On linkability and malleability in self-blindable credentials

Jaap-Henk Hoepman[1], Wouter Lueks[1], and Sietse Ringers[2]

[1] Radboud University, Nijmegen, The Netherlands
{jhh,lueks}@cs.ru.nl
[2] Johann Bernoulli Institute for Mathematics and Computer Science, University of Groningen, The Netherlands
s.ringers@rug.nl

Abstract. Self-blindable credential schemes allow users to anonymously prove ownership of credentials. This is achieved by randomizing the credential before each showing in such a way that it still remains valid. As a result, each time a different version of the same credential is presented. A number of such schemes have been proposed, but unfortunately many of them are broken, in the sense that they are *linkable* (i.e., failing to protect the privacy of the user), or *malleable* (i.e., they allow users to create new credentials using one or more valid credentials given to them). In this paper we prove a general theorem that relates linkability and malleability in self-blindable credential schemes, and that can test whether a scheme is linkable or malleable. After that we apply the theorem to a number of self-blindable credential schemes to show that they suffer from one or both of these issues.

1 Introduction

The indiscriminate collection and processing of personal data, and the consequences to the privacy of citizens, has been getting more and more attention over the last few years. As a result, there is an increasing demand for technologies that put privacy and control back in the hands of the user. In the case of digital identity management, in particular, it is both highly desirable and non-trivial to have privacy-friendly solutions.

Anonymous credentials are a promising technique for secure and privacy-friendly identity management. They are given by an issuer to the user, who can then prove possession of it to other parties. This showing should be such that it is infeasible for the issuer, the verifier or any other party to determine whether two transactions did or did not originate from the same user (this property is called *multi-show unlinkability*, or just unlinkability for short). Additionally, credentials have to be unforgeable, in the sense that the user cannot create his own credential, or modify one or more existing ones in order to obtain a new credential (this kind of forgeability is called *malleability* and plays and important role in this paper). A number of such systems already exist; we mention, for example, Idemix [4,10] and U-Prove [3,12]. Both of these are *attribute based*, meaning that a credential may contain multiple attributes (which are pieces of information or statements, generally about the owner of the credential). These systems tend to be complex, however, which is why considerable effort has gone into simpler credential systems that have no attributes (for example [9]; see also Example 3.1). Instead, such credentials are either valid or invalid, resulting in simpler constructions that are easier to study and potentially more efficient, allowing for practical implementations of such credentials on smart cards. Naturally, such credential schemes still have to be unlinkable and unforgeable.

This article was presented at *The 9th WISTP International Conference on Information Security Theory and Practice (WISTP'2015)*, and has been published in: *Information Security Theory and Practice*, Ed. by R. Akram and S. Jajodia. Vol. 9311. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 203–218. The final publication is available at link.springer.com.

A simple method to construct (not necessarily anonymous) credentials would be to sign the user's public key (for example in the form of an X.509 certificate). The signature, together with the public key, then form the credential. To prevent replay attacks (e.g., a malicious verifier reusing a user's public key and signature to authenticate itself elsewhere), when showing the credential the user proofs knowledge of the private key of his credential without disclosing the private key to the verifier (using, for example, a zero-knowledge proof or a challenge-response). A problem with this simple scheme, however, is that the user presents the same certificate on each use, making all uses of the same credential linkable. One technique for preventing such linkability is to modify the credential before each showing, in such a way that it remains valid. This is called *blinding*, and credential schemes that use this technique are called *self-blindable credential schemes*. The first example of such a scheme was given by Verheul in the same paper that defines the notion of self-blindability [13]. The advantage of blinding credentials in such a way is that it is easy for the user (blinding is usually cheap) and for the verifier (verifying a blinded signature is generally not much different from verifying an ordinary signature).

In the past decade, a number of such self-blindable credential schemes have been proposed [4,7,9,11,13]. Unfortunately, many of them are broken, in the sense that transactions are linkable or the credentials are malleable, or even both. In this paper we uncover a common theme in the cause of the problem of each of these schemes: the dependence of the public key and signature on the private key of the credential can often be exploited to achieve linkability or malleability. This suggests there is a trade-off between the two. After having introduced and defined the relevant concepts in Section 3, we show this by proving a general theorem in Section 4 that makes it easy to determine whether a self-blindable credential scheme is linkable. The theorem exhibits an interesting and strong relationship between linkability and malleability of the credential scheme. We then apply this theorem in Section 5 to show that several proposed self-blindable schemes in the literature are linkable, and present explicit counter-examples as well. The theorem also indicates in which directions to look for self-blindable credential schemes that are both unlinkable and unmalleable.

2 Notations and conventions

In this paper we use the following notations and conventions. A bilinear group pair (G_1, G_2) consists of two cyclic groups (that we will write additively), both of prime order p, such that there exists a a *bilinear map* or *pairing*; that is, a map $e\colon G_1 \times G_2 \to G_T$ (with G_T a multiplicative group of order p) satisfying the following properties:

– *Bilinearity*: for all $G, G' \in G_1$ and $H, H' \in G_2$ we have $e(G + G', H) = e(G, H)e(G', H)$ and $e(G, H + H') = e(G, H)e(G, H')$.
– *Non-degeneracy*: Denoting the generators of G_1 and G_2 with $P \in G_1$, $Q \in G_2$ respectively, the element $e(P, Q)$ is a generator of G_T (that is, it is unequal to $1 \in G_T$).
– *Computability*: There exists an efficient algorithm for computing $e(G, H)$ for any $G \in G_1$, $H \in G_2$.

Such pairings exist for some special classes of elliptic curves. Usually, three distinct types of bilinear group pairs are distinguished:

– Type 1: $G_1 = G_2$.
– Type 2: $G_1 \neq G_2$, but there exists an efficiently computable group isomorphism $\phi\colon G_2 \to G_1$.
– Type 3: $G_1 \neq G_2$, and there is no known efficiently computable group isomorphism $\phi\colon G_2 \to G_1$.

For more information about bilinear group pairs and pairings we refer to [8]; see also, for example, Chapters I and X from [1].

We consider the coefficient k of a group element $K = kP$ to be an element of $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$. Blinding factors will be denoted with Greek letters α, β, γ. We denote variables which have been blinded with a bar on top of them, for example $\overline{K}$.

3 Self-blindable credentials

A credential scheme is a set of protocols in which an *issuer* (or *identity provider*) can issue a credential to a user, who can then show this credential to a *verifier* (or *service provider*), so that the verifier becomes convinced that the user indeed has the credential, and that it was given to him by the issuer. For the purposes of this paper we assume that there is a single issuer, and that he creates all certificates using the same private key (our results easily extend to the general case). Such credential schemes must provide at least the following two protocols:

Issue: This is an interactive protocol between a user and the issuer. The user provides the issuer with the information it needs (if the issuer does not already know this information) in order to create the credential C. The issuer checks whether the user is allowed to have the credential C, and if so, creates it and sends it to the user.

ShowCredential: This is an interactive protocol between a user and a verifier, in which the user convinces the verifier that he owns a credential C and that it is valid (i.e., that it was given to him by the issuer, and if the credential scheme allows for revoking, that it has not been revoked).[3]

A credential scheme may also allow for credentials to be revoked; in that case there is also a Revoke protocol, which revokes (invalidates) a credential. Additionally, during the ShowCredential protocol an algorithm RevocationCheck is executed, which checks if a credential has been revoked.

We expect any credential scheme, be it attribute-based, self-blindable or both, to satisfy the following properties.

– *Multi-show unlinkability*: It should be impossible for any party to tell whether two executions of the ShowCredential protocol involved the same credential or two different ones.[4]
– *Issuer unlinkability*: The issuer cannot decide if a run of the Issue and a run of the ShowCredential protocol did or did not originate from the same credential.
– *Unforgeability*: Only the issuer can create valid credentials.
– *Offline issuer*: The issuer is not involved in the verification of credentials.
– *Non-transferability*: Users cannot transfer their credentials to other users.

3.1 Definitions

In all self-blindable credential schemes that we know of, a credential consists of a private key k, a corresponding public key K, and a signature S over the public key that the issuer gives to the

[3] Attribute-based credential schemes such as U-Prove and Idemix generally also allow selective disclosures of attributes. Such disclosures, however, necessarily reduces the anonymity set of the credential (and may even identify it uniquely).

[4] In the case of attribute-based credential schemes, the unlinkability that the first and second properties describe only need to hold within the set of credentials that have disclosed the same attributes. That is, for example, given two executions of the ShowCredential protocol in which the same attributes with the same values were disclosed, it should be impossible to tell whether one or two credentials were involved. A similar adaptation holds for the second property.

In the remainder of this paper, we assume for simplicity that no attributes are disclosed in the ShowCredential protocol.

owner of the credential. That is, a credential C is of the form

$$C = (k, K, S) \in \mathcal{P} \times \mathcal{K} \times \mathcal{S}$$

where $\mathcal{P}$, $\mathcal{K}$ and $\mathcal{S}$ are the sets of private keys, public keys and signatures, respectively. We shall write $\mathcal{C}$ for the product $\mathcal{C} = \mathcal{P} \times \mathcal{K} \times \mathcal{S}$. Let us say that an element $(k, K, S) \in \mathcal{P} \times \mathcal{K} \times \mathcal{S}$ is *valid* when k is the private key corresponding to K and S is a valid signature over K with respect to the issuer's signing key.

Self-blindable credentials, introduced by Verheul [13], are credentials that the user modifies each time before he shows it to a verifier, in such a way that it remains valid, and such that multiple transactions cannot be linked to each other. We define this notion as follows.[5]

Definition 3.1. *A credential scheme is called* self-blindable *if*

1. *There exists a blinding-factor space $\mathcal{B}$ and an efficiently computable map*

$$B \colon \mathcal{C} \times \mathcal{B} \to \mathcal{K} \times \mathcal{S},$$

such that if the credential $C = (k, K, S) \in \mathcal{C}$ is valid and $B(C, \alpha) = (\overline{K}, \overline{S})$ for $\overline{K} \in \mathcal{K}$ and $\overline{S} \in \mathcal{S}$, then $\overline{S}$ is a valid signature over $\overline{K}$ for any $\alpha \in \mathcal{B}$;

2. *In the* ShowCredential *protocol, the credential C is blinded to $(\overline{K}, \overline{S}) = B(C, \alpha)$ for a random $\alpha \in_R \mathcal{B}$, after which $\overline{K}$ and $\overline{S}$ are used as the public key and signature respectively in the remainder of the* ShowCredential *protocol.*

Most self-blindable credential schemes that we know of have a ShowCredential protocol of the following form:

1. The user blinds K and S using the blinding map B and sends the blinded values $\overline{K}$, $\overline{S}$ to the verifier, who then non-interactively checks that $\overline{S}$ is a valid signature over $\overline{K}$.
2. Afterwards, the user and verifier engage in a (possibly zero-knowledge) proof in which the user convinces the verifier that he knows the private key k and blinding factor α from which he calculated $\overline{K}$ (i.e., the first element from the tuple $(\overline{K}, \overline{S}) = B((k, K, C), \alpha)$).

We purposefully do not include the private key in the blinded credentials (that is, we do not demand that $B(C, \alpha) = (\overline{k}, \overline{K}, \overline{S})$, where $\overline{k}$ is the private key corresponding to $\overline{K}$), because if such a map B were to exist then anyone can, given one credential, create arbitrary new ones. That is, there would be no distinction between the creation of new credentials by the issuer and blinding an existing credential. In terms of Definition 3.3, the system would then be 1-malleable.

Example 3.1. As a first example we consider the self-blindable credential by Hoepman et al. [9], which is based on the original scheme by Verheul [13]. Here we use the Chaum–Pedersen [6] signature scheme, as follows. Consider a Type 1 pairing $e \colon G_1 \times G_2 \to G_T$, with all groups of prime order p, and take generators P and Q for G_1 and G_2 respectively. Then the private signing key of the issuer is a number $a \in \mathbb{Z}_p$, and the corresponding public key is $A = aQ \in G_2$.

The space of private keys of credentials is $\mathcal{P} = \mathbb{Z}_p$, and for a private key $k \in \mathbb{Z}_p$ the corresponding public key is $K = kP \in G_1$. The signature on K is then a Chaum–Pedersen signature $S = aK$, which can be verified by

$$e(K, A) \overset{?}{=} e(S, Q).$$

[5] In [13], Verheul puts four extra demands on the blinding map B besides item 1 in our definition, that are meant to exclude edge cases that could never lead to desirable properties in a credential schemes. Instead of including these four extra properties, we describe the role of the blinding map B more directly in the second item in Definition 3.1.

Thus, we have $\mathcal{K} = \mathcal{S} = G_1$.

Blinding the public key is done by multiplying it by a random number $\alpha \in \mathbb{Z}_p$, that is, $\overline{K} = \alpha kP$, and similarly for the signature: $\overline{S} = \alpha akP$. The verification equation then becomes

$$e(\overline{K}, A) \overset{?}{=} e(\overline{S}, Q).$$

If $\overline{K}$ and $\overline{S}$ are blinded by the same value $\alpha \in \mathbb{Z}_p$, and if the unblinded signature is a valid Chaum–Pedersen signature over the unblinded public key K, then this equation holds.

The problem of this system is not linkability, but malleability. Given a credential (k, K, S) on its private key k the user can easily create a new credential $(\alpha k, \alpha K, \alpha S)$ on any other private key αk. This means that a user that has access to the internals of his credential can create a new credential over any private key $\overline{k} \in \mathbb{Z}_p$, without involving the issuer. (Hoepman et al. mitigate this attack by storing the private key on a smart card, so that the user cannot access it directly. It is, however, still a problem, for example because revocation in such a system would be impossible, because there is nothing that binds the private key to the user.)

We will examine this form of forgeability more closely in Definition 3.3 and Example 5.3. In this case, it is a consequence of the linearity of the Chaum–Pedersen signature S in the private key k. Later on, in Example 5.1, we will see how using a signature scheme that is nonlinear in k results in linkability.

This paper is mostly concerned with how the blinded public key $\overline{K}$ and blinded signature $\overline{S}$ depend on the private key k and blinding factor α. Taking the blinded public key $\overline{K}$, we will denote the dependency of $\overline{K}$ on k by writing

$$\overline{K} = \mathsf{PubKey}(k, \alpha)$$

for a certain function $\mathsf{PubKey}\colon \mathcal{P} \times \mathcal{B} \to \mathcal{K}$. Similarly,

$$\overline{S} = \mathsf{Sig}_{SK}(k, \alpha).$$

for a certain function $\mathsf{Sig}_{SK}\colon \mathcal{P} \times \mathcal{B} \to \mathcal{S}$. Here SK is the issuer's private key. Using these functions PubKey and Sig_{SK}, we can express the blinding map B as follows:

$$B((k, K, S), \alpha) = (\overline{K}, \overline{S}) = (\mathsf{PubKey}(k, \alpha),\ \mathsf{Sig}_{SK}(k, \alpha)).$$

We stress that these functions PubKey and Sig_{SK} need not correspond to any algorithm that is run by one of the involved parties (typically, for example, the user will calculate the blinded public key using the unblinded public key, not directly from the private key). The purpose of these functions is purely to make the dependence on the private key and blinding factor explicit.

3.2 Security properties

Having defined the basic structures and the notion of self-blindability, we next turn to the security properties that we expect credential schemes to satisfy.

Definition 3.2 (Unlinkability). *A self-blindable credential scheme is* unlinkable *if no adversary can win the following game with non-negligible advantage.*

Setup *The challenger sets up the system and creates n credentials with identifiers $1, \ldots, n$. It sends the public parameters to the adversary.*

Queries *For any $i \in \{1, \ldots, n\}$ the adversary may issue the following queries:*

Verify(i) *The adversary acts as the verifier in the* ShowCredential *protocol for the credential i, with the challenger acting as the user. The adversary sees the same interaction as a normal verifier would see.*

Corrupt(i) *The adversary requests the credential i to be corrupted. The challenger gives him the internal state of credential i.*

Challenge *The adversary selects two uncorrupted credentials i_0, i_1 from the set $\{1, \ldots, n\}$ and informs the challenger of his choice. The challenger then picks a bit $b \in_R \{0, 1\}$ at random, and runs* ShowCredential *on credential i_b with the adversary playing the role of the user while the adversary acts as the verifier. The adversary outputs a bit b'. He wins if $b = b'$.*

This definition of linkability includes a stronger notion of linkability where the adversary only gets to see two traces, and has to decide whether they belong to the same user. Given such an adversary $\mathcal{A}'$ we can then build an adversary $\mathcal{A}$ satisfying the definition above by having it perform the following actions:

Setup $\mathcal{A}$ sets up the unlinkability game with his challenger.

Queries $\mathcal{A}$ chooses two credentials i_0 and i_1 at random from the list of credentials $\{1, \ldots, n\}$ and queries his challenger on i_0. He stores the trace of the protocol run.

Challenge $\mathcal{A}$ informs his challenger that he has chosen the credentials i_0 and i_1 from the previous phase. He engages in the ShowCredential protocol on i_b and stores the trace. Then, he uses the algorithm $\mathcal{A}'$ to compare the traces from i_0 and i_b. If $\mathcal{A}'$ returns that i_0 and i_b have the same public key then $\mathcal{A}$ outputs $b' = 0$ as his guess; otherwise he outputs $b' = 1$.

Then the algorithm $\mathcal{A}$ satisfies the definitions above.

Definition 3.3 (n-malleability). *Let $\{(k_1, K_1, S_1), \ldots, (k_n, K_n, S_n)\} \in \mathcal{C}^n$ be a tuple of n valid credentials. If there exists an efficiently computable map $F \colon \mathcal{C}^n \to \mathcal{C}$ which outputs a valid credential on a new private key (that is, if*

$$(k, K, S) = F\Big((k_1, K_1, S_1), \ldots, (k_n, K_n, S_n)\Big)$$

then (k, K, S) is valid and $k \neq k_i$ for all $i = 1, \ldots, n$) then we say that the credential scheme is n-malleable.

Although malleability is nothing more than a particular kind of forgeability, it warrants a separate definition because it occurs in a number of existing credential schemes, and because it plays an important role in the theorem below. The problem that the definition above aims to capture is that new credentials can be made without the involvement or knowledge of the issuer, if the user has n credentials. We see that the credential scheme from Example 3.1 has 1-malleability: in that scheme, given a credential (k, K, S) and any $\alpha \in \mathbb{Z}_p$, the credential $(\alpha k, \alpha K, \alpha S)$ is a new valid credential. This is a problem, because the blinded credential should still be bound to the original private key k.

Note, however, that if the scheme is not attribute-based but credentials are either valid or invalid, then malleability is not necessarily a problem. Modifying an existing credential into a new one does not change any of its key properties: it was valid and it remains valid, so nothing has really changed. On the other hand, we can think of the following cases in which it would be a problem.

- The public key K may contain meaningful information such as attributes (as is the case in, for example, U-Prove). In this case, the user should not be able to manipulate this meaningful data, so it should be impossible by exploiting the malleability to obtain a new valid credential whose public key contains different information. In particular, the user should not be able to create a credential whose public key is $\overline{K}$ when given a credential with public key K.

- In a self-blindable credential scheme that is not attribute-based (for example the one from Example 3.1), issuers may issue multiple credentials (signed by different keys) instead of a single credential with multiple attributes. For example, a public key signed with private key a_1 may mean that the user is over 18, while one signed with private key a_2 could mean that he is a German citizen. In such a setting it should be impossible to combine credentials issued to different users. In this case, an underage German citizen should not be able to use his foreign friend's over 18 credential to prove that he is both over 18 and a German citizen. Normally, such a proof would show that the signed public keys in both credentials are identical, thus preventing credentials from being combined. However, malleability might make it possible to change a credential over one public key (say the foreign friend's) into another public key (say of the underage German citizen). This would make credential pooling trivial.
- Similarly, the unchecked randomization of the signed public key can make revocation – an essential feature of anonymous credential systems – all but impossible.

In the next section, we show that malleability has a strong link with linkability, and then examine a number of credential schemes that suffer from these issues.

4 Relating malleability and linkability

In the credential schemes considered in this paper, the public key $K \in \mathcal{K}$ depends linearly on the private key $k \in \mathcal{P}$. Any signature over K obviously depends on K, and therefore also on k. Thus, when considering suitable signature schemes, if the set of signatures is a group then we may take one that is either linear or not linear in k. The theorem and its corollary below then say the following: if the signature scheme is *not* linear in k, then there is linkability, while if it is linear in k then the scheme *may* be malleable. Loosely speaking, this is because if the public key and the signature do not depend on the user's private key and the blinding factor in precisely the same way, then this can be exploited. Let us now make this more precise.

We assume henceforth that $\mathcal{P}$, $\mathcal{K}$ and $\mathcal{S}$ are all groups, that we will write additively. From the corollary below and onwards it will moreover be the case that the latter two are vector spaces over $\mathcal{P}$, meaning that elements from $\mathcal{K}$ and $\mathcal{S}$ can be multiplied on the left by elements from $\mathcal{P}$: for example, $kK \in \mathcal{K}$ for $k \in \mathcal{P}$ and $K \in \mathcal{K}$. We recall the following definition.

Definition 4.1. *A map $L\colon V \to W$, with V and W being vector spaces over $\mathcal{P}$, is* linear *if $L(v + v') = L(v) + L(v')$ and $L(kv) = kL(v)$ for all $v, v' \in V$ and $k \in \mathcal{P}$.*

We denote with $\mathsf{Verify}_{PK}\colon \mathcal{K} \times \mathcal{S} \to \{\mathsf{true}, \mathsf{false}\}$ the verification function of the signature scheme under consideration, where PK is the public key of the issuer. That is, Verify_{PK} is such that

$$\mathsf{Verify}_{PK}\left(\mathsf{PubKey}(k, \alpha), \mathsf{Sig}_{SK}(k, \alpha)\right) = \mathsf{true}$$

for all k, α. On the other hand, whenever $k \neq k'$ or $\alpha \neq \alpha'$ (or both), we should have (with overwhelming probability)

$$\mathsf{Verify}_{PK}\left(\mathsf{PubKey}(k, \alpha), \mathsf{Sig}_{SK}(k', \alpha')\right) = \mathsf{false}.$$

Theorem 4.1. *Consider a self-blindable credential scheme. Suppose that for each $k, k' \in \mathcal{P}$ and $\alpha, \alpha' \in \mathcal{B}$ there exist $\ell \in \mathcal{P}$ and $\beta \in \mathcal{B}$ such that*

$$\mathsf{PubKey}(k, \alpha) + \mathsf{PubKey}(k', \alpha') = \mathsf{PubKey}(\ell, \beta). \tag{1}$$

If Sig_{SK} *also has this property for the same* ℓ*,* β*, that is,*

$$\mathsf{Sig}_{SK}(k,\alpha) + \mathsf{Sig}_{SK}(k',\alpha') = \mathsf{Sig}_{SK}(\ell,\beta) \tag{2}$$

but only *when* $k = k'$*, then there is linkability. On the other hand, if* Sig_{SK} *always has this property, and*

– *the* ShowCredential *protocol allows the user to present* $(\ell,\ \mathsf{PubKey}(\ell,\beta),\ \mathsf{Sig}_{SK}(\ell,\beta))$ *as a valid credential,*
– *the user can efficiently compute* ℓ *and* β*,*

then there is 2-malleability.

Proof. Assume that Sig_{SK} has the stated property only when $k = k'$, and that equation (1) always holds. Then if $k = k'$ we have $\mathsf{Sig}_{SK}(k,\alpha) + \mathsf{Sig}_{SK}(k',\alpha') = \mathsf{Sig}_{SK}(\ell,\beta)$ and similarly for PubKey, so

$$\begin{aligned}&\mathsf{Verify}_{PK}\,(\mathsf{PubKey}(k,\alpha) + \mathsf{PubKey}(k',\alpha'),\ \mathsf{Sig}_{SK}(k,\alpha) + \mathsf{Sig}_{SK}(k',\alpha'))\\ &\qquad = \mathsf{Verify}_{PK}(\mathsf{PubKey}(\ell,\beta),\ \mathsf{Sig}_{SK}(\ell,\beta)) = \mathsf{true}.\end{aligned}$$

On the other hand, if $k \neq k'$, then $\mathsf{Sig}_{SK}(k,\alpha) + \mathsf{Sig}_{SK}(k',\alpha')$ does not evaluate to $\mathsf{Sig}_{SK}(\ell,\beta)$. Therefore

$$\begin{aligned}&\mathsf{Verify}_{PK}\,(\mathsf{PubKey}(k,\alpha) + \mathsf{PubKey}(k',\alpha'),\ \mathsf{Sig}_{SK}(k,\alpha) + \mathsf{Sig}_{SK}(k',\alpha'))\\ &\qquad = \mathsf{false}.\end{aligned}$$

Thus, the function Verify_{PK} returns true when applied to the sum of the two credentials involved if and only if $k = k'$, so that the scheme is linkable.

The second part of the statement is obvious: if the ShowProtocol protocol does not prevent the user from using $(\ell,\ \mathsf{PubKey}(\ell,\beta), \mathsf{Sig}_{SK}(\ell,\beta))$ as a valid credential then he can present it to verifiers, even though $\mathsf{Sig}_{SK}(\ell,\beta)$ was not given to him by the issuer.

Corollary 4.2. *Suppose the function* PubKey *is linear in both arguments. If* Sig_{SK} *is linear in the second but not the first argument, then there is linkability. If* Sig_{SK} *is linear in both arguments, then there is 1-malleability.*

Proof. Suppose Sig_{SK} is linear in the second but not the first argument, and that $k = k' \in \mathcal{P}$. Then

$$\begin{aligned}&\mathsf{PubKey}(k,\alpha) + \mathsf{PubKey}(k',\alpha')\\ &\qquad = \mathsf{PubKey}(k,\alpha) + \mathsf{PubKey}(k,\alpha')\\ &\qquad = \mathsf{PubKey}(k,\alpha+\alpha'),\end{aligned}$$

and since Sig_{SK} is also linear in the second argument, we will also have $\mathsf{Sig}_{SK}(k,\alpha)+\mathsf{Sig}_{SK}(k',\alpha') = \mathsf{Sig}_{SK}(k,\alpha+\alpha')$. Thus $\mathsf{Sig}_{SK}(k,\alpha+\alpha')$ will be a valid signature over $\mathsf{PubKey}(k,\alpha+\alpha')$.

On the other hand, if $k \neq k'$ then

$$\begin{aligned}&\mathsf{PubKey}(k,\alpha) + \mathsf{PubKey}(k',\alpha')\\ &\qquad = k\mathsf{PubKey}(1,\alpha) + k'\mathsf{PubKey}(1,\alpha')\\ &\qquad = k\alpha\mathsf{PubKey}(1,1) + k'\alpha'\mathsf{PubKey}(1,1)\\ &\qquad = (k\alpha + k'\alpha')\mathsf{PubKey}(1,1)\\ &\qquad = \mathsf{PubKey}(k\alpha + k'\alpha', 1),\end{aligned}$$

but now $\mathsf{Sig}_{SK}(k, \alpha) + \mathsf{Sig}_{SK}(k', \alpha') \neq \mathsf{Sig}_{SK}(k\alpha + k'\alpha', 1)$, because Sig_{SK} is not linear in its first argument. Hence the verification function Verify_{PK} over the sum of both credentials will distinguish $k = k'$ and $k \neq k'$, so that the credential scheme is linkable.

Concerning the second statement of the corollary, if both PubKey and Sig_{SK} are linear in both arguments, then

$$\mathsf{PubKey}(k, \alpha) = \alpha \mathsf{PubKey}(k, 1) = \mathsf{PubKey}(\alpha k, 1),$$

and similarly $\mathsf{Sig}_{SK}(k, \alpha) = \mathsf{Sig}_{SK}(\alpha k, 1)$, so that $(\alpha k, \mathsf{PubKey}(k, \alpha), \mathsf{Sig}_{SK}(k, \alpha))$ is a valid credential. Therefore, there is 1-malleability.

Essentially, the corollary implies that when the verification function is used directly in the ShowCredential protocol, then it is very difficult to assure that it is neither linkable nor malleable. Indeed, if the public key is linear in the private key while the signature is not, then there is likely linkability through the verification equation of the signature scheme. On the other hand, if they are both linear in the private key then it is likely that the system suffers from malleability.

In spite of this difficulty we do not believe that it is impossible to create a self-blindable credential scheme that is neither malleable nor linkable; we will discuss this in more detail in Section 6. In the next section, we discuss a number of self-blindable credential schemes, that all suffer from one of these problems.

5 Broken self-blindable credential schemes

Example 5.1. For this example we reuse the PubKey function from Example 3.1, but this time we use the (weak) Boneh–Boyen signature scheme [2] instead. In this scheme the public and private keys of the issuer are $a \in \mathbb{Z}_p$ and $A = aQ \in G_2$ respectively, as before. A signature on $k \in \mathbb{Z}_p$ is $S = \frac{1}{a+k}P$. Setting $K = kQ \in G_2$ (note that now $K \in G_2$, contrary to Example 3.1), the signature S may be verified by checking that $e(S, A + K) \stackrel{?}{=} e(P, Q)$.

We still blind the public key and signature by multiplying it with a random number α, i.e.,

$$\overline{K} = \mathsf{PubKey}(k, \alpha) = \alpha k Q$$

and

$$\overline{S} = \mathsf{Sig}_a(k, \alpha) = \frac{\alpha}{a + k} P.$$

In addition, the user will also have to send $\overline{A} = \alpha A$, $\overline{P} = \alpha P$ and $\overline{Q} = \alpha Q$ to the verifier. The verification is done by checking

$$e(\overline{S}, \overline{A} + \overline{K}) \stackrel{?}{=} e(\overline{P}, \overline{Q}).$$

The ShowCredential protocol of this scheme might look as follows.

In this case, if $k = k'$ then $\mathsf{PubKey}(k, \alpha) + \mathsf{PubKey}(k', \alpha') = \mathsf{PubKey}(k, \alpha + \alpha')$, and similarly $\mathsf{Sig}_a(k, \alpha) + \mathsf{Sig}_a(k', \alpha') = \mathsf{Sig}_a(k, \alpha + \alpha')$, so that Verify_A will return true. On the other hand, if $k \neq k'$, then

$$\mathsf{PubKey}(k, \alpha) + \mathsf{PubKey}(k', \alpha') = \mathsf{PubKey}(\alpha k + \alpha' k', 1)$$

while

$$\mathsf{Sig}_a(k, \alpha) + \mathsf{Sig}_a(k', \alpha') \neq \mathsf{Sig}_a(\alpha k + \alpha' k', 1)$$

so Verify_A will return false. Thus, this system is linkable.

User		Verifier
choose blinding $\alpha \in_R \mathbb{Z}_p$		
send $\alpha K, \alpha S$, αA, αP, αQ	$\longrightarrow$	into $\overline{K}$, $\overline{S}$, $\overline{A}$, $\overline{P}$, $\overline{Q}$
		verify $e(\overline{S}, \overline{A} + \overline{K}) \overset{?}{=} e(\overline{P}, \overline{Q})$
$\mathrm{PK}\{(\kappa) : \overline{K} = \kappa P\}$	$\longleftrightarrow$	

Fig. 1. Self-blindable credential scheme from Example 3.1 modified to use the Boneh–Boyen signature scheme.

Example 5.2. Like the scheme from Example 3.1, the self-blindable credential scheme from Kiyomoto and Tanaka [11] uses Chaum-Pedersen signatures, but this time on a Type 1 curve (i.e., $G_1 = G_2 = G$ and $P = Q$). The issuer's public key is $A = aP$.

The private key here consists of two numbers $(\kappa, \kappa') \in \mathbb{Z}_p^2$, where κ is random while $\kappa' = m\kappa$ is a *non-repudiation private key*; here m is a number encoding some valuable piece of information related to the user. This would discourage users from sharing their credential, because if another party learns κ and κ' then it could recover m. Setting $k := \kappa + \kappa'$, the corresponding public key and signature are $K = kP$ and $S = aK = akP$. The $\mathsf{ShowCredential}$ protocol of this scheme is shown in Figure 2.

User		Verifier
choose blinding $\alpha \in_R \mathbb{Z}_p$		
send $\alpha K, \alpha S$	$\longrightarrow$	into $\overline{K}$, $\overline{S}$
		verify $e(\overline{K}, A) \overset{?}{=} e(\overline{S}, P)$
		choose nonce $\eta \in_R \mathbb{Z}_p$
into N	$\longleftarrow$	send ηP
send $\alpha\kappa N$, $\alpha\kappa' N$	$\longrightarrow$	into $\overline{M}$, $\overline{M}'$
		verify $e(\overline{M} + \overline{M}', P) \overset{?}{=} e(\overline{K}, \eta P)$
		run $\mathsf{RevocationCheck}(\overline{K}, \overline{M})$

Fig. 2. Self-blindable credential scheme by Kiyomoto et al. [11] (simplified).

This scheme suffers from a number of problems. First, the relation $k = \kappa + m\kappa$ is nowhere enforced by the $\mathsf{ShowCredential}$ protocol, in the sense that the user could use $\lambda, k - \lambda$ for some random $\lambda \in \mathbb{Z}_p$ instead of κ, κ'. This means that users can easily share credentials after all, without fear of disclosing the valuable information encoded by m.

Second, without going into the details of the revocation mechanism, we remark that it relies on how k splits into $k = \kappa + \kappa'$, so that the problem above allows users to present revoked credentials without problems. (In addition, the revocation mechanism introduces linkability.)

Third, since both the public key K and signature S are linear in both the blinding factor α and private key k, by Corollary 4.2 the scheme is 1-malleable. For any α and valid credential $((\kappa, \kappa'), K, S)$ the user can present the credential $((\lambda, \alpha k - \lambda), \alpha K, \alpha S)$. (Actually, because the public key $A = aP \in G$ lives in the same group as the signatures $S = akP \in G$, anyone can easily create his own credential by setting $K = (\kappa + \kappa')P =: kP$ for some random $\kappa, \kappa' \in \mathbb{Z}_p$, and $S = kA$ – that is, the system is actually 0-malleable.)

Example 5.3. Some of the problems of the credential scheme above were pointed out by Emura et al. [7], who came up with an improved protocol that we will examine in this example. In this protocol the malleability is solved through the use of Boneh-Boyen signatures. Theorem 4.1 shows, however, that it is linkable. We explain the problem here.

User	**Verifier**
	choose nonce $\eta \in_R \mathbb{Z}_p$
into $N \longleftarrow$	send ηQ
choose blinding $\alpha \in_R \mathbb{Z}_p$	
send $\alpha S, \alpha k N, \alpha A, \alpha N, \alpha P \longrightarrow$	into $\overline{S}, \overline{K}, \overline{A}, \overline{Q}, \overline{P}$
	verify $e(\overline{P}, A) = e(P, \overline{A})$
	verify $e(\overline{P}, \eta Q) = e(P, \overline{Q})$
	verify $e(\overline{S}, \eta \overline{A} + \overline{K}) = e(\overline{P}, \overline{Q})$
	run $\mathsf{RevocationCheck}(\overline{A}, \overline{K}, \overline{Q})$

Fig. 3. Self-blindable credential scheme by Emura et al. [7] (simplified).

The $\mathsf{ShowCredential}$ protocol is shown in Figure 3. As in Example 5.1, the Boneh-Boyen signature is of the form

$$\left(A, P, Q, S = \frac{1}{a+k}P\right);$$

we include the values A, P and Q explicitly in the signature because these are blinded as well in the $\mathsf{ShowCredential}$ protocol. The blinding factor is (η, α), where η is chosen by the verifier and α by the user. The blinded signature is then $(\alpha A, \alpha P, \alpha\eta Q, \alpha S)$, while the blinded public key is $\alpha\eta K$.

Theorem 4.1 is directly applicable to this scheme; we now describe the resulting linkability attack. Suppose the $\mathsf{ShowCredential}$ protocol is executed twice, and let (η_i, α_i) be the blinding factors used in two runs of the $\mathsf{ShowCredential}$ protocol, for $i = 1, 2$. Let $\overline{A}_i, \overline{P}_i, \overline{Q}_i, \overline{S}_i, \overline{K}_i$ be the values that the user sends to the issuer. We take the sum of two traces as follows:

$$\begin{aligned}
\overline{A} &= \eta_1 \overline{A}_1 + \eta_2 \overline{A}_2 = (\alpha_1\eta_1 + \alpha_2\eta_2)A, \\
\overline{P} &= \overline{P}_1 + \overline{P}_2 = (\alpha_1 + \alpha_2)P, \\
\overline{Q} &= \overline{Q}_1 + \overline{Q}_2 = (\alpha_1\eta_1 + \alpha_2\eta_2)Q, \\
\overline{S} &= \overline{S}_1 + \overline{S}_2 = \alpha_1 S_1 + \alpha_2 S_2, \\
\overline{K} &= \overline{K}_1 + \overline{K}_2 = \alpha_1\eta_1 K_1 + \alpha_2 K_2.
\end{aligned} \tag{3}$$

Now we put these values in the third verification equation as follows:

$$e(\overline{S}, \overline{A} + \overline{K}) \stackrel{?}{=} e(\overline{P}, \overline{Q}). \tag{4}$$

If $K_1 = K_2$, then also $S_1 = S_2$ holds, and the lower two equations of (3) become $\overline{S} = (\alpha_1 + \alpha_2)S$, $\overline{K} = (\alpha_1\eta_1 + \alpha_2\eta_2)K$. Then equation (4) will hold. On the other hand, if $K_1 \neq K_2$ then equation (4) will not hold. Thus, transactions are linkable by the third verification equation.[6]

[6] Note, however, that only the verifier can calculate the element $\overline{A} = \eta_1 \overline{A}_1 + \eta_2 \overline{A}_2$ (which is needed in order to perform this linking attack), as it contains η_1, η_2 which are never sent to the user. This differs from the linkability described in Example 5.1, in which anyone that can eavesdrop on the communication between the user and verifier can execute the attack. On the other hand, transactions

6 Can unmalleable, unlinkable self-blindable credential schemes exist?

Let us briefly consider a number of ways in which the pitfall outlined by Theorem 4.1 might be avoided. Suppose first that both the public key and the signature are linear in the private key, and that the sum of a trace of the ShowCredential protocol again constitutes a valid public key and signature. Then this can only be abused by a malicious user if he is able to calculate the corresponding private key. Therefore, if this is not feasible (perhaps because the private key k can only be calculated by the issuer, or because not all private keys are valid or allowable), then the system would not be malleable in the sense of Definition 3.3.

As another approach, one might take a public key and signature scheme that are both nonlinear in the private key k, or both nonlinear in the blinding factor α. In that case neither of the statements of Theorem 4.1 would be applicable. Going further, the ShowCredential protocol may be such that it is not necessary to send the public key to the verifier at all, so that it can play no role in either linkability or malleability. (This approach is taken in Idemix; see Example 6.1 below. For this reason, as well as the fact that Idemix does not satisfy Definition 3.1, we do not consider Idemix to be self-blindable.)

Example 6.1. The Idemix credential scheme [4,10] is an attribute-based credential scheme which is neither linkable nor malleable, and indeed, Proposition 4.1 does not apply to Idemix. This is because the ShowCredential protocol is substantially different from the ones of the other schemes discussed so far. In short, it goes as follows: the user partially blinds the Camenisch–Lysyanskaya [5] signature (A, e, v), resulting into $(\overline{A}, e, \overline{v})$, and sends $\overline{A}$ to the verifier. After that, they engage in an interactive zero knowledge-proof in which the user shows that he knows e, $\overline{v}$, and his private key, without disclosing any of these. This has the following consequences:

– There is no clear separation between the sending and verification of the public key and signature on the one hand, and a proof of knowledge of the secret key on the other hand. Both of these happen in a single interactive algorithm.
– In fact, the user does not directly send the public key to the verifier at all, blinded or otherwise. As a result, the map PubKey does not play any role in the ShowCredential algorithm.
– The map Sig_{SK} is not linear in the blinding factor.

Summarizing, we do not believe that it would be impossible to create self-blindable credential schemes that are unlinkable and unmalleable, but since the margin for error seems to be small, getting it right may be difficult. Such systems would certainly be useful and interesting, however, so we would not discourage further research in this direction.

7 Conclusion

Creating a self-blindable credential scheme which is neither malleable nor linkable is hard, and indeed all self-blindable credential schemes that we have studied are broken. There is a common theme in their failures: the use of the verification equation of the signature scheme in the ShowCredential protocol may cause linkability or malleability. We believe that this observation in the

can also be linked by checking the following equation:

$$e(\alpha_1 P, \alpha_2 S_2) = e(\overline{P}_1, \overline{S}_2) \overset{?}{=} e(\overline{P}_2, \overline{S}_1) = e(\alpha_2 P, \alpha_1 S_1)$$

which will hold if and only if $S_1 = S_2$; that is, when the signatures are the same. This attack can be done by any eavesdropper.

form of Theorem 4.1 and Corollary 4.2, together with the examples showing the consequences of this observation, will be of help in the creation of new, secure and anonymous self-blindable credential schemes.

References

1. Blake, I.F., Seroussi, G., Smart, N.P. (eds.): Advances in Elliptic Curve Cryptography. Cambridge University Press (2005), cambridge Books Online
2. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. J. Cryptology 21(2), 149–177 (2008)
3. Brands, S.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press (2000)
4. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) Advances in Cryptology - EUROCRYPT 2001. Lecture Notes in Computer Science, vol. 2045, pp. 93–118. Springer (2001)
5. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Cimato, S., Galdi, C., Persiano, G. (eds.) Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers. Lecture Notes in Computer Science, vol. 2576, pp. 268–289. Springer (2002)
6. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) Advances in Cryptology - CRYPTO '92. Lecture Notes in Computer Science, vol. 740, pp. 89–105. Springer (1993)
7. Emura, K., Miyaji, A., Omote, K.: A certificate revocable anonymous authentication scheme with designated verifier. In: Proceedings of the The Forth International Conference on Availability, Reliability and Security, ARES 2009, March 16-19, 2009, Fukuoka, Japan. pp. 769–773. IEEE Computer Society (2009)
8. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for cryptographers. Discrete Applied Mathematics 156(16), 3113–3121 (2008)
9. Hoepman, J.H., Jacobs, B., Vullers, P.: Privacy and security issues in e-ticketing – Optimisation of smart card-based attribute-proving. In: Cortier, V., Ryan, M., Shmatikov, V. (eds.) Workshop on Foundations of Security and Privacy, FCS-PrivMod 2010, Edinburgh, UK, July 14-15, 2010. Proceedings (July 2010)
10. IBM Research Zürich Security Team: Specification of the Identity Mixer cryptographic library, version 2.3.4. Tech. rep., IBM Research, Zürich (feb 2012)
11. Kiyomoto, S., Tanaka, T.: Anonymous attribute authentication scheme using self-blindable certificates. In: IEEE International Conference on Intelligence and Security Informatics, ISI 2008, Taipei, Taiwan, June 17-20, 2008, Proceedings. pp. 215–217. IEEE (2008)
12. Paquin, C., Zaverucha, G.: U-prove cryptographic specification v1.1 (revision 3) (December 2013), `http://research.microsoft.com/apps/pubs/default.aspx?id=166969`, released under the Open Specification Promise
13. Verheul, E.R.: Self-blindable credential certificates from the weil pairing. In: Boyd, C. (ed.) Advances in Cryptology - ASIACRYPT. Lecture Notes in Computer Science, vol. 2248, pp. 533–551. Springer (2001)