Pseudonymous Signature on eIDAS Token – Implementation Based Privacy Threats

Mirosław Kutyłowski(✉), Lucjan Hanzlik, and Kamil Kluczniak

Faculty of Fundamental Problems of Technology,
Politechnika Wrocławska, Wrocław, Poland
{miroslaw.kutylowski,lucjan.hanzlik,kamil.kluczniak}@pwr.edu.pl

Abstract. We investigate eIDAS Token specification for Pseudonymous Signature published recently by German security authority BSI, German Federal Office for Information Security. We analyze how far the current specification prevents privacy violations by the Issuer by malicious or simply careless implementation. We find that, despite the declared design goal of protecting privacy of the citizens, it is quite easy to convert the system into a "Big Brother" system and enable spying the citizens by third parties.

We show that there is a simple and elegant way for preventing all attacks of the kind described. Moreover, we show that it is possible with relatively small amendments to the scheme.

1 Introduction

Personal identity documents are more and more frequently equipped with an electronic layer. The primary goal of this layer was to prevent forgeries by providing key data digitally signed by the document issuer. However, there is an opportunity to use it for e-services such as authentication on a (remote) terminal. This has attracted a lot of attention recently, see the eIDAS regulation of European Union [6]. It aims to create common trust levels and fundamental mechanisms enabling interoperability of authentication services. It supports many novel services and features, including use of pseudonyms.

Privacy-by-design principle introduced by new personal data protection law is another driving force in Europe. It says that the information processing systems must be based on technical security (the former approach was based on penalties for unauthorized data processing).

Privacy Protection via Unlinkability. One of the ideas to achieve privacy-by-design is to eliminate unnecessary data disclosure via authentication. In the traditional setting we authenticate ourselves with full identity and then our rights

This research has been supported by the Polish National Science Centre, project HARMONIA, DEC-2013/08/M/ST6/00928.

J.K. Liu and R. Steinfeld (Eds.): ACISP 2016, Part II, LNCS 9723, pp. 467–477, 2016.
DOI: 10.1007/978-3-319-40367-0_31

are determined based on this identity. In many cases a pseudonymous identity would be enough. However, it is not just replacing the regular identity with a pseudonymous one. The problem is that:

- in many cases the user must not be able to appear under two pseudonyms in the same system (i.e. Sybil attacks must be impossible),
- user's activities in different systems must not be linkable – the colluding systems cannot link the pseudonyms of the same person.

Restricted Identification [5] is a mechanism that aims to replace the insecure login-password mechanism and has been implemented on the German personal identity card. It creates a unique password for each *sector* in a strong cryptographic way.

Pseudonymous Signature. This is one of the mechanisms on the eIDAS Token, which has been designed presumably as a replacement for Restricted Identification. It has certain advantages:

- it does not enable to impersonate a user by an adversary knowing a so-called *group key* (see [7]),
- it enables Chip Authentication in a way that creates an undeniable evidence for later disputes.

There are also some disadvantages:

- the last property can be regarded as a disadvantage as well. Previously simultability was frequently declared as a strong privacy protection feature – an authentication proof was not transferable and therefore useless for illegal data trade,
- the seclusiveness problem has not been solved so far.

The Problem. There are two critical security assumptions behind the design of [5]: the eID chips are tamper resistant,and the Issuer of eID is trustworthy. The first assumption is critical in the sense that it is not known how to improve the scheme to make it immune against chip compromise. Some authors provide the same functionality resistant to compromise of eID chips (see e.g. [4] or [8]), but completely new protocols are used (with other disadvantages, like use of pairings). In this paper we focus on the second assumption and ask *how secure are the citizens using eIDAS token from* [5] *in case of rogue authorities?*

Even if in many cases the authorities and manufacturers are trustworthy, the eIDAS token solution might become an international standard. Therefore we cannot exclude an application of this technology in case where the Issuer cannot be trusted.

Paper Overview. In Sect. 2 we recall the technical specification of Pseudonymous Signature from [5]. In Sect. 3 we present some scenarios for rogue implementation of the Issuer in such a way that not only the Issuer can deanonymize users, but also may delegate these possibilities to third parties without giving the private keys of the users. In Sect. 4 we show a relatively simple and elegant solution to prevent all attacks of this kind in a way compliant with the specification of Pseudonymous Signatures from [5].

2 Pseudonymous Signature on eIDAS Token

Here we recall the Pseudonymous Signature from [5]. We follow the notation from [5] in order to make a direct reference to this de facto standard.

System Setup. The system is based on a cyclic group $\mathcal{G}$ of a prime order p (the specification also refers to EC groups). Let g denote a fixed generator of $\mathcal{G}$. There is a pair of keys: the secret key SK_M and the matching public key $PK_M = g^{SK_M}$.

Group Setup. For a group of eID documents the Issuer uses a pair of keys: a secret key SK_{ICC} and the public key $PK_{ICC} = g^{SK_{ICC}}$. The size of a group is a compromise between the size of anonymity set (the number of eIDs based on the same PK_{ICC}) and the cost of revocation of all eIDs using PK_{ICC} in case of leaking SK_{ICC}.

Domain Setup. For a domain *sector* there is a public key PK_{sector} generated by a trusted third party. For application scenarios requiring that the trusted authority can be asked to deanonymize a domain pseudonym of a user, the trusted authority generates PK_{sector} as $g^{SK_{sector}}$. Otherwise, "the third party SHALL generate Sector Public Keys in a way that the corresponding private keys are unknown". A common way to achieve this is to create PK_{sector} via a hash function from the domain identifier.

Issuing an ID Document. For the sake of Pseudonymous Signatures of user U the Issuer generates at random a key $SK_{ICC,2,U} < p$[1]. The second private key $SK_{ICC,1,U}$ is

$$SK_{ICC} := SK_{ICC,1,U} + SK_M \cdot SK_{ICC,2,U} \bmod p$$

The corresponding public keys $PK_{ICC,1,U} = g^{SK_{ICC,1,U}}$ and $PK_{ICC,2,U} = g^{SK_{ICC,2,U}}$ might be stored by the Issuer for the sake of deanonymization. The keys $SK_{ICC,1}, SK_{ICC,2}$ are stored on the eID document. (The keys $PK_{ICC,1}, PK_{ICC,2}$ need not to be stored there.)

Creation of Pseudonyms for a Domain. An eID document holding the private keys $SK_{ICC,1,U}$, $SK_{ICC,2,U}$ creates the pseudonyms for the sector with the public key PK_{sector}:

$$I^{sector}_{ICC,1,U} := PK^{SK_{ICC,1,U}}_{sector} \quad \text{and} \quad I^{sector}_{ICC,2,U} := PK^{SK_{ICC,2,U}}_{sector} \ .$$

[1] We change the notation from [5] and indicate explicitly the key owner.

Creation of a Pseudonymous Signature for a Domain. (We present a simplified version without some irrelevant implementation details.)

The following steps are executed by user M for signing a message M:

1. choose k_1, k_2 at random,
2. compute $Q_1 := g^{k_1} \cdot PK_M^{k_2}$,
3. [optional] compute $A_1 := PK_{sector}^{k_1}$,
4. [optional] compute $A_2 := PK_{sector}^{k_2}$,
5. compute $c := \mathrm{Hash}(Q_1, I_{ICC,1,U}^{sector}, A_1, I_{ICC,2,U}^{sector}, A_2, PK_{sector}, \mathrm{params}, M)$. (the parameters $I_{ICC,1,U}^{sector}, A_1$ and $I_{ICC,2,U}^{sector}, A_2$ are optional and omitted when, respectively, A_1 or A_2 are not computed). The argument *params* stands for some additional parameters which are not important from our point of view.
6. compute $s_1 := k_1 - c \cdot SK_{ICC,1,U} \bmod p$ and $s_2 := k_2 - c \cdot SK_{ICC,2,U} \bmod p$.
7. output the signature (c, s_1, s_2).

Signature Verification. Given a signature (c, s_1, s_2), the pseudonyms $I_{ICC,1,U}^{sector}$, $I_{ICC,2,U}^{sector}$ are to be attached, if A_1 and, respectively, A_2 have been used for signature creation.

The verification procedure looks as follows:

1. recompute Q_1 as $Q_1' := PK_{ICC}^{c} \cdot g^{s_1} \cdot PK_M^{s_2}$,
2. [optional] recompute A_1 as $A_1' := (I_{ICC,1,U}^{sector})^c \cdot PK_{sector}^{s_1}$,
3. [optional] recompute A_2 as $A_2' := (I_{ICC,2,U}^{sector})^c \cdot PK_{sector}^{s_2}$,
4. recompute c as $c' := \mathrm{Hash}(Q_1', I_{ICC,1,U}^{sector}, A_1', I_{ICC,2,U}^{sector}, A_2', PK_{sector}, \mathrm{params}, M)$ (if some arguments are omitted during signature creation, then the same arguments should be omitted here).
5. accept if $c' = c$.

Note that the verification will yield the positive result, if the signer follows the protocol:

$$
\begin{aligned}
Q_1' &= PK_{ICC}^{c} \cdot g^{s_1} \cdot PK_M^{s_2} = PK_{ICC}^{c} \cdot g^{k_1 - c \cdot SK_{ICC,1,U}} \cdot PK_M^{k_2 - c \cdot SK_{ICC,2,U}} \\
&= g^{k_1} \cdot PK_M^{k_2} \cdot \left(PK_{ICC} \cdot g^{-SK_{ICC,1,U}} \cdot PK_M^{-SK_{ICC,2,U}} \right)^c \\
&= Q_1 \cdot \left(g^{SK_{ICC}} \cdot g^{-(SK_{ICC,1,U} + SK_M \cdot SK_{ICC,2,U})} \right)^c = Q_1 \cdot 1^c = Q_1 \ . \\
A_1' &= (I_{ICC,1,U}^{sector})^c \cdot PK_{sector}^{s_1} = PK_{sector}^{c \cdot SK_{ICC,1,U}} \cdot PK_{sector}^{k_1 - c \cdot SK_{ICC,1,U}} = PK_{sector}^{k_1} = A_1 \ , \\
A_2' &= (I_{ICC,2,U}^{sector})^c \cdot PK_{sector}^{s_2} = PK_{sector}^{c \cdot SK_{ICC,2,U}} \cdot PK_{sector}^{k_2 - c \cdot SK_{ICC,2.U}} = PK_{sector}^{k_2} = A_2 \ .
\end{aligned}
$$

Differences with the Protocol from [2]. The version presented in [2] is the protocol described above with the following choice of options[2]:

[2] The description of `NymVf` contains a misprint: y should be replaced by g_2, which corresponds to PK_M in [5].

- the optional parameters $I^{sector}_{ICC,1,U}, A_1$ are used,
- the optional parameters$I^{sector}_{ICC,2,U}, A_2$ are not used,
- the discrete logarithm of PK_{sector} is always known to the Issuer.

For the protocol described in [2] certain security proofs have been given (there are some problems with them [9]). The recommendation [5] contains neither formal security proofs nor a design rationale.

3 Rogue Issuing Authority

The main purpose of Pseudonymous Signature is to protect signer's privacy. Definitely, we have to trust the Issuer, as according to [5] it creates the secret keys of each single user. The Issuer can retain these keys and use later to deanonymize the users. A silent assumption in [5] as well as in [2,3] is that this is inevitable. In Sect. 4 we show that this is not the case as we can secure the users against the Issuer.

The main problem that we discuss in this section is "delegation" of the ability to deanonymize the users. Is it easy to reveal some data to a third party, called Tracer, so that it can deanonymize as well? The volume of data forwarded to the Tracer counts very much, since the leakage can be created by rogue software installed by the honest Issuer, who himself becomes a victim of the attack: it is much easier to leak a few keys than to hand over the whole database.

In certain situations the Issuer might be forced to provide deanonymization tools to the Tracer. In this case it is important to limit the possibilities of the Tracer. For instance, it should be impossible for the Tracer to create valid secret keys for new users or to forge signatures of the existing users.

The situation described above may concern the state authorities: the Issuer of a country A might be forced to provide deanonymization tools for the security authorities of a country B due to political dependence or in course of trading secrets. However, we have to be aware that a leakage may also concern data transfer to the organized crime. This is particularly dangerous, since the signers may falsely assume that their anonymity is well protected, while it might be not true in case of their biggest foes. Protection against authorities should also be considered. For instance, if Pseudonymous Signatures are used for the sake of electronic voting, some regimes might be tempted to deanonymize the voter supporting the opposition.

Below we show methods for tracing the users of Pseudonymous Signatures.

3.1 Scenario 1: The Issuer Creates Users' Private Keys according to the Protocol

The protocol description in [5] says that the user may authenticate himself with only one pseudonym (or none of them). First let us make the following observation:

Proposition 1. *Assume that the Tracer knows SK_M and holds at least one identity document. Then given one pseudonym of a user U in a domain, he can compute the second pseudonym of U in this domain.*

Proof. First the Tracer can compute $PK_{sector}^{SK_{ICC}}$. Namely, he generates own pseudonyms $I_{ICC,1,T}^{sector}$, $I_{ICC,2,T}^{sector}$ and computes $I_{ICC,1,T}^{sector} \cdot (I_{ICC,2,T}^{sector})^{SK_M}$. Note that

$$I_{ICC,1,T}^{sector} \cdot (I_{ICC,2,T}^{sector})^{SK_M} = (PK_{sector})^{SK_{ICC,1,T}} \cdot (PK_{sector})^{SK_{ICC,2,T} \cdot SK_M} = (PK_{sector})^{SK_{ICC}}$$

Now, given the pseudonym $I_{ICC,1,U}^{sector}$, the Tracer can derive $I_{ICC,2,U}^{sector}$ as

$$(PK_{sector})^{SK_{ICC}} / I_{ICC,1,U}^{sector})^{SK_M^{-1} \bmod p}$$

Similarly, one can derive $I_{ICC,1,U}^{sector}$ from $I_{ICC,2,U}^{sector}$ as

$$(PK_{sector})^{SK_{ICC}} / (I_{ICC,2,U}^{sector})^{SK_M}.$$

□

By Proposition 1 separation of user's signatures based on the pseudonym $I_{ICC,1,U}^{sector}$ and the signatures based on the pseudonym $I_{ICC,2,U}^{sector}$ is not strict, even if the user never creates signatures based on both pseudonyms.

Remark 1. The proof does not work if we replace SK_M by SK_{ICC} in Proposition 1.

It seems that in order to trace a user U, the Issuer has to give the Tracer either $SK_{ICC,1,U}$ or $SK_{ICC,2,U}$. Since the key $SK_{ICC,2,U}$ has to be chosen at random, the Issuer has to leak the key separately for each user. This is somewhat difficult, leaking a single secret key is much easier, e.g. it can be copied to a piece of paper and taken away.

Note that revealing both private keys for 2 different users would mean revealing the system keys SK_M and SK_{ICC} and thereby would delegate the right to issue eID documents as well – which is perhaps much more than the Issuer might agree upon. Unfortunately, it is hard to exclude that the Tracer has broken two different identity documents and therefore was able to derive SK_M and SK_{ICC}. In this case leaking one of the keys $SK_{ICC,1,U}$ and $SK_{ICC,2,U}$ is enough to leak both keys. Then the Tracer would be able to impersonate a given user as well. So this kind of leakage is probably unacceptable for the Issuer.

3.2 Scenario 2: The Issuer Creates the Users' Private keys with a PRNG

Generation of private keys by the Issuer can be implemented in the following way. The Issuer holds a secret random seed s for a cryptographically

secure Pseudorandom Number Generator (PRNG) creating numbers in the range $[0, p-1]$. Then the Issuer computes $SK_{ICC,2,U} := \mathrm{PRNG}(s, \mathrm{ID}_U)$, where ID_U is the identifier of U.

Note that having s alone enables the third party to recompute $SK_{ICC,2,U}$ for each user U and thereby to compute the second pseudonym $I^{sector}_{ICC,2,U} = PK^{SK_{ICC,2,U}}_{sector}$ of U in the sector with the public key PK_{sector}.

An implementation based on Scenario 2 can be well justified. Namely, it eliminates problems related to weak sources of randomness. (Note that if the randomness is weak and $SK_{ICC,2,U}$ predictable in some sense, then a party knowing the weakness can extract the candidate keys $SK_{ICC,2,U}$ and check them against the pseudonyms.) Deploying an PRNG is also recommended by NIST [10] – no nondeterministic RNG is recommended for use (of course, the FIPS specification of DRNG requires input of entropy bits, but an external observer cannot test whether these entropy bits are really used).

Such a scenario is still problematic, as the Tracer getting s can compute $SK_{ICC,2,U}$ for any user U. A much better choice would be to enable to trace selectively some users.

3.3 Scenario 3: $SK_{ICC,1,U}$ and $SK_{ICC,2,U}$ with a hidden relationship.

For each user U there are parameters x_U and s_U generated in pseudorandom way. Then

$$\begin{cases} x_U &= SK_{ICC,1,U} + SK_{ICC,2,U} \cdot s_U \mod p, \\ SK_{ICC} &= SK_{ICC,1,U} + SK_{ICC,2,U} \cdot SK_M \mod p, \end{cases} \tag{1}$$

The service dependent trapdoor is $T_{sector,U} = PK^{x_U}_{sector}$. The Tracer gets $T_{sector,U}$ and s_U from the Issuer in order to trace the user U in this sector. The test is:

$$T_{sector,U} \stackrel{?}{=} I^{sector}_{ICC,1,U} \cdot (I^{sector}_{ICC,2,U})^{s_U}$$

Note that even if the Tracer learns SK_M, SK_{ICC}, s_U and $T_{sector,U}$, then he still cannot solve the above system of linear Eq. (1) as there are three unknowns: $SK_{ICC,1,U}$, $SK_{ICC,2,U}$ and x_U (note that x_U cannot be extracted from $T_{sector,U}$).

The question is whether additional input would ease forging pseudonymous signatures. This seems not to be the case by the following argument:

> given an instance – an input given to an adversary in a standard case, then the adversary can choose an a at random, put
>
> $$T_{sector,U} := I^{sector}_{ICC,1,U} \cdot (I^{sector}_{ICC,2,U})^{a}$$

and perform the attack using such $T_{sector,U}$. There are two cases: If for $T_{sector,U}$ constructed in this way the attack yields noticeably different results than in the real case, then we can easily build a distinguisher between the output of the PRNG and random numbers. Of course, if we apply a good PRNG, this should not be the case. The other option is that the attack based on such $T_{sector,U}$ works like for the real case. So we see that if it is possible to mount a forgery based on enhanced data, then we can mount a similar attack for the regular case.

Remarks. Note that the leakage could be selective (the Issuer betrays s_U, x_U for some users) or a global one (the Issuer betrays the secret seed s for all of them). Moreover, we may arrange the process of creating the secrets s_U in a tree-like fashion so that one can betray only the secrets from a subtree.

The above attack does not work for the former version described in [2,3], as in this case $I^{sector}_{ICC,2,U}$ is not available.

Note that the Tracer cannot learn the pseudonym of a user in a sector, if the user does not create it. The capability of the Tracer seems to be limited to deanonymization of the users which are active in a sector.

3.4 Scenario 4: Tracing with One Pseudonym

The attacks described above require both domain pseudonyms to deanonymize a user. So one may hope that if we retreat to the setting from [2], then we are again secure against deanonymization attacks enabled by a rogue Issuer. Unfortunately, we show that this is not the case.

In order to enable tracing a user U, the Tracer gets a special *shadow eID*, say for a user U'. Namely, the Issuer creates $SK_{ICC,1,U'}$, $SK_{ICC,2,U'}$ so that:

$$\begin{cases} SK_{ICC,1,U} = s_U \cdot (SK_{ICC,1,U'})^2 & \mathrm{mod} p \ , \\ SK_{ICC} \quad\ = SK_{ICC,1,U'} + SK_{ICC,2,U'} \cdot SK_M & \mathrm{mod} p \ . \end{cases} \tag{2}$$

Now, given PK_{sector}, the user U' can compute the pseudonym $I^{sector}_{ICC,1,U}$ of the user U in the following way:

1. compute its own pseudonym $I'' = I^{sector}_{ICC,1,U'}$ for PK_{sector},
2. compute $I' := (I'')^{s_U}$,
3. feed own eID with I' as the public key of a sector, consequently the eID returns $I = (I')^{SK_{ICC,1,U'}}$,
4. output I.

Note that the output is correct, since

$$(I')^{SK_{ICC,1,U'}} = PK_{sector}^{s_U \cdot (SK_{ICC,1,U'})^2} = PK_{sector}^{SK_{ICC,1,U}} = I^{sector}_{ICC,1,U} \ .$$

In the above procedure the role of s_U is to prevent detection that the eID U' is rogue. Indeed, for $s_U = 1$ an inspector holding the eID of user U' could run the above procedure and check the results. The secret s_U guarantees that such an inspection is infeasible - the holder of eID U' may deny to know any such secret.

On the other hand, even if $I^{sector}_{ICC,1,U'}$ and s_U are known, it is infeasible to compute $PK_{sector}^{s_U \cdot (SK_{ICC,1,U'})^2}$ without knowing $SK_{ICC,1,U'}$. Indeed, this is equivalent to Square Diffie-Hellman problem, which is equivalent to CDH [1]. So if the shadow user U' is behaving in a regular way, it is infeasible to derive the pseudonym of U.

It is also worth to note that $SK_{ICC,1,U'}$ can be chosen at random – then s_U is derived as $SK_{ICC,1,U}/(SK_{ICC,1,U'})^2$. So the probability distribution of the keys for the shadow user U' is the same as for the case when it is not used for tracing U. Of course, U' can trace many users: the Issuer gives U' the secret s_U for each traced user U.

4 Protection Against Rogue Issuers

If the Issuer creates the users' secret keys, we cannot exclude leaking them. Therefore, the only really effective solution would be to prevent the Issuer to know the private keys of the users. Below we propose a method that achieves this goal.

Secure Setup of Pseudonymous Signatures.
Secure initialization of the eID of a user U consists of the following steps:

1. After manufacturing time the eID chip stores two pairs of *prekeys*: $(x_{1,1}, x_{2,1})$ and $(x_{1,2}, x_{2,2})$. They satisfy the equations
   $SK_{ICC} = x_{1,i} + x_{2,i} \cdot SK_M \qquad \text{for } \ i = 1, 2.$
2. The eID document reaches the user U in the *initialization mode*. In the first step the eID document presents the following *pre-identifiers* to the document owner U:
$$IN_{1,1} = g^{x_{1,1}}, \quad IN_{2,1} = g^{x_{2,1}}, \quad IN_{1,2} = g^{x_{1,2}}, \quad IN_{2,2} = g^{x_{2,2}} \ .$$
3. The eID document owner U chooses a, b such that $a + b = 1 \bmod p$ and presents them to the eID document. Thereby he requests the eID chip to hold
$$\begin{aligned} SK_{ICC,1,U} &:= a \cdot x_{1,1} + b \cdot x_{1,2} \bmod p \ , \\ SK_{ICC,2,U} &:= a \cdot x_{2,1} + b \cdot x_{2,2} \bmod p \end{aligned}$$
as the private key for Pseudonymous Signature. Note that
$$\begin{aligned} SK_{ICC,1} + SK_{ICC,2} \cdot SK_M &= a \cdot x_{1,1} + b \cdot x_{1,2} + (a \cdot x_{2,1} + b \cdot x_{2,2}) \cdot SK_M \\ &= a \cdot (x_{1,1} + x_{2,1} \cdot SK_M) + b \cdot (x_{1,2} + x_{2,2} \cdot SK_M) \\ &= a \cdot SK_{ICC} + b \cdot SK_{ICC} = SK_{ICC} \ , \end{aligned}$$
so the derived private keys are correct. Also, for any y_1, y_2 satisfying $SK_{ICC} = y_1 + y_2 \cdot SK_M$, there is exactly one pair (a, b) such that
$$\begin{cases} y_1 = a \cdot x_{1,1} + b \cdot x_{1,2} \bmod p, \\ y_2 = a \cdot x_{2,1} + b \cdot x_{2,2} \bmod p, \end{cases} \tag{3}$$
and $a + b = 1 \bmod p$. Indeed,
$$\begin{aligned} \begin{vmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{vmatrix} &= \begin{vmatrix} x_{1,1} + x_{2,1} \cdot SK_M & x_{1,2} + x_{2,2} \cdot SK_M \\ x_{2,1} & x_{2,2} \end{vmatrix} \\ &= \begin{vmatrix} SK_{ICC} & SK_{ICC} \\ x_{2,1} & x_{2,2} \end{vmatrix} = SK_{ICC} \cdot (x_{2,2} - x_{2,1}) \neq 0 \bmod p \end{aligned}$$

so there are a and b that satisfy (3). Moreover,

$$SK_{ICC} = y_1 + SK_M \cdot y_2 = (a \cdot x_{1,1} + b \cdot x_{1,2}) + SK_M \cdot (a \cdot x_{2,1} + b \cdot x_{2,2})$$
$$= a \cdot (x_{1,1} + SK_M \cdot x_{2,1}) + b \cdot (x_{1,2} + SK_M \cdot x_{2,2}) = a \cdot SK_{ICC} + b \cdot SK_{ICC} \bmod p$$

Hence $a + b = 1 \bmod p$. Finally we may conclude that the key pair $(SK_{ICC,1,U}, SK_{ICC,2,U})$ is uniformly distributed in the set of all private key pairs.

4. For the future use the eID document owner retains

$$I_1 := IN_{1,1}^a \cdot IN_{1,2}^b, \quad I_2 := IN_{2,1}^a \cdot IN_{2,2}^b \ .$$

5. At this moment the eID document erases the pre-keys, the initialization procedure terminates and the eID document can create pseudonymous signatures with the keys $SK_{ICC,1,U}$, $SK_{ICC,2,U}$.

Anytime the user U can test whether the keys $SK_{ICC,1,U}$, $SK_{ICC,2,U}$ are really used by his eID document. For this purpose the user U asks for identifiers for a sector with $PK_{sector} = g^h$, where h is known to him. The pseudonyms $I_{ICC,1}^{sector}$, $I_{ICC,2}^{sector}$ returned by the eID chip should satisfy the following equalities:

$$I_{ICC,1,U}^{sector} = PK_{sector}^{SK_{ICC,1,U}} = PK_{sector}^{a \cdot x_{1,1} + b \cdot x_{1,2}} = g^{h \cdot a \cdot x_{1,1}} \cdot g^{h \cdot b \cdot x_{1,2}} = IN_{1,1}^{h \cdot a} \cdot IN_{1,2}^{h \cdot b} = I_1^h$$
$$I_{ICC,2,U}^{sector} = PK_{sector}^{SK_{ICC,2,U}} = PK_{sector}^{a \cdot x_{2,1} + b \cdot x_{2,2}} = g^{h \cdot a \cdot x_{2,1}} \cdot g^{h \cdot b \cdot x_{2,2}} = IN_{2,1}^{ha} \cdot IN_{2,2}^{hb} = I_2^h$$

So the document owner performs the test

$$I_{ICC,1}^{sector} \stackrel{?}{=} I_1^h \quad \text{and} \quad I_{ICC,2}^{sector} \stackrel{?}{=} I_2^h \tag{4}$$

If the test fails, then the eID chip is cheating about the choice of the private key.

The eID chip may attempt to guess the moment of the test. However, this would be equivalent to guessing whether the document owner knows the discrete logarithm of the element presented as the public key of a sector. Since deanonymization requires that somebody knows this discrete logarithm, it is infeasible to demand from the owner a proof that he does not know the discrete logarithm.

Note that the above method works also for the original scheme from [3]. Then the test concerns only one equality.

The only problem with the above approach is that it precludes deanonymization. In order to enable it, one can extend the protocol so that the life-cycle of an eID document consists of the configuration phase and the application phase. After the configuration phase the eID document enters the application phase and there is no way back to the configuration phase. The configuration phase consists of the following steps:

- generate the private keys $SK_{ICC,1,U}$, $SK_{ICC,2,U}$ as described above,
- generate the pseudonyms $P_1 = g^{SK_{ICC,1,U}}$, $P_2 = g^{SK_{ICC,2,U}}$ and a Pseudonymous Signature for $PK_{sector} = g$, send the pseudonyms and the signature to the Issuer over a secure channel,

- enter the application phase after receiving an acknowledgement of the Issuer confirming P_1 and P_2.

Given P_1, P_2, deanonymization may be executed as for the original eIDAS token [5].

5 Conclusions

Despite the careful design of [5], it turns out that some details of the specification need to be carefully reviewed. We need a complete system description with a corresponding security model taking into account malicious behavior of protocol participants. Potential mistakes may have deep impact, as decisions concerning electronic identity documents have their long term consequences due to the typical exchange period of 10 years.

References

1. Bao, F., Deng, R.H., Zhu, H.: Variations of Diffie-Hellman problem. In: Qing, S., Gollmann, D., Zhou, J. (eds.) ICICS 2003. LNCS, vol. 2836, pp. 301–312. Springer, Heidelberg (2003)
2. Bender, J., Dagdelen, Ö., Fischlin, M., Kügler, D.: Domain-specific pseudonymous signatures for the German identity card. IACR Cryptology ePrint Archive **2012**, 558 (2012)
3. Bender, J., Dagdelen, Ö., Fischlin, M., Kügler, D.: Domain-specific pseudonymous signatures for the German identity card. In: Gollmann, D., Freiling, F.C. (eds.) ISC 2012. LNCS, vol. 7483, pp. 104–119. Springer, Heidelberg (2012)
4. Bringer, J., Chabanne, H., Lescuyer, R., Patey, A.: Efficient and strongly secure dynamic domain-specific pseudonymous signatures for ID documents. In: Christin, N., Safavi-Naini, R. (eds.) FC 2014. LNCS, vol. 8437, pp. 252–269. Springer, Heidelberg (2014)
5. BSI: Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token 2.20. Technical Guideline TR-03110-2 (2015)
6. European Parliament the Council: Regulation (EU) No 910/2014 of the European Parliament and of the Council on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC (2014)
7. Hanzlik, L., Kutyłowski, M.: Insecurity of anonymous login with German personal identity cards. In: Security and Privacy in Social Networks and Big Data (SocialSec), pp. 39–43. IEEE Computer Society (2015)
8. Kluczniak, K.: Anonymous authentication using electronic identity documents. Ph.D. Dissertation, submitted (2016)
9. Kluczniak, K.: Domain-specific pseudonymous signatures revisited. IACR Cryptology ePrint Archive **2016**, 70 (2016)
10. NIST: Annex C: Approved random number generators for FIPS PUB 140–2, security requirements for cryptographic modules, January 2016. http://csrc.nist.gov/publications/fips/fips140-2/fips1402annexc.pdf