Anonymous Credential Schemes with Encrypted Attributes

Jorge Guajardo[1], Bart Mennink[2] and Berry Schoenmakers[3]

[1] Information and System Security Group
Philips Research, Eindhoven, The Netherlands
jorge.guajardo@philips.com
[2] Dept. Electrical Engineering, ESAT/COSIC and IBBT
Katholieke Universiteit Leuven, Belgium
bart.mennink@esat.kuleuven.be
[3] Dept. of Mathematics and Computer Science
Technische Universiteit Eindhoven, The Netherlands
berry@win.tue.nl

Abstract. In anonymous credential schemes, users obtain credentials on certain attributes from an issuer, and later show these credentials to a relying party anonymously and without fully disclosing the attributes. In this paper, we introduce the notion of (anonymous) credential schemes with *encrypted* attributes, in which issuers certify credentials on *encrypted* attributes to users. These schemes allow for the possibility that none of the involved parties, including the user, learns the values of the attributes. In fact, we will treat several variations differing in which parties see which attributes in the clear. We present efficient constructions of these new credential schemes, starting from a credential scheme by Brands, and we show that the security of Brands' original scheme is retained. Finally, we sketch several interesting applications of these novel credential schemes.

1 Introduction

Anonymous credential schemes, credential schemes for short, allow users to obtain credentials on particular attributes from issuers certifying that the users comply with particular conditions. These credentials can subsequently be shown to a relying party (or, verifier) in order to gain access to a service. Credential schemes were introduced by Chaum [14,15], and many efficient constructions are known [16,20,5,7,22,24,10,11,12,2], as well as several variations and extensions [17,3,4,1,9,8] (incl. anonymous cash). Credential schemes can be seen as a refined form of blind signatures, inheriting the unforgeability and unlinkability properties, but adding an extra level of privacy by allowing users to control which private information is disclosed when showing a credential. For example, irrelevant attributes need not be disclosed at all [7,1], or a zero-knowledge proof that the attributes satisfy some given constraints may suffice [6].

In general, a credential scheme consists of a key generation algorithm, an issuance protocol and a verification (or, showing) protocol. The key generation algorithm supplies an issuer with a key pair, which is used to issue credentials on lists of attributes $x = (x_1, \ldots, x_l)$ to users. Such a credential is of the form $(p, s, \sigma(p))$, where p is the public part authenticated by $\sigma(p)$ and s is a secret part (including x) corresponding to p. To show a credential to a verifier, a user sends the public parts p and $\sigma(p)$, and proves knowledge of the secret part s, possibly revealing some of the attributes in x. Common to all credential schemes in the literature, however, is the property that the focus is on *authentication via credentials* to protect access to services. Naturally, the user knows the attributes in these applications. As we will see below, however, in many applications the user is not allowed to know (some of) the attributes or does not want to know these. In this paper, we therefore introduce credential schemes with encrypted attributes, or *encrypted credential schemes* for short, in which credentials may contain encrypted attributes. In the most general case, all parties involved (issuer, user, and verifier) might have access to the attributes in encrypted form only. We note that these encrypted credential schemes still offer the authentication property, as described above.

Applications. Basically, there are two types of applications of encrypted credential schemes: one can think of scenarios where the user *is not allowed* to learn the attributes, as well as cases where the user *does not want* to learn the attributes. As a simple example of the former case, consider the use of confidential letters of recommendation: if a person wants to apply for a job or wants to enter into a graduate program at a university, he can request a letter of recommendation from his former supervisor, which he in turn shows to the potential employer. Since the holder of this letter is not allowed to learn the supervisor's opinions, the letter can be implemented as an encrypted credential. Note that this allows one to apply for jobs anonymously (at first) and without the supervisor knowing of the job applications. For the case where the user does not want to learn the attributes, one can think of people who do not want to learn about (genetic) diseases they suffer, but still need credentials on these data for various purposes.

More generally, encrypted credential schemes provide the missing link between anonymous credentials (where credentials are merely issued on data), and secure multiparty computation (MPC) based on threshold homomorphic cryptosystems (where one can usually trace which outputs are used in subsequent computations). For instance, one can consider a first MPC where the total wealth of a party is computed based on his salary, registered possessions (such as real estate), etc. This computation may involve various database lookups, and in particular, the identity of the party may be known. The computed result representing the total wealth, however, should remain hidden, and will only be given in encrypted form to the credential issuer. A second MPC could then be the millionaires protocol, or a more elaborate computation, on the wealths of a number of parties, where all involved parties should remain anonymous. Here, one needs the encrypted data to be accompanied with an anonymous credential. The parties performing the secure computations will remain oblivious to the links between the inputs and outputs of the secure computations.

Challenges and Technical Issues. Observe that one cannot solve this problem by naively issuing 'standard' credentials on the ciphertexts (as being the attributes): while the message to be signed can be re-blinded by the user, the attributes themselves *cannot*. As a consequence, the ciphertexts cannot be re-randomized in this solution to the problem. More generally, common credential schemes rely in an essential way on the fact that (at least) the user has access to the attributes in the clear. Instead, to handle encrypted attributes some major changes are needed. Below we highlight some of the issues.

- If users do not know the attributes of a credential, they cannot render zero-knowledge proofs for these attributes as part of the verification protocol. To resolve this issue, verification of a credential with encrypted attributes will involve some type of plaintext equality test, ensuring minimal disclosure of the attributes. The secret key used for verification will in general be distributed among multiple parties;
- Given the use of some type of plaintext equality test in the verification protocol, it must be prevented that a malicious user abuses the verification protocol to gain partial information on the encrypted attributes (which in general need to remain hidden from the user). This contrasts with the verification protocol in common credential schemes, where the verifier only generates a random challenge and has no private inputs;
- To achieve unlinkability for the credential scheme, honest users should be able to blind the encrypted attributes as provided by the issuer, basically by performing random re-encryptions of these attributes. Malicious users, however, should not be able to abuse this mechanism by replacing a particular encrypted attribute with a target encryption. Based on the success or failure in a run of the verification protocol, a malicious user would then be able to find out if the encryption of the attribute and the target encryption contain the same plaintext or not.

Our Contributions. We introduce and define the notion of *encrypted credential schemes* as a new concept in the area of anonymous credentials (Sect. 3). Our definition captures the case of an issuer certifying *encrypted* attributes, such that none of the involved participants learns the encryptions. For a variation where the issuer knows the attributes (see the above-mentioned applications), we introduce a concrete construction of an efficient scheme (Sect. 4), but we notice that the scheme can be easily adjusted to a more general case where the issuer does not learn (some of) the attributes (Sect. 6). The security of the scheme is analyzed in Sect. 5. The construction of our scheme is based on an efficient and well-established credential scheme by Brands [7]. By combining the newly introduced schemes with Brands' original scheme, we also obtain schemes in which users learn *some* of the attributes in the clear, and *some* in encrypted form (Sect. 6). Since Brands' schemes are providing single-use credentials, our schemes do so as well[4]. We leave it as an open problem to construct multi-use

[4] As observed in [8], Brands' schemes allow for efficient issuance of multiple credentials on the same attribute list. The same remark applies to our schemes.

credential schemes with encrypted attributes (e.g., starting from [11,12]), or to show the impossibility of such construction.

2 Preliminaries

Below, we introduce some well-known cryptographic primitives along with some relevant notation. In particular, we give some background on the credential scheme by Brands on which our constructions are based. Throughout, we use $x \in_R V$ to denote that x is drawn uniformly at random from V, and $\mathcal{H}$ to denote a cryptographic hash function (viewed as a random oracle) with range equal to $\mathbb{Z}_q$. Here, q is a prime of bit-length k, for a security parameter k. Furthermore, we let $\langle g \rangle$ denote a cyclic group of prime order q, representing a suitable discrete log setting.

ElGamal Cryptosystem. Our scheme uses the additively homomorphic ElGamal cryptosystem. For cyclic group $\langle g \rangle$, the secret key is a $\lambda \in_R \mathbb{Z}_q$ and the corresponding public key is the group element $f = g^\lambda$. A message $x \in \mathbb{Z}_q$ is encrypted by taking an $r \in_R \mathbb{Z}_q$ and computing $c = (g^r, g^x f^r) =: [\![g^x]\!]$. For efficient decryption, x must be limited to a sufficiently small set (but this is no limitation for our setting). The decryption function is denoted by D. The homomorphic properties ensure that an encryption can be re-blinded by multiplying it with a random zero-encryption $[\![g^0]\!]$.

Σ-Protocols. Informally, a zero-knowledge proof of knowledge is a two-party protocol for a prover to convince a verifier that he knows something, without leaking any information other than the value of the assertion that is being proved. More specifically, for a relation $R = \{(x; w)\}$ and for an x, common input for the prover and verifier, the prover proves in zero-knowledge that he knows a value w (the witness) such that $(x; w) \in R$. We use the notion of Σ-protocols, cf. Cramer et al. [19]. A Σ-protocol consists of a conversation (a, c, r), where the prover sends a commitment a, the verifier returns a random challenge c and the prover sends a response r. Afterwards, the verifier either accepts or rejects. A Σ-protocol needs to satisfy three properties: *completeness*, *special soundness* and *special honest-verifier zero-knowledge.*

Brands' Credential Scheme. We will consider Brands' credential scheme [7] based on the blind Chaum-Pedersen signature scheme [18][5]. Given cyclic group $\langle g \rangle$, the issuer's public key consists of the group elements $h_0, g_1, \ldots, g_l \in_R \langle g \rangle$, and a credential on attribute list $(x_i)_{i=1}^l$ is a tuple $(h', (x_i)_{i=1}^l, \alpha, \sigma(h'))$ satisfying

$$\sigma(h') \text{ is a signature on } h', \text{ and } (g_1^{x_1} \cdots g_l^{x_l} h_0)^\alpha = h'. \qquad (1)$$

[5] Brands also introduced a more efficient DL-based scheme, but this scheme does not offer the possibility for the issuer to issue a credential without knowing the attributes in the clear, while this is clearly a requirement in encrypted credential schemes.

Upon verification of a credential, the user sends the public part of the credential, $(h', \sigma(h'))$, to the verifier, and executes a Σ-protocol to prove knowledge of $((x_i)_{i=1}^{l}, \alpha)$ satisfying $(g_1^{x_1} \cdots g_l^{x_l} h_0)^\alpha = h'$. A summary of Brands' scheme is given in App. A.

3 Definition of Encrypted Credential Schemes

In this section, the notion of encrypted credential schemes will be introduced more precisely. As mentioned above, an encrypted credential scheme considers the case of an issuer certifying *encrypted* attributes to users, such that none of the involved participants learns the attributes. The basic ingredients for this type of schemes are three protocols: a key generation protocol for generating public and secret keys, and protocols for issuance and verification of encrypted credentials. Informally stated, the security and privacy of these schemes comprise the following. Security roughly means (1) that credentials are unforgeable, meaning that it is hard for a user to convince the verifier with a forged credential and (2) that no unauthorized participant learns the encrypted attributes. Privacy means that anonymity of the users is guaranteed and executions of the verification protocol cannot be linked. More precisely, we propose the following definition of encrypted credential schemes, taking into account the technical issues mentioned in Sect. 1.

Definition 1. *An* encrypted credential scheme *consists of the following protocols, where $\mathcal{I}$ denotes an issuer, $\mathcal{U}$ denotes a user, and $\mathcal{V}$ denotes a verifier:*

- *A* key generation protocol *for $\mathcal{I}$ and $\mathcal{V}$, that on input of security parameter k outputs public/secret keys $(pk, sk_{\mathcal{I}}, sk_{\mathcal{V}})$, where pk includes the system parameters. It also includes a key pair for an encryption scheme, of which the secret key is owned by $\mathcal{V}$. We write $(pk, sk_{\mathcal{I}}, sk_{\mathcal{V}}) \leftarrow \mathsf{keygen}_{\mathcal{I},\mathcal{V}}(k)$;*
- *An* issuance protocol *for $\mathcal{I}$ and $\mathcal{U}$, that on input of pk and a list of encrypted attributes C, together with $\mathcal{I}$'s secret key, outputs a credential $(p, s, \sigma(p))$ for the user. This credential satisfies that $\sigma(p)$ is a signature on p, that p is a public key part for which s is a secret key, and that p contains re-encryptions of the encryptions in C. The protocol is denoted by $(p, s, \sigma(p)) \leftarrow \mathsf{issue}_{\mathcal{I}(sk_{\mathcal{I}});\mathcal{U}}(pk, C)$;*
- *A* verification protocol *for $\mathcal{U}$ and $\mathcal{V}$, that on input of pk, $\mathcal{U}$'s input $(p, s, \sigma(p))$ and $\mathcal{V}$'s secret key outputs a bit, representing either acceptance or rejection. We write $\mathsf{verify}_{\mathcal{U}(p,s,\sigma(p));\mathcal{V}(sk_{\mathcal{V}})}(pk)$ to denote a run of the protocol, which outputs a bit.*

These protocols satisfy the following properties for any $(pk, sk_{\mathcal{I}}, sk_{\mathcal{V}})$ resulting from an execution of the key generation protocol:

- Completeness. *For any honest $\mathcal{I}$, $\mathcal{U}$ and $\mathcal{V}$, the credential obtained by $\mathcal{U}$ in the execution of the issuance protocol, will be accepted in the verification protocol;*

- Security. *The credentials are unforgeable and no unauthorized party learns the encrypted attributes;*
- Privacy. *The scheme offers unlinkability, and anonymity of the users is guaranteed.*

In practice, the verifier's secret key can be shared among multiple verifiers using threshold cryptography, such that the user can execute the verification protocol with any qualified set of verifiers [25]. The definition is formulated in a general way, but variations are possible as well. The definition can for instance be adjusted to the case where $\mathcal{I}$ learns the attributes, but $\mathcal{U}$ and $\mathcal{V}$ do not. Note that this is precisely the case in the specific applications mentioned in Sect. 1. Furthermore, we notice that Def. 1 does not restrict credentials to be single-use or multi-use. The remainder of the paper, however, concentrates on single-use credentials. In particular, Def. 2 below states completeness, security and privacy properties more concretely for single-use credentials, following Brands' definition of secure credential schemes [7]. Throughout, a (potentially) malicious participant is indicated by an apostrophe, as in $\mathcal{U}'$. A participant is called semi-honest in case he follows the protocol but tries to obtain as much information as possible. For simplicity, we consider semi-honest verifiers only. This can be guaranteed by implementing $\mathcal{V}$ as a set of parties using threshold cryptography (see also Sect. 6).

Definition 2. *A* key generation, issuance *and* verification *protocol involving parties $\mathcal{I}$,$\mathcal{U}$ and $\mathcal{V}$ constitute a secure encrypted credential scheme (cf. Def. 1) if the following properties are satisfied for any $(pk, sk_{\mathcal{I}}, sk_{\mathcal{V}})$ resulting from an execution of the key generation protocol:*

- Completeness. *For any attribute list C and honest $\mathcal{I}$,$\mathcal{U}$ and $\mathcal{V}$, the issuance protocol on input of C results in a valid credential for $\mathcal{U}$. More formally, for any C we have*

$$\Pr\Big(\mathsf{verify}_{\mathcal{U}(p,s,\sigma(p));\mathcal{V}(sk_{\mathcal{V}})}(pk) = 1 \,\Big|\, (pk, sk_{\mathcal{I}}, sk_{\mathcal{V}}) \leftarrow \mathsf{keygen}_{\mathcal{I};\mathcal{V}}(k); \\ (p, s, \sigma(p)) \leftarrow \mathsf{issue}_{\mathcal{I}(sk_{\mathcal{I}});\mathcal{U}}(pk, C)\Big) = 1;$$

- User privacy. *For any two issued credentials, a malicious $\mathcal{I}'$ cannot distinguish between the public key parts of these credentials. More formally, there exists a negligible $\nu(k)$, such that for any C_0, C_1 we have*

$$\Pr\Big(\mathcal{I}'(pk, sk_{\mathcal{I}'}, (p, \sigma(p))_b, (p, \sigma(p))_{1-b}, \mathsf{view}_0, \mathsf{view}_1) = b \,\Big| \\ (pk, sk_{\mathcal{I}'}, sk_{\mathcal{V}}) \leftarrow \mathsf{keygen}_{\mathcal{I}';\mathcal{V}}(k);\ b \in_R \{0, 1\}; \\ (p, s, \sigma(p))_j \leftarrow \mathsf{issue}_{\mathcal{I}'(sk_{\mathcal{I}'});\mathcal{U}}(pk, C_j) \text{ for } j = 0, 1\Big) < \frac{1}{2} + \nu(k),$$

where view_j *denotes $\mathcal{I}'$'s view on the j-th issuing execution ($j = 0, 1$), i.e. all values $\mathcal{I}'$ sees during the execution;*

– One-more unforgeability. *Suppose that for any $K \geq 0$, malicious $\mathcal{U}'$ can perform K arbitrarily interleaved credential queries on adaptively chosen attribute lists C_j $(j = 1, \ldots, K)$. Then, the probability that $\mathcal{U}'$ outputs $K+1$ distinct credentials is negligible in k. More formally, there exists a negligible $\nu(k)$, such that for any $K \geq 0$ we have*

$$\Pr\Big(\forall_{i=1}^{K+1}\big[\mathsf{verify}_{\mathcal{U}'((p,s,\sigma(p))_i);\mathcal{V}(sk_{\mathcal{V}})}(pk) = 1\big] \;\Big|\; (pk, sk_{\mathcal{I}}, sk_{\mathcal{V}}) \leftarrow \mathsf{keygen}_{\mathcal{I};\mathcal{V}}(k); \\ \{(p, s, \sigma(p))_i\}_{i=1}^{K+1} \leftarrow \mathcal{U}'^{\,\mathsf{issue}_{\mathcal{I}(sk_{\mathcal{I}});\mathcal{U}'}(pk,\cdot)}\Big) < \nu(k),$$

where $\mathcal{U}'$ queries its oracle K times;

– Blinding-invariance unforgeability. *Suppose that for any $K \geq 0$, malicious $\mathcal{U}'$ can perform K arbitrarily interleaved credential queries on adaptively chosen attribute lists C_j $(j = 1, \ldots, K)$, and that $\mathcal{U}'$ outputs L credentials $((p, s, \sigma(p))_i)_{i=1}^{L}$ for some $L \leq K$. Then, for any of the attribute lists in these L credentials, the number of credentials on this list does not exceed the number of times a credential has been issued on this list. More formally, there exists a negligible $\nu(k)$, such that for any $K \geq L \geq 0$ we have*

$$\Pr\Big(\forall_{i=1}^{L}\big[\mathsf{verify}_{\mathcal{U}'((p,s,\sigma(p))_i);\mathcal{V}(sk_{\mathcal{V}})}(pk) = 1\big] \;\wedge\; R \not\subseteq S \;\Big| \\ (pk, sk_{\mathcal{I}}, sk_{\mathcal{V}}) \leftarrow \mathsf{keygen}_{\mathcal{I};\mathcal{V}}(k); \\ R := \{D(\mathsf{inv}(p_i))\}_{i=1}^{L} \text{ and } S := \{D(C_j)\}_{j=1}^{K} \text{ multisets}; \\ \big(\{(p, s, \sigma(p))_i\}_{i=1}^{L}, Q\big) \leftarrow \mathcal{U}'^{\,\mathsf{issue}_{\mathcal{I}(sk_{\mathcal{I}});\mathcal{U}'}(pk,\cdot)}\Big) < \nu(k),$$

where $\mathcal{U}'$ queries its oracle K times, and $Q = \{C_j\}_{j=1}^{K}$ are the corresponding attribute lists. Here, inv *is some non-constant function that, on input of the public key part of a credential, outputs the corresponding list of encrypted attributes (cf. Def. 1), and R and S denote multisets of plaintext (decrypted) attribute lists;*

– Secure verification. *For any credential $(p, s, \sigma(p))$, the verification protocol is a secure two-party protocol for proving knowledge of s such that $(p, s, \sigma(p))$ is a valid credential, where $\mathcal{U}$ sent $(p, \sigma(p))$ to $\mathcal{V}$.*

Additionally, no unauthorized party learns the encrypted attributes.

We note that these properties indeed cover the privacy and security requirements informally introduced in the beginning of this section. In particular, the two unforgeability statements encompass any possible forgery: a forger can either construct more credentials than he is issued on, or less but on different attributes[6]. The property that no unauthorized party learns the attributes usually follows directly from the other properties (cf. App. B). In particular, the encrypted attributes do not leak during the verification execution, as the verifier is semi-honest and this protocol is a secure two-party protocol.

[6] Even though the idea of encrypted credential schemes is that the user will not learn the attributes in the clear, the unforgeability requirements are defined so as to cover security against malicious users adaptively choosing the attributes. This is done in order to achieve similar security results as in the credential scheme by Brands.

4 An Encrypted One-Show Credential Scheme

In this section, we construct an encrypted credential scheme. The scheme presented below is for the case where the issuer knows the attributes in the clear, while the user and verifier[7] do not learn these. This is one of the most interesting variations (see the applications in Sect. 1). In Sect. 6 we will consider the extension to the case where the issuer does not learn the attributes either. For simplicity, it is assumed that the attributes are binary, i.e., $x_1^*, \ldots, x_{l-1}^* \in \{0,1\}$. The scheme is based on a credential scheme by Brands [7], and in particular our scheme can be combined with Brands' scheme, for instance such that the user learns x_1^*, x_2^*, but does *not* learn $x_3^*, \ldots, x_{l-1}^*$. See also Sect. 6.

The encrypted credential scheme will be introduced from a constructive point of view: at first the ideas of the protocols are described with respect to Brands' scheme, and then the mathematical descriptions of the protocols are given. A general remark is that the issuer will actually certify attributes $x_i = x_i^* + \phi_i$ for some $\phi_i \in_R \mathbb{Z}_q$ unknown to the user (for $i = 1, \ldots, l-1$), and an additional $x_l \in_R \mathbb{Z}_q$. This adjustment turns out to be important for solving the third technical issue of Sect. 1: without this modification, the verification of a credential with an encrypted attribute replaced with a target or faked encryption would succeed with significant probability for attributes from a limited range (e.g. binary attributes). By artificially extending their range to $\mathbb{Z}_q$, such attack succeeds with negligible probability only. Interestingly, it turns out that an issuer can use the same tuple $(\phi_i)_{i=1}^{l-1}$ for all executions of the issuance protocol.

4.1 Key Generation

Essentially, we combine the key generation algorithms of Brands' scheme and of the ElGamal cryptosystem. Additionally, the values $(\phi_i)_{i=1}^{l-1}$ are needed as well, as mentioned above. However, in our scheme the key generation is actually a protocol between issuer and verifier, because the verifier needs secret data as well. The verifier will use its secret data in a plaintext equality test, cf. Eq. (2b), and these values are not needed by the issuer for issuing credentials.

The key generation protocol can now be described as follows. Given a security parameter k, system parameters (q, g), with prime $q > 2^k$, are generated first. Then, public key $(h_0, f, \hat{f}, (g_i)_{i=1}^{l}, (f_i)_{i=1}^{l-1})$ is generated jointly, corresponding to the issuers secret key $x_0, (\phi_i)_{i=1}^{l-1} \in_R \mathbb{Z}_q$ and the verifiers secret key $\lambda, (y_i)_{i=1}^{l} \in_R \mathbb{Z}_q$ satisfying

$$h_0 = g^{x_0}, \quad f = g^{\lambda}, \quad \hat{f} = f^{x_0} = h_0^{\lambda}, \quad \forall_{i=1}^{l} : g_i = g^{y_i}, \quad \forall_{i=1}^{l-1} : f_i = g^{\phi_i}.$$

4.2 Credential Issuance

As in Brands' issuance protocol, the user's attributes are signed indirectly via the group element $h = g_1^{x_1} \cdots g_l^{x_l} h_0 \neq 1$. The attributes are provided to the user in

[7] Recall that we consider semi-honest verifiers only.

encrypted form only, and are blinded by the users by random re-encryption. To indeed restrict the users to random re-encryptions of the encrypted attributes, the issuer uses the values $(\phi_i)_{i=1}^{l-1}$ when forming h and the encryptions c_i of the attributes, as well as the values $z_i = c_i^{x_0}$ and $e_i = c_i^w$. The protocol for issuing a credential on $(x_i^*)_{i=1}^{l-1}$ is given in Fig. 1. It results in a credential for $\mathcal{U}$, which consists of a tuple $(h', (c_i')_{i=1}^l, \alpha, z', (z_i')_{i=1}^l, c', r')$ satisfying

$$c' = \mathcal{H}([c_i', z_i', (c_i')^{r'}(z_i')^{-c'}]_{i=1}^l; h', z', g^{r'}h_0^{-c'}, (h')^{r'}(z')^{-c'}), \tag{2a}$$

$$\text{and } (D((c_1')^{y_1}\cdots(c_l')^{y_l})h_0)^\alpha = h' \neq 1, \tag{2b}$$

where $c_i' = [\![g^{x_i^*}f_i]\!]$ for $i = 1, \ldots, l-1$, and $c_l' = [\![g^{x_l}]\!]$ for $x_l \in_R \mathbb{Z}_q$.

Note that these credentials mainly differ from Brands' credentials in the second part (2b). By defining $x_i := x_i^* + \phi_i$ for $i = 1, \ldots, l-1$, we have $(c_i')^{y_i} = [\![g^{x_i}]\!]^{y_i} = [\![g_i^{x_i}]\!]$ for all i. Consequently, (2b) simplifies to $(g_1^{x_1}\cdots g_l^{x_l}h_0)^\alpha = h'$, which is the same equation as in Brands' credential scheme, cf. (1). The crucial difference is that the verification of (2b) is done through a plaintext equality test and requires access to a secret key.

4.3 Credential Verification

For verification of a credential $(h', (c_i')_{i=1}^l, \alpha, z', (z_i')_{i=1}^l, c', r')$, the user sends the public part (all values except for α) to the verifier, and proves knowledge of α such that (2) holds. This protocol is given in Fig. 2. Upon successful verification, the verifier will extract the encrypted attributes $[\![g^{x_i^*}]\!]$ by computing $c_i'[\![f_i]\!]^{-1}$ (for $i = 1, \ldots, l-1$).

The verification protocol can be viewed as a proof of knowledge for relation $\{(h', (c_i')_{i=1}^l; \alpha) \mid (h')^{\alpha^{-1}} = D((c_1')^{y_1}\cdots(c_l')^{y_l})h_0 \wedge \alpha \neq 0\}$, except that the verifier uses a secret input as well for the evaluation of a plaintext equality test. For this reason, the protocol is not a Σ-protocol, and an explicit fourth round has been added to inform the user whether verification succeeded.

5 Security Analysis

Using Def. 2, we analyze the security of the above encrypted credential scheme.

Theorem 1. *The protocols introduced in Sect. 4 constitute a secure encrypted credential scheme cf. Def. 2.*

The proof is rather technical, and is included in App. B. It is based on Ass. 2. Intuitively, this assumption states that if a malicious user can succeed showing a credential, then (with overwhelming probability) he has been issued a credential on precisely the same attributes, and he moreover knows the blinding factors $(\delta_i)_{i=1}^l$ corresponding to this issuance. It corresponds to the fourth property of Def. 2, and is similar to an assumption Brands needed to prove his scheme secure (Ass. 3). In particular, the level of security of Brands' scheme is retained. We refer to [23, App. A] for a detailed heuristic analysis of Ass. 2.

$$
\begin{array}{rcl}
\mathcal{U} & & \mathcal{I} \\
 & & \text{(knows: } (x_i^*)_{i=1}^{l-1}; x_0, (\phi_i)_{i=1}^{l-1}) \\
 & & (r_i)_{i=1}^{l}, x_l \in_R \mathbb{Z}_q \\
 & & \left(c_i \leftarrow (g^{r_i}, g^{x_i^*} f_i f^{r_i})\right)_{i=1}^{l-1} \\
 & & c_l \leftarrow (g^{r_l}, g^{x_l} f^{r_l}) \\
 & & h \leftarrow \prod_{i=1}^{l-1} g_i^{x_i^*+\phi_i} g_l^{x_l} h_0 \\
 & & z \leftarrow h^{x_0}, \quad (z_i \leftarrow c_i^{x_0})_{i=1}^{l} \\
 & & w \in_R \mathbb{Z}_q, \quad a \leftarrow g^{w}, \quad b \leftarrow h^{w} \\
 & & \tilde{f} \leftarrow f^{w}, \quad (e_i \leftarrow c_i^{w})_{i=1}^{l} \\
 & \xleftarrow{\substack{h,z,(c_i,z_i)_{i=1}^{l}; \\ a,b,\tilde{f},(e_i)_{i=1}^{l}}} & \\
\alpha \in_R \mathbb{Z}_q^*, \quad \beta, \gamma \in_R \mathbb{Z}_q & & \\
h' \leftarrow h^{\alpha}, \quad z' \leftarrow z^{\alpha} & & \\
a' \leftarrow h_0^{\beta} g^{\gamma} a, \quad b' \leftarrow (z')^{\beta} (h')^{\gamma} b^{\alpha} & & \\
\left(\begin{array}{r} \delta_i \in_R \mathbb{Z}_q, \quad c_i' \leftarrow c_i \cdot (g,f)^{\delta_i} \\ z_i' \leftarrow z_i \cdot (h_0, \hat{f})^{\delta_i} \\ e_i' \leftarrow (z_i')^{\beta} (c_i')^{\gamma} e_i \cdot (a, \tilde{f})^{\delta_i} \end{array}\right)_{i=1}^{l} & & \\
c' \leftarrow \mathcal{H}([c_i', z_i', e_i']_{i=1}^{l}; h', z', a', b') & & \\
c \leftarrow c' + \beta \bmod q & \xrightarrow{c} & \\
 & \xleftarrow{r} & r \leftarrow c x_0 + w \bmod q \\
a \overset{?}{=} g^{r} h_0^{-c}, \quad b \overset{?}{=} h^{r} z^{-c} & & \\
\tilde{f} \overset{?}{=} f^{r} \hat{f}^{-c}, \quad (e_i \overset{?}{=} c_i^{r} z_i^{-c})_{i=1}^{l} & & \\
r' \leftarrow r + \gamma \bmod q & &
\end{array}
$$

Fig. 1. Issuance protocol of the encrypted credential scheme.

$$
\begin{array}{rcl}
\mathcal{U} & & \mathcal{V} \\
\text{(knows: } h', z', (c_i', z_i')_{i=1}^{l}, c', r'; \alpha) & & \text{(knows: } (y_i)_{i=1}^{l}, \lambda) \\
u \in_R \mathbb{Z}_q, \quad a \leftarrow (h')^{u} & & \\
 & \xrightarrow{\substack{a;(c_i',z_i')_{i=1}^{l}, \\ h',z',c',r'}} & \\
 & \xleftarrow{c} & c \in_R \mathbb{Z}_q \\
r \leftarrow u + c\alpha^{-1} \bmod q & \xrightarrow{r} & \\
 & & b_1 \leftarrow c' \overset{?}{=} \mathcal{H}([c_i', z_i', (c_i')^{r'} (z_i')^{-c'}]_{i=1}^{l}; \\
 & & \qquad h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'}) \\
 & & b_2 \leftarrow (h')^{r} \overset{?}{=} a(D((c_1')^{y_1} \cdots (c_l')^{y_l}) h_0)^{c} \\
 & & b \leftarrow [b_1 \wedge b_2] \\
 & \xleftarrow{b} &
\end{array}
$$

Fig. 2. Verification protocol of the encrypted credential scheme.

Assumption 2. *If $\mathcal{U}'$ produces, after $K \geq 0$ arbitrarily interleaved executions of the protocol in Fig. 1 on adaptively chosen $\left(x^*_{ji}\right)_{i=1}^{l-1}$ $(j = 1, \ldots, K)$ a tuple $(h', (c'_i)_{i=1}^{l}, \alpha, z', (z'_i)_{i=1}^{l}, c', r')$, then this tuple does not satisfy (2), or with overwhelming probability there exists a $j \in \{1, \ldots, K\}$ such that*

$$\mathcal{U}' \text{ knows values } (\delta_i)_{i=1}^{l} \text{ such that } (c'_i)_{i=1}^{l} = \left(c_{ji}(g, f)^{\delta_i}\right)_{i=1}^{l}, \tag{3}$$

where $(c_{ji})_{i=1}^{l}$ is the list of encryptions coming from the first round of the j-th issuance execution. More formally, there exists a p.p.t. extractor $\mathcal{E}$ that may use $\mathcal{U}'$ as a subroutine and also outputs a tuple $(h', (c'_i)_{i=1}^{l}, \alpha, z', (z'_i)_{i=1}^{l}, c', r')$, but additionally outputs the values $(h_j, (c_{ji})_{i=1}^{l})_{j=1}^{K}$ on which the user is issued credentials, and a value $\tau \in \{0, \ldots, K\}$: $\tau = 0$ meaning that (3) is not satisfied for any j (and implying that (2) is not satisfied), and $\tau \neq 0$ meaning that it is satisfied for $j = \tau$, in which case the extractor also outputs a tuple $(\delta_i)_{i=1}^{l}$ satisfying (3).

6 Variations

It is possible to adjust the scheme of Sect. 4 to the scenario where all parties *only* learn the encryptions $(c^*_i)_{i=1}^{l-1}$. However, it turns out that for the computation of h the issuer then needs secret values $(y_i)_{i=1}^{l}, \lambda$, and thus we need to identify the role of $\mathcal{I}$ with the role of $\mathcal{V}$. This adjustment is quite simple: in Fig. 1, the issuer now computes $(c_i)_{i=1}^{l-1}$ and h as

$$(c_i \leftarrow c^*_i \cdot (g^{r_i}, f_i f^{r_i}))_{i=1}^{l-1}, \qquad h \leftarrow D(c_1^{y_1} \cdots c_l^{y_l})h_0.$$

The remainder of the scheme remains unchanged (this variation is discussed in detail in [23]). Furthermore, it is possible to combine our encrypted credential schemes with Brands' credential scheme [7]: for instance, it is straightforward to construct a scheme for the case that both the user and issuer only know a specific (possibly non-overlapping) subset of the attributes in the clear. For these constructions, the security proofs are similar. Recall that our schemes achieve the same level of security as Brands' schemes.

We notice that the semi-honest behavior of the verifier (as mentioned in Sect. 5) can be achieved by implementing $\mathcal{V}$ as a set of parties using threshold cryptography. Indeed, the secret keys can be threshold shared among the parties using a distributed key generation protocol [21], and the plaintext equality test in the protocol of Fig. 2 can then be securely evaluated. Additionally, the possibility of multiple verifiers can be realized using verifiable secret redistribution, where the verifier redistributes his secret key to other verifiers (possibly also implemented as sets of parties) [25].

7 Conclusions

The notion of *encrypted credential schemes* is introduced and defined as a new concept in the area of anonymous credentials. We have presented and analyzed various efficient constructions of this new type of digital credential schemes, starting from a credential scheme by Brands [7]. Our schemes are comparable in security and efficiency to Brands' schemes, except that the cost grows linearly with the number of encrypted attributes. These new credential schemes have a lot of interesting applications, in particular to scenarios where the user *is not allowed* to learn the attributes in the clear (e.g., letters of recommendation), or where the user *does not want* to learn these data (e.g., medical information about illnesses). The schemes can also be used in the context of secure multiparty computation, where credentials can be issued on the results of a secure computation, which may be used as input in another secure computation, without the parties performing the computations learning anything about the links between these computations, and about the secret data.

The encrypted credential schemes constructed in this paper operate with single-use credentials. It would be interesting to extend existing multi-use credential schemes (such as [11,12]) with the functionality of encrypted attributes. Since our techniques do not readily apply to this case, we leave this as an open problem. Additionally, the construction of an encrypted credential scheme with publicly verifiable credentials remains open.

Acknowledgments. This work has been funded in part by the European Community's Sixth Framework Programme under grant number 034238, SPEED project - Signal Processing in the Encrypted Domain, in part by the IAP Program P6/26 BCRYPT of the Belgian State (Belgian Science Policy), and in part by the European Commission through the ICT program under contract ICT-2007-216676 ECRYPT II. The second author is supported by a Ph.D. Fellowship from the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

1. Bangerter, E., Camenisch, J., Lysyanskaya, A.: A cryptographic framework for the controlled release of certified data. In: Security Protocols. LNCS, vol. 3957, pp. 20–42. Springer-Verlag, Berlin (2004)
2. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A.: P-signatures and non-interactive anonymous credentials. In: TCC '08. LNCS, vol. 4948, pp. 356–374. Springer-Verlag, Berlin (2008)
3. Brands, S.: Untraceable off-line cash in wallet with observers. In: CRYPTO '93. LNCS, vol. 773, pp. 302–318. Springer-Verlag, Berlin (1993)
4. Brands, S.: Off-line electronic cash based on secret-key certificates. In: LATIN '95. LNCS, vol. 911, pp. 131–166. Springer-Verlag, Berlin (1995)
5. Brands, S.: Restrictive blinding of secret-key certificates. In: EUROCRYPT '95. LNCS, vol. 921, pp. 231–247. Springer-Verlag, Berlin (1995)
6. Brands, S.: Rapid demonstration of linear relations connected by boolean operators. In: EUROCRYPT '97. LNCS, vol. 1233, pp. 318–333. Springer-Verlag, Berlin (1997)
7. Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates - Building In Privacy. Ph.D. thesis, Eindhoven University of Technology, Eindhoven (1999), http://www.credentica.com/the_mit_pressbook.html

8. Brands, S., Demuynck, L., De Decker, B.: A practical system for globally revoking the unlinkable pseudonyms of unknown users. In: ACISP '07. LNCS, vol. 4586, pp. 400–415. Springer-Verlag, Berlin (2007)
9. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Compact e-cash. In: EUROCRYPT '05. LNCS, vol. 3494, pp. 302–321. Springer-Verlag, Berlin (2005)
10. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: EUROCRYPT '01. LNCS, vol. 2045, pp. 93–118. Springer-Verlag, Berlin (2001)
11. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: SCN '02. LNCS, vol. 2576, pp. 268–289. Springer-Verlag, Berlin (2002)
12. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: CRYPTO '04. LNCS, vol. 3152, pp. 56–72. Springer-Verlag, Berlin (2004)
13. Canetti, R.: Security and composition of multi-party cryptographic protocols. Journal of Cryptology 13(1), 143–202 (2000)
14. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO '82. pp. 199–203. LNCS, Plenum Press (1983)
15. Chaum, D.: Security without identification: Transaction systems to make big brother obsolete. Communications of the ACM 28(10), 1030–1044 (1985)
16. Chaum, D., Evertse, J.: A secure and privacy-protecting protocol for transmitting personal information between organizations. In: CRYPTO '86. LNCS, vol. 263, pp. 118–167. Springer-Verlag, Berlin (1987)
17. Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: CRYPTO '88. LNCS, vol. 403, pp. 319–327. Springer-Verlag, Berlin (1990)
18. Chaum, D., Pedersen, T.: Wallet databases with observers. In: CRYPTO '92. LNCS, vol. 740, pp. 89–105. Springer-Verlag, Berlin (1993)
19. Cramer, R.: Modular Design of Secure yet Practical Cryptographic Protocols. Ph.D. thesis, University of Amsterdam, Amsterdam (1997)
20. Damgård, I.: Payment systems and credential mechanisms with provable security against abuse by individuals. In: CRYPTO '88. LNCS, vol. 403, pp. 328–335. Springer-Verlag, Berlin (1988)
21. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. In: EUROCRYPT '99. LNCS, vol. 1592, pp. 295–310. Springer-Verlag, Berlin (1999)
22. Lysyanskaya, A., Rivest, R., Sahai, A., Wolf, S.: Pseudonym systems. In: SAC '99. LNCS, vol. 1758, pp. 184–199. Springer-Verlag, Berlin (1999)
23. Mennink, B.: Encrypted Certificate Schemes and Their Security and Privacy Analysis. Master's thesis, Eindhoven University of Technology, Eindhoven (2009)
24. Verheul, E.: Self-blindable credential certificates from the weil pairing. In: ASIACRYPT '01. LNCS, vol. 2248, pp. 533–551. Springer-Verlag, Berlin (2001)
25. Wong, T., Wang, C., Wing, J.: Verifiable secret redistribution for archive system. In: IEEE Security in Storage Workshop. pp. 94–106 (2002)

A Brands' Credential Scheme

In this appendix, the credential scheme by Brands, as introduced in Sect. 2, is discussed in technical detail.

Key Generation. Given a security parameter k, system parameters (q, g), with prime $q > 2^k$, are generated first, followed by the generation of a secret key

$x_0 \in_R \mathbb{Z}_q$. Finally the public key $(h_0, g_1, \ldots, g_l)$ is generated as $h_0 = g^{x_0}$ and $g_1, \ldots, g_l \in_R \langle g \rangle$.

Credential Issuance. Given attribute list $(x_i)_{i=1}^{l}$, one sets $h = g_1^{x_1} \cdots g_l^{x_l} h_0 \neq 1$. The issuance protocol is given in Fig. 3. It results in a credential for $\mathcal{U}$ on $(x_i)_{i=1}^{l}$, which consists of a tuple $(h', (x_i)_{i=1}^{l}, \alpha, z', c', r')$ satisfying

$$c' = \mathcal{H}(h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'}), \text{ and } (g_1^{x_1} \cdots g_l^{x_l} h_0)^{\alpha} = h' \neq 1. \qquad (4)$$

$$\begin{array}{rcl}
\mathcal{U} & & \mathcal{I} \\
(\text{knows: } h) & & (\text{knows: } h; x_0) \\
 & & z \leftarrow h^{x_0}, \quad w \in_R \mathbb{Z}_q \\
 & \xleftarrow{\quad z;a,b \quad} & a \leftarrow g^{w}, \quad b \leftarrow h^{w} \\
\alpha \in_R \mathbb{Z}_q^*, \quad \beta, \gamma \in_R \mathbb{Z}_q & & \\
h' \leftarrow h^{\alpha}, \quad z' \leftarrow z^{\alpha} & & \\
a' \leftarrow h_0^{\beta} g^{\gamma} a, \quad b' \leftarrow (z')^{\beta} (h')^{\gamma} b^{\alpha} & & \\
c' \leftarrow \mathcal{H}(h', z', a', b'), \quad c \leftarrow c' + \beta \bmod q & \xrightarrow{\quad c \quad} & \\
 & \xleftarrow{\quad r \quad} & r \leftarrow cx_0 + w \bmod q \\
a \stackrel{?}{=} g^{r} h_0^{-c}, \quad b \stackrel{?}{=} h^{r} z^{-c} & & \\
r' \leftarrow r + \gamma \bmod q & &
\end{array}$$

Fig. 3. Issuance protocol.

Credential Verification. For verification of a credential, the user sends the public part (h', z', c', r') to the verifier, and proves knowledge of $((x_i)_{i=1}^{l}, \alpha)$ such that (4) holds. This is a Σ-protocol for relation $\{(h'; (x_i)_{i=1}^{l}, \alpha) \mid h_0 = (h')^{\alpha^{-1}} g_1^{-x_1} \cdots g_l^{-x_l} \wedge \alpha \neq 0\}$ (note that $\alpha \neq 0$ should indeed hold as $h' \neq 1$), and it is given in Fig. 4.

$$\begin{array}{rcl}
\mathcal{U} & & \mathcal{V} \\
(\text{knows: } h', z', c', r'; (x_i)_{i=1}^{l}, \alpha) & & \\
u_1, \ldots, u_l, u_{\alpha} \in_R \mathbb{Z}_q & & \\
a \leftarrow (h')^{u_{\alpha}} g_1^{-u_1} \cdots g_l^{-u_l} & \xrightarrow{\quad a; h', z', c', r' \quad} & \\
 & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_q \\
(r_i \leftarrow u_i + c x_i \bmod q)_{i=1}^{l} & & \\
r_{\alpha} \leftarrow u_{\alpha} + c\alpha^{-1} \bmod q & \xrightarrow{\quad (r_i)_{i=1}^{l}, r_{\alpha} \quad} & \\
 & & c' \stackrel{?}{=} \mathcal{H}(h', z', g^{r'} h_0^{-c'}, (h')^{r'} (z')^{-c'}) \\
 & & (h')^{r_{\alpha}} g_1^{-r_1} \cdots g_l^{-r_l} \stackrel{?}{=} a h_0^{c}
\end{array}$$

Fig. 4. Verification protocol.

Security Analysis. Security of the above credential scheme is analyzed in [7]. Apart from some standard assumptions, the following specific assumption is needed as well.

Assumption 3. *If $\mathcal{U}'$ produces, after $K \geq 0$ arbitrarily interleaved executions of the protocol in Fig. 3 on adaptively chosen $(x_{ji})_{i=1}^{l}$ ($j = 1, \ldots, K$) a valid tuple $(h', (x_i)_{i=1}^{l}, \alpha, z', c', r')$, then this tuple does not satisfy (4), or with overwhelming probability there exists a $j \in \{1, \ldots, K\}$ such that $(x_i)_{i=1}^{l} = (\alpha x_{ji} \bmod q)_{i=1}^{l}$.*

B Proof of Thm. 1

In this appendix we prove Thm. 1. The properties to be proven (cf. Def. 2) can be divided in the first four properties concerning the issuance protocol, and the last concerning the verification protocol. These protocols will be proven secure in Sects. B.1 and B.2, respectively. We will consider the five properties for any probabilistic key generation execution, resulting in a tuple $(pk, sk_{\mathcal{I}}, sk_{\mathcal{V}})$. We assume that this protocol execution is done properly, i.e. that the system parameters are correctly constructed. Notice that the issuer is the only party who learns the encrypted attributes: the verification protocol is a secure two-party protocol, and in the issuing execution the user only learns perfectly hiding commitments of the encrypted data.

B.1 Correctness of Issuance Protocol

Proposition 1 (Completeness). *If both $\mathcal{U}$ and $\mathcal{I}$ follow the protocol, then for any attribute list $C = (x_i^*)_{i=1}^{l-1}$, the resulting credential of the issuance execution will be accepted upon verification.*

Proof. See [23, Sect. 6.4]. □

Proposition 2 (User privacy). *For any pair of attribute lists C_0, C_1, if $\mathcal{U}$ and $\mathcal{I}'$ engaged in the issuance execution for both lists, obtaining credentials $(p, s, \sigma(p))_0$, $(p, s, \sigma(p))_1$, then it is hard for malicious $\mathcal{I}'$ to guess b correctly, given $(p, \sigma(p))_b$ and $(p, \sigma(p))_{1-b}$ with $b \in_R \{0, 1\}$.*

Proof. The game played by $\mathcal{I}'$ and $\mathcal{U}$ is the following: given any two different attribute lists C_0, C_1, $\mathcal{I}'$ and $\mathcal{U}$ engage in an issuance execution for C_j ($j = 0, 1$), $\mathcal{U}$ takes $b \in_R \{0, 1\}$ and sends the public parts of the b-th and $(1-b)$-th credential to $\mathcal{I}'$ (in that order). $\mathcal{I}'$ wins if he guesses b correctly. Denote by $\Pr(A)$ the success probability of $\mathcal{I}'$ in this game. We slightly change this game, obtaining game B. Now, in each issuance execution $\mathcal{U}$ sets for each $i = 1, \ldots, l$:

$$c_i' \leftarrow (g, f)^{\delta_i}, \qquad z_i' \leftarrow (h_0, \hat{f})^{\delta_i}, \qquad e_i' \leftarrow (z_i')^{\beta}(c_i')^{\gamma}(a, \tilde{f})^{\delta_i}, \tag{5}$$

and executes the remainder as is. (Note that the resulting tuple does not yield a valid credential as $(D((c_1')^{y_1} \cdots (c_l')^{y_l})h_0)^{\alpha} = h'$ need not be satisfied. However, $\mathcal{I}'$ will not notice as he is p.p.t. and does not have the decryption key.) Denote $\mathcal{I}'$'s success probability in the new game by $\Pr(B)$. Now, the only difference between the games is in the encryptions, and as $\mathcal{I}'$ is p.p.t. and does not have the decryption key, if $\mathcal{I}'$ is able to distinguish between the two games, he is able to distinguish between the constructions of one of the $6l$ encryptions. Hence, the success probabilities in the different games are of negligible difference by the semantic security of the cryptosystem. Formally, there exists a negligible $\nu(k)$ such that

$$|\Pr(A) - \Pr(B)| < \nu(k). \tag{6}$$

We consider the success probability of $\mathcal{I}'$ in game B. We will first prove that for any public part of a credential, and any view on an issuance execution by $\mathcal{I}'$, there is exactly *one* possible secret random tuple $\mathcal{U}$ could have chosen. In particular this means that from $\mathcal{I}'$'s point of view, $(p, \sigma(p))_b$ could have come from the 0-th or 1-th issuance execution with equal probability, and similar for $(p, \sigma(p))_{1-b}$. Then, as $\mathcal{U}$ takes his values uniformly at random, $\mathcal{I}'$ can only succeed in guessing b correctly with probability $\frac{1}{2}$. Hence $\Pr(B) = \frac{1}{2}$, which by (6) implies that the success probability in the original game is upper bounded by $\frac{1}{2} + \nu(k)$ for negligible $\nu(k)$.

So we prove that for any public part of a credential, $(h', z', (c_i', z_i')_{i=1}^l, c', r')$, and all values a malicious $\mathcal{I}'$ sees during the issuance execution of a credential, $(h, z, (c_i, z_i)_{i=1}^l)$ and $(a, b, \tilde{f}, (e_i)_{i=1}^l, c, r)$ satisfying (as $\mathcal{U}$ accepted)

$$a = g^r h_0^{-c}, \qquad b = h^r z^{-c}, \qquad \tilde{f} = f^r \hat{f}^{-c}, \qquad \forall_{i=1}^l : \ e_i = c_i^r z_i^{-c}, \tag{7}$$

there exists exactly *one* possible combination of random values $\alpha, \beta, \gamma, (\delta_i)_{i=1}^l$ that $\mathcal{U}$ could have chosen to end up with that credential. The values α, β and γ are determined by (h, h'), (c, c') and (r, r'), namely as $\alpha = \log_h h'$, $\beta = c - c' \bmod q$ and $\gamma = r' - r \bmod q$. Furthermore, for each i, δ_i is determined by c_i' as $\delta_i = \log_g (c_i')_1 = \log_f (c_i')_2$. Remains to prove that this choice satisfies $c' = \mathcal{H}([c_i', z_i', e_i']_{i=1}^l; h', z', a', b')$. But the issued credential satisfies $c' = \mathcal{H}([c_i', z_i', (c_i')^{r'}(z_i')^{-c'}]_{i=1}^l; h', z', g^{r'} h_0^{-c'}, (h')^{r'}(z')^{-c'})$, from which the equality follows if $a' = g^{r'} h_0^{-c'}$, $b' = (h')^{r'}(z')^{-c'}$ and $\forall_{i=1}^l : \ e_i' = (c_i')^{r'}(z_i')^{-c'}$. But the first two equations are easy to check, and for the third we have for all $i = 1, \ldots, l$:

$$\begin{aligned}
(c_i')^{r'}(z_i')^{-c'} &= (c_i')^{\gamma}(z_i')^{\beta}(c_i')^{r}(z_i')^{-c} && \{\text{setup } r', c'\} \\
&= (c_i')^{\gamma}(z_i')^{\beta}(g^r h_0^{-c}, f^r \hat{f}^{-c})^{\delta_i} && \{\text{equation (5)}\} \\
&= (c_i')^{\gamma}(z_i')^{\beta}(a, \tilde{f})^{\delta_i} && \{\text{equation (7)}\} \\
&= e_i' && \{\text{equation (5)}\}. \qquad \square
\end{aligned}$$

Remark 1. For the proof of Prop. 2, $\mathcal{I}'$ may only work in probabilistic polynomial time[8], simply because different issuance executions might involve different encryptions. However, if the two attribute lists are the same, so $C_0 = C_1$, then the changeover to game B is unnecessary. In particular, the issuance executions then become unlinkable even for issuers with unlimited resources. This is relevant in case the issuer issues many credentials on the same attribute list.

[8] In case the issuer would know the secret decryption key, e.g. if the issuer plays the role of the verifier as well, we moreover require the issuer to be semi-honest. However, as $\mathcal{V}$ is semi-honest (Sect. 5), this is naturally enforced.

The proof of one-more unforgeability relies on tightly reducing the credentials to signatures of the blind signature scheme by Chaum and Pedersen [18]. Briefly, the blind Chaum-Pedersen signature scheme considers a cyclic group $\langle g\rangle$ and a public $h \in \langle g\rangle$ corresponding to secret key x, known by the signer. The issuance of a signature on message m happens in four rounds, starting with the user blinding m and sending it to the signer. The protocol results in a signature (m, z, c', r') such that $c' = \mathcal{H}(m, z, g^{r'} h^{-c'}, m^{r'} z^{-c'})$. The reader is referred to [18] for a more detailed discussion of the scheme. In what follows, we assume this scheme to be secure. Note that forging a Chaum-Pedersen signature is just as hard as forging a signature of the form (ξ, m, z, c', r') such that $c' = \mathcal{H}(\xi, m, z, g^{r'} h^{-c'}, m^{r'} z^{-c'})$ for any arbitrary bit string ξ. This is due to the properties of the cryptographic hash function.

Proposition 3 (One-more unforgeability). *Under the assumption that the blind Chaum-Pedersen signature scheme is secure against one-more forgeries, it is impossible for a user $\mathcal{U}'$ to, after $K \geq 0$ arbitrarily interleaved executions of Fig. 1 on adaptively chosen attribute lists $C_j = (x^*_{ji})_{i=1}^{l-1}$ ($j = 1, \ldots, K$), with non-negligible probability output $K + 1$ different credentials satisfying (2).*

Proof. Suppose it is possible, so after K executions of the protocol of Fig. 1, on adaptively chosen $(x^*_{ji})_{i=1}^{l-1}$ for $j = 1, \ldots, K$, $\mathcal{U}'$ can output $K + 1$ different credentials $(h', (c'_i)_{i=1}^{l}, \alpha, z', (z'_i)_{i=1}^{l}, c', r')$ satisfying (2), with non-negligible probability. We construct an interactive polynomial time forger $\mathcal{F}$ that is issued K Chaum-Pedersen signatures by a Chaum-Pedersen signer $\mathcal{S}$, possibly on different messages m for each execution $j = 1, \ldots, K$, and uses $\mathcal{U}'$ to output $K + 1$ different Chaum-Pedersen signatures. By assumption that is impossible, and hence we obtain a contradiction.

Let $\langle g\rangle, h_{CP}$ be the system parameters of the Chaum-Pedersen signature scheme, for which $\mathcal{S}$ knows $x = \log_g h_{CP}$. Now $\mathcal{F}$ simulates the credential issuer for Fig. 1 as follows:

1. *Initialization*: For the encryption scheme, $\mathcal{F}$ takes secret key $\lambda \in_R \mathbb{Z}_q$ and publishes $f = g^{\lambda}$. Furthermore, $\mathcal{F}$ inherits $\mathcal{S}$'s system parameters, and takes moreover $(y_i)_{i=1}^{l}, (\phi_i)_{i=1}^{l-1} \in_R \mathbb{Z}_q$ and publishes $h_0 = h_{CP}$, $f_i = g^{\phi_i}$ ($i = 1, \ldots, l-1$), $g_i = g^{y_i}$ ($i = 1, \ldots, l$), and $\hat{f} = h_0^{\lambda}$;
2. *Issuance*: For each of the K issuance protocol executions, $\mathcal{F}$ operates as follows[9]:
   i. *Commitment part 1*: $\mathcal{F}$ obtains $x^*_i \in \{0, 1\}$ from $\mathcal{U}'$ ($i = 1, \ldots, l-1$).[10] He takes $(r_i)_{i=1}^{l}, x_l \in_R \mathbb{Z}_q$, sets $(x_i \leftarrow x^*_i + \phi_i \bmod q)_{i=1}^{l-1}$, sets
$$c_i \leftarrow (g^{r_i}, g^{x_i} f^{r_i}) \text{ and } z_i \leftarrow (h_0^{r_i}, h_0^{x_i+\lambda r_i}), \text{ for each } i = 1, \ldots, l,$$

[9] For ease of presentation, the first round of the original protocol in Fig. 1, the commitment part, is separated into two phases i and ii. That is, firstly $(h, z, (c_i, z_i)_{i=1}^{l})$ is sent to $\mathcal{U}'$, and then $(a, b, \tilde{f}, (e_i)_{i=1}^{l})$.

[10] Recall that $\mathcal{U}'$ can adaptively choose the attribute list. If $\mathcal{U}'$ would adaptively choose *encrypted* attributes $[\![x^*_i]\!]$ instead (for instance in the variation of the scheme, cf. Sect. 6), $\mathcal{F}$ can still obtain the plaintext attributes by using the decryption key λ.

and $h \leftarrow g_1^{x_1} \cdots g_l^{x_l} h_0$. For the setup of z, $\mathcal{F}$ sends $\tilde{m} \leftarrow h$ to $\mathcal{S}$, in order to obtain $\tilde{z}$. The forger sends $z \leftarrow \tilde{z}$ to $\mathcal{U}'$;
   ii. *Commitment part 2*: $\mathcal{F}$ receives $\tilde{a}, \tilde{b}$ from $\mathcal{S}$, he sets $(a, b) \leftarrow (\tilde{a}, \tilde{b})$ and $\tilde{f} \leftarrow a^{\lambda}$, and for each $i = 1, \ldots, l$ he takes $e_i \leftarrow (a^{r_i}, a^{x_i + \lambda r_i})$. He sends $(a, b, \tilde{f}, (e_i)_{i=1}^{l})$ to $\mathcal{U}'$;
   iii. *Challenge*: $\mathcal{F}$ receives c from $\mathcal{U}'$ and sends $\tilde{c} \leftarrow c$ to $\mathcal{S}$;
   iv. *Response*: $\mathcal{F}$ receives $\tilde{r}$ from $\mathcal{S}$ and sends $r \leftarrow \tilde{r}$ to $\mathcal{U}'$;
3. *Signature forging*: Now $\mathcal{U}'$ outputs, with non-negligible probability, $K+1$ distinct credentials $(h', (c_i')_{i=1}^{l}, \alpha, z', (z_i')_{i=1}^{l}, c', r')$. For each of these credentials $\mathcal{F}$ computes Chaum-Pedersen forgery

$$(\overline{\xi}, \overline{z}, \overline{c}, \overline{r}, \overline{m}) \leftarrow ([c_i', z_i', (c_i')^{r'} (z_i')^{-c'}]_{i=1}^{l}, z', c', r', h'), \tag{8}$$

and he outputs these $K+1$ Chaum-Pedersen signatures.

The proof that this reduction works can be found in [23, Sect. 6.4]. □

The proof of blinding-invariance unforgeability relies on Ass. 2. However, this assumption is not sufficient: it essentially says that a malicious user cannot with non-negligible probability output any credential on a *different* plaintext attribute list than he is issued credentials on, while blinding-invariance unforgeability more generally requires that for any attribute list the user cannot output more credentials on it than he is issued. So similar to Brands' scheme [7, Ass. 4.4.5], blinding-invariance unforgeability of our scheme is slightly more general than the corresponding assumption. Therefore, it is stated without proof.

Proposition 4 (Blinding-invariance unforgeability). *If $\mathcal{U}'$ comes, after $K \geq 0$ arbitrarily interleaved executions of Fig. 1 on adaptively chosen attribute lists $C_j = (x_{ji}^*)_{i=1}^{l-1}$ $(j = 1, \ldots, K)$, with L different credentials satisfying (2), Then, for any of the attribute lists in these L credentials, the number of credentials on this list does not exceed the number of j's such that this attribute list equals C_j.*

B.2 Correctness of Verification Protocol

We need to prove that the verification protocol in Fig. 2 is a secure two-party protocol for proving knowledge of α such that $(h', (c_i')_{i=1}^{l}, \alpha, z', (z_i')_{i=1}^{l}, c', r')$ is a valid credential. The equality $c' \overset{?}{=} \mathcal{H}(\cdot)$ can be checked publicly and is therefore assumed to hold. Consequently, the verification protocol of Fig. 2 simplifies to Fig. 5, where $\mathcal{U}$ can be an active attacker, but $\mathcal{V}$ can only be passive.

So, we need to prove that the protocol in Fig. 5 is a secure two-party protocol for $\mathcal{U}$ to prove knowledge of α such that $(h')^{\alpha^{-1}} = D((c_1')^{y_1} \cdots (c_l')^{y_l}) h_0$. The protocol should be a secure proof of knowledge for relation

$$R = \{(h', (c_i')_{i=1}^{l}; \alpha) \mid (h')^{\alpha^{-1}} = D((c_1')^{y_1} \cdots (c_l')^{y_l}) h_0 \wedge \alpha \neq 0\}.$$

We need to prove that the protocol in Fig. 5 is a proof of knowledge, and that it is secure. Demonstrating that it is a proof of knowledge is captured by proving

$$\begin{array}{rcl}
\mathcal{U} & & \mathcal{V} \\
(\text{knows: } h', (c'_i)_{i=1}^{l}; \alpha) & & (\text{knows: } h', (c'_i)_{i=1}^{l}; (y_i)_{i=1}^{l}, \lambda) \\
u \in_R \mathbb{Z}_q, \quad a \leftarrow (h')^u & \xrightarrow{\quad a \quad} & \\
 & \xleftarrow{\quad c \quad} & c \in_R \mathbb{Z}_q \\
r \leftarrow u + c\alpha^{-1} \bmod q & \xrightarrow{\quad r \quad} & b \leftarrow \left[(h')^r \stackrel{?}{=} a(D((c'_1)^{y_1} \cdots (c'_l)^{y_l})h_0)^c\right] \\
 & \xleftarrow{\quad b \quad} &
\end{array}$$

Fig. 5. Simplified verification protocol of the encrypted credential scheme.

'completeness' and 'special soundness' (cf. Sect. 2) for relation R. For security, using the multiparty computation model of [13], we need to prove that the adversarial view on the protocol can be simulated for any allowed adversary structure: $\mathcal{V}$ being semi-honest or $\mathcal{U}$ being malicious. Therefore, we construct two simulators that may both use the adversarial party as a subroutine, and that simulate the conversations of the corrupted party with an honest participant in an indistinguishable way, on any common input $(h', (c'_i)_{i=1}^{l})$.

Proposition 5. *The protocol in Fig. 5 is complete and special sound.*

Proof. See [23, Sect. 6.4]. $\square$

Proposition 6. *For any common input $(h', (c'_i)_{i=1}^{l})$, the protocol in Fig. 5 can be simulated in a perfectly indistinguishable way, for any semi-honest $\mathcal{V}'$.*

Proof. Given a common input $(h', (c'_i)_{i=1}^{l})$. For any honest prover $\mathcal{U}$ and semi-honest verifier $\mathcal{V}'$ following the protocol, the real conversations satisfy the following distribution[11]:

$$\big\{(a, c, r, b) \mid u, c \in_R \mathbb{Z}_q; a \leftarrow (h')^u; r \leftarrow u + c\alpha^{-1} \bmod q;$$
$$b \leftarrow \left[(h')^r \stackrel{?}{=} a(D((c'_1)^{y_1} \cdots (c'_l)^{y_l})h_0)^c\right]\big\}.$$

This distribution is perfectly simulated by:

$$\big\{(a, c, r, b) \mid c, r \in_R \mathbb{Z}_q; a \leftarrow (h')^r(D((c'_1)^{y_1} \cdots (c'_l)^{y_l})h_0)^{-c};$$
$$b \leftarrow \left[(h')^r \stackrel{?}{=} a(D((c'_1)^{y_1} \cdots (c'_l)^{y_l})h_0)^c\right]\big\}.$$

Note that the simulator knows the values $((y_i)_{i=1}^{l}, \lambda)$ as he may use $\mathcal{V}'$ as subroutine, and therefore he can compute $D((c'_1)^{y_1} \cdots (c'_l)^{y_l})$, where D is the decryption function. $\square$

The construction of a simulator for the view of a malicious prover $\mathcal{U}'$ on the protocol relies on Ass. 2. We can assume that $\mathcal{U}'$ did $K \geq 0$ credential issuance queries, and output a tuple $(h', (c'_i)_{i=1}^{l}, \alpha, z', (z'_i)_{i=1}^{l}, c', r')$. We recall that the equation $c' = \mathcal{H}(\cdot)$ of (2) is assumed to hold.

[11] Effectively, $b = 1$ by construction. To keep the simulation clear, it is however denoted in full.

Proposition 7. *For any common input $(h', (c_i')_{i=1}^l)$, the protocol in Fig. 5 can be simulated in a perfectly indistinguishable way, for any malicious $\mathcal{U}'$.*

Proof. Given a common input $(h', (c_i')_{i=1}^l)$. For any prover $\mathcal{U}'$ and honest verifier $\mathcal{V}$, the real conversations are as follows:

– Receive a from $\mathcal{U}'$, send $c \in_R \mathbb{Z}_q$ to $\mathcal{U}'$, and receive r from $\mathcal{U}'$;
– Set $b \leftarrow \left[(h')^r \overset{?}{=} a(D((c_1')^{y_1} \cdots (c_l')^{y_l})h_0)^c\right]$, and output (a, c, r, b).

We construct a simulator that also has input $(h', (c_i')_{i=1}^l)$ and may use $\mathcal{U}'$ as a subroutine:

– Receive a from $\mathcal{U}'$, send $c \in_R \mathbb{Z}_q$ to $\mathcal{U}'$, and receive r from $\mathcal{U}'$;
– Use the extractor $\mathcal{E}$ of Ass. 2 to obtain $(h_j, (c_{ji})_{i=1}^l)_{j=1}^K$ and $\tau \in \{0, \ldots, K\}$;
– Set $b \leftarrow \begin{cases} 1, \text{ if } \tau \neq 0 \text{ and } (h')^r = ah_\tau^c, \\ 0, \text{ if } \tau = 0 \text{ or } (h')^r \neq ah_\tau^c; \end{cases}$
– Output (a, c, r, b).

Remains to prove that these two distributions are indistinguishable, given any common input $(h', (c_i')_{i=1}^l)$. But the values (a, c, r) are constructed the same in both conversations, remains to show that b is distributed the same in both sets.

Suppose that in the real conversation $b = 1$. By the special soundness property (Prop. 5), with overwhelming probability $\mathcal{U}'$ knows an α such that $(h')^{\alpha^{-1}} = D((c_1')^{y_1} \cdots (c_l')^{y_l})h_0$. By Ass. 2 (and as $c' = \mathcal{H}(\cdot)$ holds), this implies that with overwhelming probability there exists a j such that (3) holds, which by definition means that $\tau \neq 0$. It moreover implies that:

$$\begin{aligned} (h')^r &= a(D((c_1')^{y_1} \cdots (c_l')^{y_l})h_0)^c && \{\text{since } b = 1\} \\ &= a(D((c_{\tau 1})^{y_1} \cdots (c_{\tau l})^{y_l})h_0)^c && \{\text{equation (3)}\} \\ &= ah_\tau^c && \{\text{by construction}\}. \end{aligned}$$

So by construction the simulator sets $b = 1$ as well.

Conversely, suppose that in the simulated conversation $b = 1$. By construction this means that $\tau \neq 0$ and $(h')^r = ah_\tau^c$. By definition, $\tau \neq 0$ implies that (3) is satisfied with $j = \tau$. Now:

$$\begin{aligned} (h')^r &= ah_\tau^c && \{\text{since } b = 1\} \\ &= a(D((c_{\tau 1})^{y_1} \cdots (c_{\tau l})^{y_l})h_0)^c && \{\text{by construction}\} \\ &= a(D((c_1')^{y_1} \cdots (c_l')^{y_l})h_0)^c && \{\text{equation (3)}\}, \end{aligned}$$

which implies that also in the real execution $b = 1$. Concluding, with overwhelming probability b is computed the same in both conversations, and hence the real and simulated conversations are perfectly indistinguishable. $\square$