of Brands and in our scheme the certificates can be obtained independently of the access control policies for which they will be used. Instead in the system of [9] and in the one of Verheul [10] the certificates depend on the access control formula. Therefore, should the access control policy of the service change, a new certificate suited for the new formula should be requested by the user. The number of modular exponentiations for accessing a service is linear in the number of credentials for both the constructions; however the system of [10] is efficient and the constant hidden in the O notation is small.

The scheme presented in this paper allows one to use the same certificate for accessing α services each δ times while preserving anonymity and unlinkability. In particular, this means that services can change their access control policies without requiring the users to get new certificates from the issuing organizations. This property is not obtained for free. Indeed the number of modular exponentiations for each access to a service is still linear in the number β of certificates and it is the sum of two factors: a factor (proportional to β with a small hidden constant) due to a proof of knowledge of an RSA representation that is very efficient and a constant factor due to one proof of knowledge of a double discrete log. The suggestions discussed in Section 4.5 improve such a constant factor. Notice that when the number of encoded credentials is large such a constant factor becomes negligible and the overhead inferred by our system decreases.

5 Conclusion

In this paper we have presented an efficient and usable non-transferable multi-show anonymous credential system. We have shown the advantages of our system with respect to the recent results of [9, 10, 4, 5] since it allows a user prove efficiently general statements on the credentials encoded in his certificate and, at the same time, it is multi-show and non-transferable.

References

1. Housley, R., Polk, W., Ford, W., Solo, D.: Internet X509 Public Key Infrastructure: Certificate and Certificate Revocation List (CRL) Profile. Network Working Group, RFC 3280 (2002)
2. De Santis, A., Di Crescenzo, G., Persiano, G., Yung, M.: On Monotone Formula Closure of SZK. In: Proceedings of the 35th Symposium on Foundations of Computer Science, (FOCS '94). (1994) 454–465

3. De Santis, A., Di Crescenzo, G., Persiano, P.: Communication-Efficient Anonymous Group Identification. In: Procedings of the 5th ACM Conference on Computer and Communications Security, ACM (1998) 73–82
4. Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates; Building in Privacy. MIT Press (2000)
5. Brands, S.: Rapid Demonstration of Linear Relations Connected by Boolean Operators. In Fumy, W., ed.: Advances in Cryptology – Eurocrypt '97. Volume 1223 of Lecture Notes in Computer Science., Springer-Verlag (1997) 318–333
6. Chaum, D., Evertse, J.: A Secure and Privacy-Protecting Protocol for Transmitting Personal Information between organizations. In: Advances in Cryptology – Crypto '86. Volume 263 of Lecture Notes in Computer Science., Springer-Verlag (1987) 118–167
7. Lysyanskaya, A., Rivest, R., Sahai, A., Wolf, S.: Pseudonym Systems. In: Proceedings of Selected Areas in Cryptography. Volume 1758 of Lecture Notes in Computer Science., Springer-Verlag (1999) 184-199
8. Chen, L.: Access with Pseudonyms. In: Cryptography: Policy and Algorithms. Volume 1029 of Lecture Notes in Computer Science., Springer-Verlag (1995) 232–243
9. Camenisch, J., Lysyanskaya, A.: An Efficient Non-Transferable Anonymous Multi-Show Credential System with Optional Anonymity Revocation. In: Advances in Cryptology – Eurocrypt '01. Volume 2656 of Lecture Notes in Computer Science., Springer-Verlag (2001) 93–118
10. Verheul, E.: Self-Blindable Credential Certificates from the Weil Pairing. In Boyd, C., ed.: Advances in Cryptology – Asiacrypt '01. Volume 2248 of Lecture Notes in Computer Science., Springer-Verlag (2001) 533–551
11. Persiano, P., Visconti, I.: An Anonymous Credential System and a Privacy-Aware PKI. In: Proceedings of the 8th Australasian Conference on Information Security and Privacy, (ACISP '03). Volume 2727 of Lecture Notes in Computer Science., Springer-Verlag (2003) 27–38
12. Ateniese, G., Tsudik, G.: Some Open Issues and New Directions in Group Signatures. In Franklin, M., ed.: Financial Cryptography. Volume 1648 of Lecture Notes in Computer Science., Springer-Verlag (1999) 196–211
13. Camenisch, J., Stadler, M.: Efficient Group Signature Schemes for Large Groups. In: Advances in Cryptology – Crypto 97. Volume 1294 of Lecture Notes in Computer Science., Springer-Verlag (1997) 410–424
14. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A Practical and Provably Secure Coalition-Resistant Group Signature Scheme. In: Advances in Cryptology - Crypto '00. Volume 1880 of Lecture Notes in Computer Science., Springer-Verlag (2000) 255–270
15. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
16. Ateniese, G., De Medeiros, B.: Efficient Group Signatures without Trapdoors. In: Advances in Cryptology – Asiacrypt '03. Lecture Notes in Computer Science, Springer-Verlag (2003) 246-268
17. Bresson, E., Stern, J.: Proofs of Knowledge for Non-Monotone Discrete-Log Formulae and Applications. In: Proceedings of International Security Conference (ISC '02). Volume 2433 of Lecture Notes in Computer Science., Springer-Verlag (2002) 272-288
18. Camenisch, J., Michels, M.: Proving in Zero-Knowledge that a Number is the Product of Two Safe Primes. In: Advances in Cryptology – Eurocrypt '99. Volume 1592 of Lecture Notes in Computer Science., Springer-Verlag (1999) 106–121