A Self-Certified and Sybil-Free Framework for Secure Digital Identity Domain Buildup

Christer Andersson[1], Markulf Kohlweiss[2], Leonardo A. Martucci[1] and Andriy Panchenko[3]

[1] Karlstads Universitet, Department of Computer Science
Universitetsgatan 2, 651-88 Karlstad, Sweden
[2] Katholieke Universiteit Leuven, ESAT/COSIC
Kasteelpark Arenberg, 10 B-3001 Leuven-Heverlee, Belgium
[3] RWTH Aachen University, Department of Computer Science
Informatik IV, Ahornstr. 55, D-52074 Aachen, Germany
{christer.andersson, leonardo.martucci}@kau.se, markulf.kohlweiss@esat.kuleuven.be, panchenko@cs.rwth-aachen.de

Abstract. An attacker who can control arbitrarily many user identities can break the security properties of most conceivable systems. This is called a "Sybil attack". We present a solution to this problem that does not require online communication with a trusted third party and that in addition preserves the privacy of honest users. Given an initial so-called Sybil-free identity domain, our proposal can be used for deriving Sybil-free unlinkable pseudonyms associated with other identity domains. The pseudonyms are self-certified and computed by the users themselves from their cryptographic long-term identities.

1 Introduction

Today, users often need to communicate and cooperate in networked environments. Virtual / Online communities, peer-to-peer systems, and applications for anonymous communication are only some prominent examples. Often, these systems depend on a majority of users being honest for tasks like voting in virtual community, reputation computation, Byzantine fault tolerance, or traffic mixing. Unless such systems implement expensive countermeasures they fall prey to the Sybil Attack [17], which entails a single attacker controlling arbitrarily many user accounts (called Sybil identities). Moreover, both identity certificates and the most advanced non-centralized Sybil defence mechanisms [23, 29] are privacy-invasive.

This paper defines *identity domains* as domains that uniquely specifies the context in which a set of identifiers is used. The purpose of a domain is to build an anonymity set i. e. a set of identifiers within an user is not identifiable. The context may include validity time, location, application, or other parameters. A secure identity domain should provide a Sybil-free environments (i. e., absent of Sybil identities) in which applications can be deployed. This paper shows how, given one initial Sybil-free identity domain, we can propagate the Sybil-freeness to arbitrary many identity domains. In every identity domain each user

is known under a different and unique pseudonym, and further there is no need of the continuous involvement of a Trusted Third Party (TTP). Access to a Certificate Authority (CA) is required only for the bootstrapping of a Sybil-free domain[4].

We call our solution *self-certified Sybil-free pseudonyms*[5]. These pseudonyms do not depend on the continuous availability of a TTP and, they are fully unlinkable. This is achieved using a self-certification mechanism: self-certified Sybil-free pseudonyms use concepts such as anonymous credentials and group signatures to enable the generation of an arbitrary number of anonymous certificates – however, only one certificate per identity domain and user identity. Access to the certificate authority (CA) is required only for acquiring the membership certificated from which the self-certified pseudonyms are derived from. Our solution can be seen as a framework that enables privacy-enhanced and Sybil-resistant buildup of user groups. We use periodic n-times spendable e-tokens [9] as a base for the instantiation, although there are also other cryptographic primitives that can be used to create such pseudonyms.

A user that wants to participate in the system first enrolls with the CA to acquire exactly one *membership certificate*. Thereby, we establish the initial Sybil-free identity domain. Using the certificate, the user can create one *self-certified pseudonym* per newly created identity domain. Membership certificates can be used for issuing pseudonyms for arbitrarily many identity domains, but the pseudonyms are only valid within the domain they were issued for. Further, pseudonyms issued for different identity domains are mutually unlinkable. Specifically, they cannot be linked to the underlying membership certificate even by the CA itself. The self-generated pseudonym certificates that come with self-certified pseudonyms provide three main functions: (*i*) binding of a freshly generated public key to the pseudonym (as with identity certificates); (*ii*) verification of the pseudonym and the binding; and (*iii*) disclosure of the user identity and revocation of her certificates, if the same membership certificate is used to create two different pseudonym certificates for the same domain.

This paper is organized as follows. The assumptions are presented in Section 2. Section 3 discusses applications that can benefit from our solution. The underlying cryptographic mechanism for our solution – n-spendable e-tokens – are introduced in Section 4. Then, the instantiation of our Sybil-free self-certified based on the e-tokens is described in Section 5. Finally, Section 6 concludes the paper.

---

[4] Identity domains can be constructed assuming continuous availability of a TTP. The problems with this approach are that the TTP can link all pseudonyms to the issuing user and the availability requirements on the TTP.

[5] Self-certified pseudonyms are also discussed in [2]. Whereas this paper presents a detailed description on their construction, the main focus of [2] is on their application in mobile ad hoc networks.

2 Assumptions

We assume that: (*i*) the CA is capable of establishing the initial Sybil-free domain[6]; (*ii*) identity domain identifiers ctx are unique; and, (*iii*) devices are capable of performing the necessary cryptographic functions. Regarding the attacker model, the attacker has two main goals: (*i*) deploy a Syibil attack in a given identity domain and (*ii*) identify a relationship between two pseudonyms generated for different identity domains to find out if those pseudonyms belong to a same user. We assume that attackers are able to eavesdrop all network communication, but each attacker has at most one membership certificate $cert_{\mathcal{U}}$.

A serious challenge regarding an eventual real life implementation of our system is the realization of the initial Sybil-free domains. Yet, this assumption is not exclusive for our scheme. In fact, for any scheme based on certificates, regardless of whether a fully distributed or a completely distributed security model is used (or something in between, such as a threshold scheme), some entity (or a cluster of several entities) *must* be trusted not to hand out more that one credential per identity (at least, it must be made very costly to obtain several credentials).

3 The Need for Sybil-Free Applications

The number of applications where a group of users interact electronically is endless: numerous instant messaging applications, chat rooms, forums and e-commerce platforms are only a few examples of widely used applications. Often, such applications allow users to slip into different roles, and behave accordingly. However, with growing size and sophistication of such communities and applications, the amount of required administration tasks grows: misbehaving users need to be excluded, user contributions need to be evaluated based on user reputation, and work tasks need to be distributed. In short, such applications and communities develop their own social dynamics, and there is a need to make decision processes work in a more automated way. For instance, such decisions could be based on majority voting, seniority, or reputation.

Truly anonymous or pseudonymous applications are currently debated, partly because they can enable misbehaving users to create social problems within their communities. Although these users can be banned from such applications, it is often easy for the wrongdoers to simply re-register using a new name. To change IP addresses using proxies or similar techniques is enough to thwart most existing countermeasures. Reputation systems also break under such an attack as users can register multiple times to collaboratively increase the reputation of all of their pseudonyms. Further, they can manipulate the allocation of resources and the distribution of work. Evil users can also choose names similar to other users

[6] This assumption is not exclusive for our scheme. In any certificate-based scheme, be it a fully centralized or completely distributed model (or something in between), some entity/entities *must* be trusted not to hand out more that one credential per identity.

to abuse their reputation. Finally, users that control multiple identities can more easily spread rumors and influence voting results to their own advantage.

Nonetheless, this separation between real world identities and different virtual worlds that allows the support of pseudonymous and anonymous users is a valued feature. As networks are "unforgetful" and may log and remember a close to infinite number of network interactions, this separation decreases the privacy risks associated with interacting in a computer network. Many scientific papers have been dedicated to various types of pseudonymity [4, 5, 19] or the graceful degradation of anonymity towards full identification [1] using existing approaches.

Numerous applications would benefit from having such a basic building block in place (even if the Sybil protection of the initial domain would only be approximate):

- *Resilient systems* require a majority of users to be honest in order to achieve Byzantine fault tolerance;
- *Peer-to-peer systems* need to manage reputation, some of which may rely on dummy e-currencies and distributed double-spending detection;
- *Online communities.* On platforms like eBay, a protection against self-ranking can be provided. Furthermore, if a user deletes his account and joins again (to get rid of "bad" reputation), both actions can be linked. Dating communities can be protected in the way that only one profile can be posted per physical user. Some online forums provide automatic banning of users if some fraction of the users vote for it. By using surveys, it is possible to make sure that a disjoint set of people was questioned;
- *Online multi-user games* need to be protected against cheaters. Privacy-friendly subscriptions with protection against sharing can be provided. Further, exclusion of bots from the game can be achieved;
- *Anonymous communication systems* require some portion of the users to be honest. They assume that nodes on the path between the sender and the receiver belong to different entities and do not cooperate. Otherwise, anonymity can be easily compromised. We investigate an example of such a system in [2].

We expect Sybil-free self-certified pseudonyms to be used in admission control schemes [21, 25, 26] to aid applications, such as those discussed in this section to manage anonymous or pseudonymous users in a secure and privacy-respecting manner. Privacy-friendly admission control allows to create and manage identity domains comprised of several parallel and unlinkable identity domains. Thus, a user can be part of multiple identity domains simultaneously (e. g. different online communities) and keep the identities used in different domains unlinkable.

4 Preliminaries: *k*-Spendable E-Tokens

We use a special signature scheme for creating pseudonym certificates: Camenisch *et al.* have proposed a protocol for periodically spendable e-tokens [9].

In their scenario, sensors spend an e-token whenever they report some data. Yet, it is only possible to compute k different e-tokens per time period. Consequently, sensors can file at most k reports per time period anonymously. Otherwise the sensors have to spend some e-token twice, which allows everyone to compute the sensor's identity from these two e-token show transcripts.

The e-token based signature scheme consists of the algorithms IKg, UKg, $Obtain$, $Issue$, $Sign$, $Verify$, $Identify$, and $Revoke$. These algorithms are executed by the issuer $\mathcal{I}$ of e-token dispensers, the user $\mathcal{U}$, and the signature verifiers:

- $IKg(1^k)$ and $UKg(1^k, pk_{\mathcal{I}})$ – creates the issuers key pair $(pk_{\mathcal{I}}, sk_{\mathcal{I}})$ and the user's key pair $(pk_{\mathcal{U}}, sk_{\mathcal{U}})$, respectively. The value k is the security parameter;
- $Obtain(pk_{\mathcal{I}}, sk_{\mathcal{U}}) \leftrightarrow Issue(pk_{\mathcal{U}}, sk_{\mathcal{I}})$ – at the end of this protocol between a user and the e-token issuer, the user obtains an e-token dispenser D that can be used to create one e-token based signature per ctx. I stores $pk_{\mathcal{U}}$ and revocation information r_D under the user's identity;
- $Sign(m, D, pk_{\mathcal{I}}, ctx)$ – shows an e-token from dispenser D in context ctx to sign a message m. The outputs are a token serial number (TSN) S, a transcript τ, and an updated e-token dispenser D';
- $Verify(m, S, \tau, pk_{\mathcal{I}}, ctx)$ – checks that S and τ were created by a valid dispenser D to sign a message m in context ctx;
- $Identify(pk_{\mathcal{I}}, S, \tau, \tau', m, m')$ – given two records (S, τ) and (S, τ') created by a dispenser D when signing m and m', $m \neq m'$, respectively, $Identify$ computes the public key $pk_{\mathcal{U}}$ of the owner of D;
- $Revoke(sk_{\mathcal{I}}, pk_{\mathcal{I}}, r_D)$ – takes as input $sk_{\mathcal{I}}$ and $pk_{\mathcal{I}}$ and the revocation information r_D that corresponds to a particular user (see $Obtain$). It outputs an updated issuer public key $pk'_{\mathcal{I}}$. In the rest of the paper, we assume that all parties use the most up-to-date issuer key for signing and verification.

4.1 Cryptographic Related Work

Different cryptographic systems can be used to create unlinkable and unique pseudonyms. As long as the identification of "double-spent" pseudonyms is not an issue, such pseudonyms can be realized based on the so-called epoch number of direct anonymous attestation [8]. Schemes that support identification were presented in [9] and [15]. By binding a different tag to every identity domain, k-times anonymous authentication [28] can be used to create unique pseudonyms. Our scheme uses the cryptographic techniques of Camenisch *et al.* [9] (i. e., e-tokens), but can be seen as a more general systems framework that could also be instantiated using other cryptographic techniques.

4.2 Realization of the Cryptographic Algorithms

Briefly, the above functionality can be realized as follows. The issuer and the user both generate key pairs. Let the user's key pair be $(pk_{\mathcal{U}}, sk_{\mathcal{U}})$, where $pk_{\mathcal{U}} = g^{sk_{\mathcal{U}}}$ and g generates a group $\mathbb{G}$ of known order. The issuer's key pair is used for

creating and verifying CL signatures. We use a PRF f_s whose range is the group $\mathbb{G}$. Using *Obtain*, the user interacts with the issuer running the $Issue$ algorithm and obtains an e-token dispenser D that allows her to show one e-tokens per context. The dispenser D is comprised of seed s for the PRF f_s, the user's secret key sk_U, and the issuer's CL signature on $(s, sk_{\mathcal{U}})$. CL signatures are used to prevent the issuer from learning anything about s or $sk_{\mathcal{U}}$. Moreover, the dispenser D is revoked by revoking the corresponding CL signature [11]. In the $Sign$ algorithm, the user shows her token for the context ctx: she releases a serial number $S = f_s(0\|ctx)$, a double-show tag $E = pk_{\mathcal{U}} \cdot f_s(1\|ctx)^{h(m)}$, and using the Fiat-Shamir heuristic [18] creates a non-interactive ZK proof σ that (S, E) correspond to a valid dispenser for context ctx (i. e., the user proves in zero-knowledge that S and E were properly formed from values $(s, sk_{\mathcal{U}})$ signed by the issuer). To sign message m, m is hashed into the challenge together with the first message and the public parameters of the proof. The transcript τ contains both E and σ. An e-token is verified by checking the non-interactive proof.

Unlinkability and Identification. As f_s is a pseudo-random function, and all proof protocols are zero-knowledge, it is computationally infeasible to link the resulting e-token to the user, the dispenser D, or any other e-tokens corresponding to D. If a user shows two e-tokens in the same context to authenticate two messages m and m', then both e-tokens *must* use the same serial number. The issuer can easily detect the violation and compute $pk_{\mathcal{U}}$ from the two double-show tags, $E = pk_{\mathcal{U}} \cdot f_s(1\|ctx)^{h(m)}$ and $E' = pk_{\mathcal{U}} \cdot f_s(1\|ctx)^{h(m')}$. From the equations above, $f_s(1\|ctx) = (E/E')^{(h(m)-h(m'))^{-1}}$ and $pk_{\mathcal{U}} = E/f_s(1\|ctx)^{h(m)}$. For a more detailed security analysis, we refer the reader to [9].

4.3 Cryptographic Details

This writeup is based on a similar writeup for compact e-cash [10]. See the Appendix for more details about the cryptographic primitives used in the writeup. To provide the full protocols for e-token signatures, we provide details about the CL signature scheme used by the issuer. Let QR_n denote the set of quadratic residues modulo n. Let Z, U, V be elements of QR_n that are part of the public key of the issuer. Let l_n denote the number of bits of the issuer's RSA modulus n and let l_o be a security parameter controlling the statistically zero knowledge property of the proof protocol as well as the statistically hiding property of the commitment schemes we use. A signature of the issuer on the seed (message) s consists of the values (Q, e, v), where $e \in \{2^{l_e} - 2^{l_{e'}}, 2^{l_e} + 2^{l_{e'}}\}$ is a random prime, $v \in \{0,1\}^{l_n+l_o}$ is a random integer, and $Q \in \langle U \rangle \subset QR_n$, such that the following holds:

$$Z \equiv Q^e V^v U^s \ (mod\ n).$$

The issuer does not learn s when issuing this signature. Rather, the issuer and user run a two-party protocol where the output of the user will be (Q, e, v). If the issuer signs a block of messages at once, say (u, s), we replace V by (V_1, V_2) in the public key. The signature still consists of values (Q, e, v) such that

$$Z \equiv Q^e V_1^u V_2^s U^s \ (mod\ n).$$

We now describe how the user creates an e-token signature in more detail. Recall that the user has obtained from the issuer a signature (Q, e, v) on u and s.

1. The user computes the serial number $S = g^{1/(s+0\|ctx)}$ and a security tag $E = pk_{\mathcal{U}} g^{H(m)/(s+1\|ctx)}$
2. The user chooses a random r_B and computes the commitments $B = \mathbf{g}^u \mathbf{h}^{r_B}$.
3. The user chooses a random r and computes $Q' := QU^r$. Note that $(Q', e, v + r)$ is also a valid signature on the message u and s but that Q' and Q are statistically independent.
4. The user computes the following signature proof of knowledge:

$$\begin{aligned}\sigma = SPK\{(\alpha,\mu,\gamma,\zeta,\epsilon,\rho_1,\rho_2) : \\ Z = \pm Q'^{\epsilon} V_1^{\mu} V_2^{\gamma} U^{\zeta} \wedge \\ g = S^{\gamma} S^{0\|ctx} \wedge B = \mathbf{g}^{\mu} \mathbf{h}^{\rho_2} \wedge \\ 1 = B^{\gamma} B^{1\|ctx} (1/\mathbf{g})^{\alpha} \mathbf{h}^{\rho_1} \wedge \\ g^{H(m)} = E^{\gamma} E^{1\|ctx} (1/g)^{\alpha} \wedge \\ \gamma \in \{0,1\}^{l_m + l_o + l_n + 2} \wedge \\ (\epsilon - 2^{l_e}) \in \{0,1\}^{l_{e'} + l_o + l_n + 1} \\ \}(Z,V_1,V_2,U,\mathbf{g},\mathbf{h},n,g,S,E,B,m) \end{aligned} \quad (1)$$

The parts of the proof related to the CL signature are done over QR_n while the proofs for S and E are done in $\mathbb{G}$. Elements g and h are generators of $\mathbb{G}$; $\mathbf{g}$ and $\mathbf{h}$ are generators of QR_n.

5 Self-certified Sybil-free Pseudonyms

In this section we describe how self-certified Sybil-free pseudonyms can be constructed and used.

5.1 Instantiation based on E-Token Signatures

In this section, we describe how to implement self-certified Sybil-free pseudonyms by using e-token signatures as a base. The pseudonym certificates $cert_{(\mathcal{U},ctx)}$ that come with the self-certified pseudonyms provide three main functions: (i) the binding of a freshly generated public key to the pseudonym (as with identity certificates); (ii) the verification of the pseudonym and the binding, and; (iii) the disclosure of the user identity and revocation of her certificates should the same membership certificate be used to create two different pseudonym certificates for the same identity domain.

While k-spendable e-tokens provide the necessary main functionality for fulfilling our requirements, we adapt their solution in several ways: (i) while their show protocol is interactive we require non-interactive publicly verifiable shows for signature verification; (ii) we bind a temporal public key to the e-token show

Table 1. A summary of the notation used on the conceptual and the solution level.

Conceptual Level	**Solution Level**
membership certificate $cert_{\mathcal{U}}$	dispenser D
pseudonym certificate $cert_{(\mathcal{U},ctx)}$	transcript τ
pseudo-random pseudonym $P_{(\mathcal{U},ctx)}$	serial number S
domain identifier, context descriptor ctx	

– the public key is the message that is signed; (*iii*) instead of time periods we limit the number of generated e-tokens per signing context – while a context has a validity period, it may also have a name and other characteristics; and, (*iv*) we use a version optimized for $k = 1$. The first two properties are obtained by applying the Fiat-Shamir heuristic [18], a cryptographic trick that turns certain interactive identification protocols into signature schemes. Instead of a time period t, we use an arbitrary context identifier ctx. The value ctx can be seen as identifying the context in which a signer is allowed to sign only once.

The interaction model of our proposal consists of two "phases", one *enrollment* phase in which an initial Sybil-free identity domain is established, and one *identity domain buildup and use* phase where users create and maintain identity domains derived from the original identity domain. See Table 1 for a summary of our notation for the conceptual and the solution level, respectively, and the entities of our system and the roles they may assume are listed in Table 2.

Table 2. A summary of the system entities and their respective roles.

Entities	**Trusted**	**Possible Roles**
Certificate Authority	Yes	Issuer
User	No	User, verifier, domain controller

Enrollment This phase involves several users and one issuer – the certificate authority $\mathcal{I}$. Initially $\mathcal{I}$ generates an e-token issuing key pair $(pk_{\mathcal{I}}, sk_{\mathcal{I}})$ using IKg. To enroll, a user $\mathcal{U}$ creates a membership key pair $(pk_{\mathcal{U}}, sk_{\mathcal{U}})$ using UKg. She transfers $pk_{\mathcal{U}}$ to $\mathcal{I}$ and authenticates under her identity for the Sybil-free identity space. In turn, $\mathcal{U}$ and $\mathcal{I}$ interact using the $Obtain(pk_{\mathcal{I}}, sk_{\mathcal{U}}) \leftrightarrow Issue(pk_{\mathcal{U}}, sk_{\mathcal{I}})$ protocol. In this way $\mathcal{U}$ obtains an e-token dispenser D. It is used as her membership certificate $cert_{\mathcal{U}}$ (see Table 1).

Identity domain buildup and use In this phase, users collectively buildup and participate in identity domains. It consists of three subphases, during which a subset of the users may take the roles of domain controller and / or verifier.

– *Identity domain context creation.* To create a context for an identity domain, a domain controller publishes a domain identifier ctx. As a heuristic, a long-

lived ctx should follow some kind of URI-like (Uniform Resource Identifier) scheme and a short-lived ctx should include the identity domain's validity time. The uniqueness of the domain identifiers used by a user $\mathcal{U}$ can be guaranteed under three conditions: (*i*) $\mathcal{U}$ never turns back her clock; (*ii*) $\mathcal{U}$ keeps a list of all the domain identifiers she has used, and removes records from the list only if the corresponding identity domains have expired; (*iii*) $\mathcal{U}$ only joins domains that have not yet expired and whose domain identifiers are not already on her list.

In addition, ctx may contain the name of the domain, the public key of the domain controller, or even a contract that all of the users who join the domain should agree on. From a practical point of view, there is no hard limit on the size of ctx. It can be hashed down to a constant size value before being used in the cryptographic algorithms. Appending the hash to the validity time makes the uniqueness of ctx independent from the collision resistance of the hash function.

As the identity domain controller does not need to be trusted, any user (or several users) could perform this role. Besides publishing ctx, the identity domain controller will often be responsible for distributing pseudonym certificates. The user that controls the identity domain controller can also participate in the domain, issuing its own pseudonym certificate.

– *Pseudonym certificate creation and verification.* Registration at an identity domain is done using the triplet $(pk_{(\mathcal{U},ctx)}, P_{(\mathcal{U},ctx)}, cert_{(\mathcal{U},ctx)})$, generated as follows: a user $\mathcal{U}$ with a membership certificate $cert_{\mathcal{U}}$ wants to certify a new application specific and hitherto uncertified public / private key pair, which we will from now on call $(pk_{(\mathcal{U},ctx)}, sk_{(\mathcal{U},ctx)})$. She creates a pseudo-random pseudonym $P_{(\mathcal{U},ctx)}$ for a given ctx using the e-token to sign $pk_{(\mathcal{U},ctx)}$. The $Sign(pk_{(\mathcal{U},ctx)}, cert_{\mathcal{U}}, pk_{\mathcal{I}}, ctx)$ algorithm outputs an e-token-based signature (S, τ). $\mathcal{U}$ uses the e-token's serial number S as her pseudo-random pseudonym $P_{(\mathcal{U},ctx)}$, and the transcript τ as her pseudonym certificate $cert_{(\mathcal{U},ctx)}$ (see Table 1). Hence, the domain controller cannot prevent a qualified user (i. e. an user $\mathcal{U}$ with a membership certificate $cert_{\mathcal{U}}$) that knows ctx to join the domain.

  Any user can now verify the correctness of $cert_{(\mathcal{U},ctx)}$ using $Verify(pk_{(\mathcal{U},ctx)}, P_{(\mathcal{U},ctx)}, cert_{(\mathcal{U},ctx)}, pk_{\mathcal{I}}, ctx)$. Afterwards, the uniqueness of the pseudonym can be checked by comparing $P_{(\mathcal{U},ctx)}$ with the pseudonyms of the other certificates for this domain by executing $Identify$. These verifications can be done by any node that is part of the domain, at any choosen time.

– *Misuser identification and revocation.* By executing $Identify$, it is possible to extract the membership public key $pk_{\mathcal{U}}$ of a user from two pseudonym registrations $(pk_{(\mathcal{U},ctx)}, P_{(\mathcal{U},ctx)}, cert_{(\mathcal{U},ctx)})$ and $(pk'_{(\mathcal{U},ctx)}, P'_{(\mathcal{U},ctx)}, cert'_{(\mathcal{U},ctx)})$ if $P_{(\mathcal{U},ctx)} = P'_{(\mathcal{U},ctx)}$ (assured by the system) and $pk_{(\mathcal{U},ctx)} \neq pk'_{(\mathcal{U},ctx)}$. $Identify(pk_T, P_{(\mathcal{U},ctx)}, cert_{(\mathcal{U},ctx)}, cert'_{(\mathcal{U},ctx)}, pk_{(\mathcal{U},ctx)}, pk'_{(\mathcal{U},ctx)})$ will output $pk_{\mathcal{U}}$. Then, $cert_{\mathcal{U}}$ can be revoked using $Revoke$. Note that we do not view a user who reuses the same public key $pk_{(\mathcal{U},ctx)}$ as a Sybil attacker. She is just using the same short term identity again.

5.2 Efficiency

The overall costs of our system are linear in the size of the identity domain with respect to users joining the domain, and quadratic with respect to the verification of the Sybil property: every user needs to execute the *Sign* algorithm for herself, and the *Verify* algorithm for all other users. The construction in [9] requires 10 multi-base exponentiations for pseudonym certificate creation and a similar number of multi-exponentiations for verification. Using multi-base exponentiation tricks, multi-base exponentiations can be made almost as efficient as normal exponentiations. This compares to schemes that do not support identification with about half the number of multi-exponentiations, and ordinary CA-issued pseudonym certificates with one or two exponentiations. Verification may not be needed in all cases, e. g., if users trust the domain controller to verify users on their behalf, or if the application bases its security properties on the assumption that only a set of key users are not Sybil nodes, rather than every single user.

5.3 Security Analysis

This section discusses some security properties of our proposed self-certified pseudonyms.

- *Sybil-Proof Property:* the cryptographic properties of e-token signatures ensure that for each valid membership certificate there can exist only one unique pseudonym $P_{(\mathcal{U},ctx)}$ per identity domain (see Section 4). However, as there is no inherent trust in any user in the identity domain (including the domain controller), users have to check the correctness of the pseudonym certificate $cert_{(\mathcal{U},ctx)}$ of all other users in the domain by locally running $Verify$. After an honest user has finished this verification and has checked the uniqueness of $P_{(\mathcal{U},ctx)}$, she is assured that her communication partner is a real user with public key $pk_{(\mathcal{U},ctx)}$, provided that she authenticated with $sk_{(\mathcal{U},ctx)}$.
- *Unlinkability Property:* our approach has strong unlinkability properties as the cryptographic properties of the e-token signatures ensure the algorithmic unlinkability of two pseudonym certificates generated for different domains (see Section 4). However, should the users violate precautions on the network or application layers, the attacker may still be able to make an educated guess on whether two arbitrary pseudonym certificates from different identity domains are related or not. In a real word scenario, a variety of different information could help the attacker to make such a guess, for instance, the location property of the identity domain or the location of the user. A traffic analysis of each setting is required to assess the concrete threats to the users' privacy.

6 Summary & Outlook

In this paper, we have described the construction of a solution to the Sybil attack that does not require online connectivity to a TTP but preserves user privacy: self-certified Sybil-free pseudonyms. We have discussed some real-world applications that would benefit from our solution. Future work includes implementing the proposed solution in a real system.

Acknowledgments

This research was funded by the European Network of Excellence Future of Identity in the Information Society (FIDIS) and by the European Integrated Project for Privacy and Identity Management for Europe (PRIME), both under the 6^{th} Framework Program for Research and Technological Development within the Information Society Technologies (IST) priority.

References

1. Christer Andersson, Jan Camenisch, Stephen Crane, Simone Fischer-Hübner, Ronald Leenes, Siani Pearson, John Sören Pettersson, and Dieter Sommer. Trust in PRIME. In *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology*, pages 552–559, 2005.
2. Anonymous Authors. Anonymized title. In *Proceedings of Anonymized Conference*, 2008.
3. Endre Bangerter, Jan Camenisch, and Ueli M. Maurer. Efficient proofs of knowledge of discrete logarithms and representations in groups with hidden order. In Serge Vaudenay, editor, *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 154–171. Springer, 2005.
4. Abhilasha Bhargav-Spantzel, Jan Camenisch, Thomas Gross, and Dieter Sommer. User centricity: a taxonomy and open issues. In *DIM '06: Proceedings of the second ACM workshop on Digital identity management*, pages 1–10, New York, NY, USA, 2006. ACM Press.
5. Katrin Borcea-Pfitzmann, Elke Franz, and Andreas Pfitzmann. Usable presentation of secure pseudonyms. In *DIM '05: Proceedings of the 2005 workshop on Digital identity management*, pages 70–76, New York, NY, USA, 2005. ACM Press.
6. Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT '00*, volume 1807 of *Lecture Notes in Computer Science*, pages 431–444, 2000.
7. Stefan Brands. Rapid demonstration of linear relations connected by boolean operators. In *EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 318–333, 1997.
8. Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In *CCS '04: Proceedings of the 11th ACM conference on Computer and communications security*, pages 132–145, New York, NY, USA, 2004. ACM Press.
9. Jan Camenisch, Susan Hohenberger, Markulf Kohlweiss, Anna Lysyanskaya, and Mira Meyerovich. How to win the clone wars: Efficient periodic n-times anonymous authentication. In *ACM Conference on Computer and Communications Security*. ACM, 2006.

10. Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 302–321. Springer, 2005.
11. Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In Moti Yung, editor, *CRYPTO*, volume 2442 of *Lecture Notes in Computer Science*, pages 61–76. Springer, 2002.
12. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *SCN 2002*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289, 2002.
13. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72, 2004.
14. Jan Camenisch and Markus Stadler. Proof systems for general statements about discrete logarithms. Technical Report TR 260, Institute for Theoretical Computer Science, ETH Zürich, March 1997.
15. Ivan Damgård, Kasper Dupont, and Michael Østergaard Pedersen. Unclonable group identification. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 555–572. Springer, 2006.
16. Yevgeniy Dodis and Aleksandr Yampolskiy. A Verifiable Random Function with Short Proofs an Keys. In *Public Key Cryptography*, volume 3386 of LNCS, pages 416–431, 2005.
17. John R. Douceur. The Sybil Attack. In *Peer-to-Peer Systems: Proceedings of the 1^{st} Int Peer-to-Peer Systems Workshop (IPTPS)*, volume 2429, pages 251–260. Springer-Verlag, 7–8 Mar 2002.
18. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Verlag, 1987.
19. Elke Franz and Katrin Borcea-Pfitzmann. Intra-application partitioning in an elearning environment - a discussion of critical aspects. In *ARES '06: Proceedings of the First International Conference on Availability, Reliability and Security (ARES'06)*, pages 872–878, Washington, DC, USA, 2006. IEEE Computer Society.
20. Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 16–30, 1997.
21. Yongdae Kim, Daniele Mazzocchi, and Gene Tsudik. Admission control in peer groups. In *NCA*, pages 131–139. IEEE Computer Society, 2003.
22. Sébastien Kunz-Jacques, Gwenaëlle Martinet, Guillaume Poupard, and Jacques Stern. Cryptanalysis of an efficient proof of knowledge of discrete logarithm. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 27–43. Springer, 2006.
23. Brian Neil Levine, Clay Shields, and N. Boris Margolin. A survey of solutions to the sybil attack. Tech report 2006-052, University of Massachusetts Amherst, Amherst, MA, Oct 2006.
24. Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '92*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, 1992.
25. Nitesh Saxena, Gene Tsudik, and Jeong Hyun Yi. Admission control in peer-to-peer: design and performance evaluation. In Sanjeev Setia and Vipin Swarup, editors, *SASN*, pages 104–113. ACM, 2003.

26. Nitesh Saxena, Gene Tsudik, and Jeong Hyun Yi. Efficient node admission for short-lived mobile ad hoc networks. In *ICNP*, pages 269–278. IEEE Computer Society, 2005.
27. Claus P. Schnorr. Efficient signature generation for smart cards. *Journal of Cryptology*, 4(3):239–252, 1991.
28. Isamu Teranishi, Jun Furukawa, and Kazue Sako. k-times anonymous authentication (extended abstract). In *ASIACRYPT*, volume 3329 of *Lecture Notes in Computer Science*, pages 308–322. Springer, 2004.
29. Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham Flaxman. SybilGuard: defending against sybil attacks via social networks. In *SIGCOMM '06*, pages 267–278, New York, NY, USA, 2006. ACM Press.

A Cryptographic Building Blocks

A *zero-knowledge* (ZK) proof is an interactive proof in which the verifier learns nothing besides the fact that the statement that is proven is true. This notion is defined by means of a *simulator*, which can reproduce the communication knowing only what the verifier knows. A *proof of knowledge* is an interactive proof in which the prover succeeds in *convincing* a verifier that it knows something. What it means for a machine to *know something* is defined in terms of computation. A machine *knows something*, if this something can be computed, given the machine as an input. The machine extracting the knowledge is called the *knowledge extractor*. Protocols with a simulator and a knowledge extractor are called zero-knowledge proofs of knowledge.

For some protocols only simulators that work for honest verifiers are known. These are verifiers that choose the challenge according to a predetermined distribution. Honest-verifier zero-knowledge proofs-of-knowledge protocols that have a three move structure – commitment, challenge and response – are called *sigma protocols*. Such protocols can be made non-interactive by applying a cryptographic trick called *Fiat-Shamir heuristic* [18]. This heuristic uses a cryptographic hash function to allow the prover to compute the challenge herself without involving the verifier. Non-interactive proofs of knowledge have the advantage that they do not require interaction between the prover and the verifier. In addition, they allow to sign any message by hashing it together with the first message when creating the challenge.

Sigma protocols exist for proving knowledge of discrete logarithm (DL), equality of DLs, and linear relations between DLs in groups of known [7, 14, 27], and hidden order [3, 22]. This allows us to prove statements about certain algorithms (some of wich are detailed below) that operate in these groups, for instance that two commitments contain the same value or that a committed value lies in a certain interval [6], that we know a signature for a value or a committed value, that a value was verifiable encrypted, or that a value was correctly created using a pseudo-random function and a secret seed.

B Cryptographic Primitives

DY Pseudorandom Function (PRF). Let $\mathbb{G} = \langle g \rangle$ be a group of prime order $q \in \Theta(2^k)$. Let a be a random element of $\mathbb{Z}_q^*$. Dodis and Yampolskiy [16] showed that $f_{g,a}^{DY}(x) = g^{1/(a+x)}$ is a pseudorandom function, under the decisional Diffie-Hellman inversion assumption (y-DDHI), when either: (1) the inputs are drawn from the restricted domain $\{0,1\}^{O(\log k)}$ only, or (2) the adversary specifies a polynomial-sized set of inputs from $\mathbb{Z}_q^*$ *before* a function is selected from the PRF family (i.e., before the value a is selected). For our purposes, we require something stronger: that the DY construction work for inputs drawn arbitrarily and adaptively from $\mathbb{Z}_q^*$. Dodis-Yampolskiy PRF is adaptively secure for inputs in $\mathbb{Z}_q^*$ under the SDDHI assumption [9].

Pedersen and Fujisaki-Okamoto Commitments. Recall the Pedersen commitment scheme [24], in which the public parameters are a group $\mathbb{G}$ of prime order q, and generators $(g_0, \ldots, g_m)$. To commit to the values $(v_1, \ldots, v_m) \in {\mathbb{Z}_q}^m$, pick a random $r \in \mathbb{Z}_q$ and set $C = PedCom(v_1, \ldots, v_m; r) = g_0^r \prod_{i=1}^m g_i^{v_i}$. Fujisaki and Okamoto [20] showed how to expand this scheme to composite order groups.

CL Signatures. The Camenisch and Lysyanskaya signature scheme [12] includes two protocols: (1) An efficient protocol for a user to obtain a signature on the value in a Pedersen (or Fujisaki-Okamoto) commitment [20, 24] without the signer learning anything about the message. (2) An efficient proof of knowledge of a signature protocol. Security is based on the Strong RSA assumption. Using bilinear maps, we can use other signature schemes [13] for shorter signatures.