Extensibility in a Privacy-preserving eID: Towards a Mobile eID System for Real-world Identification and Offline Verification

Michael Hölzl[1], Michael Roland[2], and René Mayrhofer[1]

[1] Insitute of Networks and Security, JKU Linz, Austria,
hoelzl@ins.jku.at, mayrhofer@ins.jku.at
[2] University of Applied Sciences Upper Austria, Hagenberg, Austria
michael.roland@fh-hagenberg.at,

Abstract. There is a broad range of existing electronic identity (eID) systems which provide methods to sign documents or authenticate to online services (e.g. governmental eIDs, FIDO). However, these solutions mainly focus on the validation of an identity to a web page. That is, they lack in providing proper techniques to use them as regular ID cards to digitally authenticate an eID holder to another physical person in the real world. We envision a mobile eID which provides such a functionality and enables extensibility for its use with numerous different public and private services (e.g. for loyalty programs, public transport tickets, students cards), while protecting the privacy of the eID holder. In this paper, we present a general architecture and efficient protocols for such a privacy-preserving mobile eID that allows identity validation in a similar fashion as regular ID cards and makes carrying around various physical cards unnecessary.

1 Introduction

Many governments already provide their citizens with an electronic identity (eID) infrastructure to handle administrative tasks like doing taxes or applying for subsidies (cf. survey of European governmental eIDs by Lehman et al. [15]). However, they lack appropriate methods to allow eID holders to use the eID in a privacy-preserving manner. In our terms, such a privacy-preserving eID gives the prover (i.e. the eID holder) the capability to only reveal and prove the validity of certain attributes to a verifier. For example, an eID holder wants to prove to the bouncer at a disco that she is above 18 years old without revealing the name or even the actual date of birth. Furthermore, a privacy-preserving eID should also not leak any usage behavior to the verifier (e.g. how often does a specific eID holder enter the disco).

There are existing solutions for such a privacy-preserving eID, where the most recent ones are based on attribute-based credentials (ABC) [5, 6]. ABC allow eID holders to prove a subset of their personal data attributes (e.g. age, name, citizenship) without revealing the full set. They have already been actively used for pilot studies in the ABC4Trust project [4] and have been implemented on smart cards [3, 10, 21]. Alpár and Jacobs [2] discuss the difficulties of such ABC systems (which applies to smart card-based identity systems in general):

- Controlling attribute access for verifiers requires either additional technical restrictions (i.e. let each verifier get a signed list of readable attributes from the identity manager), legal restrictions (i.e. only verifiers with legal contracts are allowed to read attributes), or additional monitoring on the card.
- Verifying that the person presenting the smart card is the actual eID holder requires additional communication channels (e.g. picture on the card).
- The usage of PIN protection for smart card ensures confidentiality, user consent and authentication, but adds additional complexity (e.g. the PIN may have to be entered in every verification on the card reader of the verifier).

One possible solution to these issues of smart card-based eIDs is the usage of a mobile eID (e.g. Jensen's evaluation of ABC on smart phones in [14]). However, current eID solutions (e.g. governmental IDs, OpenID, ABC4Trust, etc.) mainly focus on identifying a user to a backend infrastructure and lack in discussing the additional use cases of such a mobile eID: (i) Provers can use their mobile eID in a similar fashion as regular ID cards to prove their identity (we refer to this as *real-world identification*). Identification should even work with turned-off prover devices and in an offline setting. (ii) The mobile eID may be used for multiple public or private services (we refer to them as *domains*). Services could be loyalty programs, time-limited public transport tickets, students cards, etc.

We envision a mobile eID system which provides the capability of real-world identification, similar to regular ID cards, and the possibility to easily extend it for numerous use cases. Furthermore, the system shall respect the privacy of the eID holder. That is, none of the verifiers should be able to acquire data attributes irrelevant to them or trace the users' activities. In this paper we describe the general architecture of such a mobile eID scheme and propose protocols to enroll to numerous domains and verify data attributes in an efficient way. The architecture allows to provide proofs of single eID attributes in a privacy-preserving manner and builds upon state-of-the-art technologies in that field.

2 Related Work

A specification for eIDs that has recently become famous is provided by the FIDO Alliance in [8]. This consortium aims to improve the usability of user authentication on the Internet by reducing the reliance on passwords. With one specification for biometric authentication and one for second-factor authentication, they provide schemes for secure identity verification to any online service.

Concerning governmental eIDs, the survey by Lehman et al. [15] about eIDs in the European Union shows that current systems do not provide sufficient privacy-preserving verification methods. Only the Austrian and German eID cards support notable features for protecting the privacy of the user (i.e. generation of pseudonyms, selective attribute disclosure).

Nyman et al. [17] define a governmental and privacy-preserving eID architecture that is based on the use of so-called Trusted Platform Modules (TPM). They build upon version 2.0 of the TPM specification and evaluate its feasibility as an identity token on PC as well as mobile platforms. Similar to our concept, their system relies on the tamper resistance of additional hardware in computing devices but do not provide an easily extensible solution.

Attribute-based credentials (ABC) [5, 7] build the basis for another field of research in the area of privacy-preserving eID schemes. Most important technologies in that field are the Identity Mixer (Idemix) [13], developed by IBM Research, and Microsoft's U-Prove system [18]. In an ABC scheme, a credential is referred to as a cryptographic container for multiple attributes. An attribute, on the other hand, is a property about a person that some trusted authority attested. The ABC4Trust project [4] also analyzed possible schemes to implement ABC on mobile platforms [14] but lack in providing solutions to the additional challenges of such a mobile eID (e.g. can run out of battery, phone gets stolen).

The benefit of ABCs is that besides ensuring authenticity and integrity of eID attributes, it also provides some privacy guarantees for credential owners. That is, it allows the eID holder to prove certain predicates of an attribute without revealing the actual content. Moreover, each verification of a single eID appears unrelated and can therefore not be linked by a verifier. The downside of ABC is the higher complexity of issuing attributes and creating proofs. Even as they have been successfully deployed on smart cards [3, 10], they are still considerably slower than ordinary signature schemes and can become the bottleneck in a privacy-preserving eID system. For example, Vullers and Alpár report the results of a 1024 bit smart card implementation in [21]. Disclosing and creating a proof for only one credential (with 4 attributes) takes already 1 second (incl. overhead). They also make a reference that increasing the security level to a 2048 bit modulus would more than double the computation time. However, we assume that a regular transaction requires more than only 4 attributes and that a user is not willing to wait for more than 2 seconds to finish the identification process. Hence, for our use case, the performance of an eID system solely based on ABC is not sufficient.

3 Threat Model

In our terms, a privacy-preserving eID has to consider the following threats:

- Forging identities. An adversary could attempt to fake an identity or modify data attributes. This would allow an attacker to adapt single data attributes for an attack (e.g. modify the age or place of residence), impersonate someone else, or even result in digital identity theft (e.g. take over mail and bank accounts). In order to prevent such attacks, the eID system needs to make assurances on the integrity of the identity and only allow authorized entities (e.g. government authorities) to modify the data.
- Data leakage. As information is digitally processed, it becomes difficult for users to stay in control over their eID data. In an identification process, they cannot be sure that the data they transmit to a verifier is adequately

protected, only used for the claimed purpose (of identification), and not stored or passed on to other parties. Hence, in order to protect the privacy it is important that as little information as necessary is given to verifiers.
- Tracing identities. An adversary could use the digital information provided by the eID to trace activities of a eID holder. For example, the disco bouncer or the public transport system that use our extensible eID system could track all identification processes and therefore trace all activities of a single user. The eID system should therefore protect against such attacks on privacy.
- Linking pseudonyms. A system that provides pseudonymity shall not allow verifiers to link single pseudonyms to each other. For example, a shop, where the user has a loyalty card enrolled, should not be able to derive, link, or determine other pseudonyms of the user.

4 Extensible and Privacy-preserving Mobile eID

We propose a government issued eID on mobile devices that has the flexibility to be used as a regular identification document and for the use by numerous services. More specifically, this eID shall provide:

- Real-world identification. The mobile eID can be used as a replacement for regular ID cards and can be used for identification and verification of attributes (e.g. age of the eID holder).
- Extensibility. An eID can be used as simple identity token or for numerous different services with the possibility to create pseudonyms for each service. A specific design decision for our architecture was simplicity when service providers want to integrate with an existing eID based on our scheme. Any service provider can easily and rapidly establish their own e.g. loyalty program on top of our mobile eID system.
- Privacy. Despite the extensive usage of the eID for multiple services, verifiers are required to have proper authorization to obtain a pseudonym from the eID. Moreover, they cannot link multiple pseudonyms to a single eID or trace a specific eID. Furthermore, users stay in full control of their data.
- Capable for offline and turned-off devices. The prover mobile device does not need to be powered on in order to verify the identity of the eID holder and no constant online connectivity to a central server shall be required.

4.1 Stakeholders

Figure 1 depicts the proposed eID architecture consisting of four stakeholders:

- The *eID issuer* is the central authority that controls the enrollment of new eIDs and provides an interface to acquire the public system parameters for eID verification (e.g. governmental authority in a nationwide eID).
- The *prover* is the actual owner of the eID and consists of a mobile device equipped with a secure element (SE). Communication to the SE can be done directly over near field communication (NFC) or through the eID management application (eID-MA) on the mobile device.

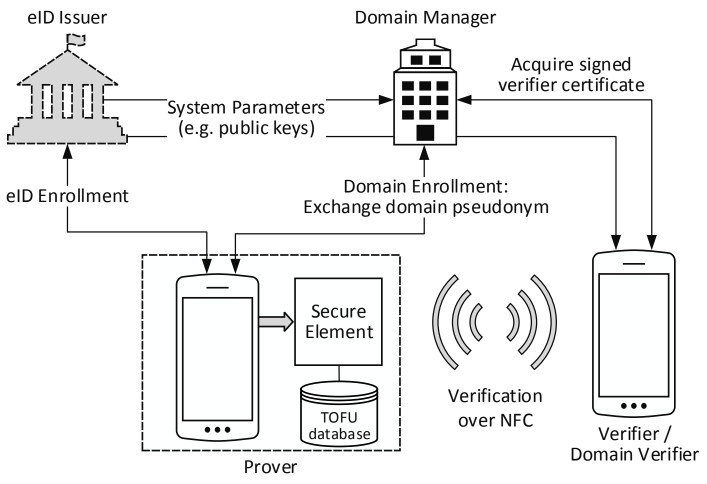
eID Issuer
Domain Manager
Acquire signed verifier certificate
System Parameters (e.g. public keys)
eID Enrollment
Domain Enrollment: Exchange domain pseudonym
Secure Element
TOFU database
Prover
Verification over NFC
Verifier / Domain Verifier

Fig. 1: General architecture of the proposed mobile and extensible eID system.

- The *domain manager* is responsible for controlling the enrollment of a prover to a specific *domain* (i.e. a service). A domain may have additional attributes associated to an eID or require a pseudonym for the prover.
- The *verifier* can be any user who wants to identify the eID holder and verify certain attributes. For example, a disco bouncer wants to verify that a guest has a valid eID and is above 18 years. Additionally, a verifier can be domain member and read domain-specific attributes (referred to as *domain verifier*).

4.2 Building Blocks

We fulfill privacy and security requirements by building upon and combining several techniques. That is, we make use of secure elements for the protection of sensitive data and use ABCs to verify the eID in a privacy-preserving manner as well as authenticate an additional secure channel:

Secure Elements (SE) Our architecture assumes the existence of a trustworthy SE on the prover's device. An SE is usually shipped as an embedded integrated circuit in mobile devices together with NFC [16] (e.g. as a SIM card) and brings two main security advantages: (i) it protects against unauthorized access as well as tampering, and (ii) small applications (applets) can be executed directly on the card in the trusted execution environment. Another advantage of SEs is that they can be powered by the NFC field when the prover device is turned off (provided that the NFC controller in the prover device supports this feature).

In our eID scenario, the SE shall protect the identities of the eID holders as well as their attributes. All computations that require these data need to be performed within the SE. However, the constrained execution performance and memory on the SE have strong implications on the protocols and architectural design of the eID system (see performance evaluation in [11]). Our proposed architecture acknowledges these requirements and can be executed within this secure but constrained environment in reasonable time.

Attribute-based Credentials (ABC) In our proposed architecture, we make use of ABC for attesting the validity of the eID in a privacy-friendly way. We assume the following properties of ABC: (i) With a *selective disclosure* mechanism, the holder of the credential can reveal any subset of attributes and provide a validity proof of them (i.e. they have been attested by the trusted authority); (ii) Ownership of a credential can be proven without revealing the attributes itself; (iii) Verification of credentials are unlinkable to verifier and issuer.

In order to protect against replay attacks, the selective disclosure mechanism allows the verifier to send a random challenge. The prover responds to this with a non-interactive-zero-knowledge (NIZK) proof, which is also a signature of this challenge. Using this mechanism, we establish an authenticated secure channel as described in [1], and therefore introduce a simple notation for selective disclosure:

$$\pi = \mathrm{SD}\left(\mathcal{A}, ch\right), \tag{1}$$

where $\mathcal{A}$ is the subset of disclosed attributes from a credential and ch is the random challenge. We assume that this operation can be executed on the SE. However, due to its complexity and the computational constraints of the SE, only few selective disclosure operations shall be used.

Note that the main focus in this paper is the extensibility of a mobile eID and a usage of ABC for attesting the validity of an eID in a privacy-friendly manner. A detailed discussion and evaluation of ABC is out of scope.

4.3 Extensibility and Privacy-preserving Mechanisms

A central component of our architecture is the possibility to extend the eID usage to numerous domains. Such a domain could be any public or private service (e.g. loyalty card programs, public transport tickets). Our proposed architecture thereby aims for simplicity for domain managers and users. Hence, it should be easy for service provider to make use of our mobile eID system and for eID holders to control their data. We define three main mechanism for that purpose:

Profiles In order to give the user control over the data, we introduce the concept of profiles to the mobile eID architecture. A profile defines data attributes which are accessible for a specific purpose or a group of verifiers. A profile could be a birthday verification profile where only the date of birth is accessible, an identity verification profile where only the portrait picture and the name are accessible, etc. The management application (eID-MA) running on the mobile device of the prover maintains these profiles. It is also possible to associate one profile with a domain in a trust-on-first-use (TOFU) database. This enables the user to remember data attributes that can be retrieved by specific domain verifiers. In order to support verification with a turned-off prover device, the SE stores a profile for each domain as well as a default profile.

Trust-on-First-Use (TOFU) Each SE of an eID holder possesses a TOFU database with information about enrolled domains. An entry consists of the identifier (i.e. public key) of the domain D_{id} and a profile, which defines the attributes a domain verifier can query. In addition, there might be additional attributes stored for that specific domain (e.g. validity of loyalty program membership).

Domain Pseudonyms An eID holder that enrolls to a number of domains has unlinkable pseudonyms for each of them. They are derived from the identity of the eID holder as well as the identifier of the domain manager and can be used for domain-specific identification (e.g. for bonus point system in loyalty programs). We use a mechanism that does not require additional space on the SE. It also provides the capability for multiple devices of an eID holder to derive the same pseudonym for a specific domain.

5 Protocols

The protocols in our scheme use profiles for easy attribute selection and builds upon ABC to validate the eID as well as authenticate an additional secure channel. This secure channel is used to efficiently transfer the data attributes of the profile to the verifier. In addition, we introduce a simple mechanism to derive domain pseudonyms, which do not require additional space on the SE, and a TOFU database to store domain-specific profiles and attributes on the prover device (i.e. the SE). The notation of the protocol is listed in Table 1.

Table 1: Notation used in this paper.

id_u	Secret identifier of the user.
da_u	Data attributes of the user.
$\mathcal{C}_u$	ABC key credential of user u for eID validation.
$N_{u,d}, n_{u,d}$	Derived domain pseudonym and the corresponding secret key.
G	Elliptic curve generator point.
D_{id}, d_{sk}	Public/private key-pair of domain manager d.
la_d	List of attribute identifiers which a domain d wants to access.
V_{pk}, v_{sk}	Public/private key-pair of verifier v.
$ca_{v,d}$	Certificate of domain verifier v.
$\mathrm{H}(m)$	One-way hash function over message m.
$\mathrm{Enc}(K, m)$	Symmetric encryption of message m using key K.
$\mathrm{Sign}(sk, m)$	Signature creation over message m with private key sk.
$\mathrm{SD}(\mathcal{A}, ch)$	Create a NIZK proof of an attribute-set $\mathcal{A}$ in a given ABC credential, while using the random challenge ch.

5.1 Setup

During setup, every involved party receives the public system parameters of the used ABC system as well as the elliptic curve parameters. Every domain manager creates a public/private key pair $(d_{sk}, D_{id} := d_{sk} \cdot G)$, where the public key D_{id} is also serves as the domain identifier, and define a list la_d, which specifies the attributes they want to access from a user. Each verifier generates a key-pair $(v_{sk}, V_{pk} := v_{sk} \cdot G)$ and the TOFU databases of each SE are empty.

5.2 Prover Enrollment

There are two types of enrollment: *eID* and *domain* enrollment. The enrollment of the eID can only be done once for every SE while the domain enrollment is only limited to the available storage space for the TOFU database on the SE.

eID Enrollment During the initial eID enrollment, the SE of the prover and the eID issuer communicate in a secure channel using GlobalPlatform card management [9]. The eID-MA acts as a proxy between them. The process is initiated by the eID holder and presumably involves an additional out-of-band identity verification (the detailed steps of this enrollment are out-of-scope of this paper). We assume that during this process the SE acquires the secret identifier id_u and the data attributes of the eID holder da_u. The SE also acquires an ABC key credential $\mathcal{C}_u$ from the issuer, which is used to validate the eID and authenticate an additional secure channel with the method described by Alpár and Hoepmann in [1].

Domain Enrollment and Pseudonym Derivation An exemplary scenario for this enrollment would be an eID holder who would like to join a loyalty card system. This enrollment process is performed by the manager of a domain, the prover's mobile device and the SE. The eID-MA running on the prover's mobile device acts as a proxy between domain manager and SE. In contrast to the eID enrollment, domain enrollment does not require GlobalPlatform card management. Hence, user approval is sufficient (e.g. through entering a PIN/password that is verified on the SE) to add domains to the provers' SE.

The protocol steps of the domain enrollment consists of establishing an authenticated and privacy-friendly secure channel with ABC credentials (based on the scheme in [1]). This secure channel is then used to efficiently transfer eID data attributes directly between the SE of the prover and the domain manager. Additionally, during this process the domain manager and the SE authenticate the pseudonym of the user and the domain public key, respectively:

1. The process is initiated by the eID-MA, for example, when the user taps an NFC tag with in the shop with a loyalty program. In this first step, the eID-MA sends the enrollment request to the domain manager.

2. The domain manager creates a new ephemeral key-pair $(a, A := a \cdot G)$ as well as a signature $\sigma_{i,d}$ over the public part A as well as la_d:
$$\sigma_{i,d} = \mathrm{Sign}\left(d_{sk}, \mathrm{H}\left(A \,||\, la_d\right)\right) \tag{2}$$
The manager sends $(\sigma_{i,d}, A, D_{id}, la_d)$ to the eID-MA.
3. The eID-MA asks the user to confirm the enrollment and the attribute disclosure of la_d. If the user rejects, the enrollment aborts. Otherwise, he has to authenticate himself with a previously defined PIN/password and the enrollment message is forwarded to the SE. Note that the user may only confirm a subset of the attributes in la_d. The SE conceals the remaining attributes from the domain.
4. On successful authentication, the SE proceeds and verifies the signature $\sigma_{i,d}$ with the received domain public key D_{id}. Furthermore, the SE checks in the TOFU database if an entry with the domain public key D_{id} already exists. If the signature is invalid or an entry exists, the SE sends an error message to eID-MA and aborts. Otherwise, it derives a pseudonym $N_{u,d}$ for that domain:
$$n_{u,d} = \mathrm{H}\left(id_u \,||\, D_{id}\right) \tag{3}$$
$$N_{u,d} = n_{u,d} \cdot G \tag{4}$$
Note that the pseudonym does not have to be stored on the SE (i.e. no additional space required) and can be easily derived in each verification (see Section 5.4). Also multiple devices of a user will derive the same pseudonym for a domain (i.e. all SEs receive the same id_u during eID enrollment).
For the next step, the SE creates a new ephemeral key-pair $(b, B := b \cdot G)$ and a NIZK proof over A, B, and $N_{u,d}$, using the ABC key credential $\mathcal{C}_u$. Furthermore, the signatures $\sigma_{i,N}$ and $\sigma_{i,B}$ verify the knowledge of the secret-key $n_{u,d}$ and ephemeral key b, respectively:
$$\pi_i = \mathrm{SD}\left(\mathcal{C}_u, \mathrm{H}\left(A \,||\, B \,||\, N_{u,d}\right)\right) \tag{5}$$
$$\sigma_{i,n} = \mathrm{Sign}\left(n_{u,d}, \mathrm{H}\left(A \,||\, \pi_i\right)\right) \tag{6}$$
$$\sigma_{i,b} = \mathrm{Sign}\left(b, \mathrm{H}\left(A \,||\, \sigma_{i,n}\right)\right) \tag{7}$$
The SE sends $(N_{u,d}, B, \pi_i, \sigma_{i,n}, \sigma_{i,b})$ to the domain manager.
5. With the NIZK proof π_i, the ephemeral public key A, and the received pseudonym $N_{u,d}$, the domain manager verifies the eID validity. The signature verification ensures the validity of the pseudonym and the ephemeral key. If any verification fails, the manager aborts. Otherwise, she creates a signature $\sigma_{i,A}$ to verify the ephemeral key and computes the session key K_i with
$$\sigma_{i,a} = \mathrm{Sign}\left(a, \mathrm{H}\left(B \,||\, N_{u,d}\right)\right) \tag{8}$$
$$K_i = \mathrm{H}\left(a \cdot B\right), \tag{9}$$
and outputs $(\sigma_{i,a})$ to the SE.
6. If the signature $\sigma_{i,a}$ is valid, the SE also computes the session key K_i:
$$K_i = \mathrm{H}\left(b \cdot A\right) \tag{10}$$

7. The SE and the domain manager use the session key K_i for an authenticated secure channel and to exchange data attributes now. The domain manager can only request attributes which have been confirmed by the user. Additionally, the SE adds a new entry to the TOFU database, with the domain identifier D_{id} and the accepted attributes of the attribute identifier list la_d (i.e. as the profile for that domain). The domain manager might also send additional attributes (e.g. loyalty card validity period), which are also stored on the SE and linked to the TOFU entry.
8. The domain manager stores the pseudonym and the data attributes.

5.3 Profile Selection

Prior to the verification, the user selects a currently active profile within the eID-MA. This profile is then stored on the SE as the default profile and is also active if the mobile device of the prover is turned-off. An example of such a profile would be the birthday verification profile where only the birthday attribute is accessible for verifiers.

In the case where the verifier belongs to a domain, the profile will be automatically selected from the TOFU database after a successful membership verification of the verifier. This is part of the verification protocol.

5.4 Verification

Verification of the eID is done between verifier and prover over NFC. On the prover mobile device the communication is either transfered through the management application to the SE (using NFC host-card emulation) or directly with the SE (if the device is turned-off). Both communication paths use the same protocol steps described in this section. However, the host-card emulation enables the management application to display additional information of the verifier to the user (e.g. domain name, id, etc.)

The verifier can be any user who downloaded and installed the verifier application or can be a specific verifier of a domain. In the latter case, we assume the existence a certificate $ca_{v,d}$, which is essentially a signature of the long-term public key V_{pk} with the secret key of the domain manager d_{sk}.

The protocol steps are similar to the domain enrollment protocol and mainly consist of establishing an authenticated secure-channel between the SE of the prover and the verifier. This channel is based on ABC credentials to validate the eID and allows an efficient data attribute exchange:

1. The process is initiated by the verifier, for example, when the phones of verifier and prover are tapped together and communication over NFC is established. The verifier creates a new ephemeral key-pair (c, $C := c \cdot G$) and sends (D_{id}, $ca_{v,d}$, V_{pk}, C) to the SE. If the verifier is not part of a domain, the domain public key D_{id} and the certificate $ca_{v,d}$ are omitted.

2. The SE also chooses a new ephemeral key-pair $(b, B := b \cdot G)$, creates a NIZK proof over B as well as C and creates a signature using the new secret key:

$$\pi_i = \mathrm{SD}\left(\mathcal{C}_u, \mathrm{H}\left(C \,||\, B\right)\right) \tag{11}$$
$$\sigma_{i,b} = \mathrm{Sign}\left(b, \mathrm{H}\left(C \,||\, \pi_i\right)\right) \tag{12}$$

   The SE sends $(B, \pi_i, \sigma_{i,b})$ to the verifier.
3. The verifier proves the validity of the eID using the NIZK proof π_i and checks the signature $\sigma_{i,b}$ to ensure the validity of the ephemeral key. If any verification fails, the manager aborts; otherwise proceeds by creating a signature using its own ephemeral key c and the long-term secret key v_{sk}:

$$\sigma_{i,v} = \mathrm{Sign}\left(v_{sk}, \mathrm{H}\left(B \,||\, C\right)\right) \tag{13}$$
$$\sigma_{i,c} = \mathrm{Sign}\left(c, \mathrm{H}\left(\sigma_{i,v}\right)\right) \tag{14}$$

   The verifier sends $(\sigma_{i,v}, \sigma_{i,c})$ to the SE and computes the session key K_i:

$$K_i = \mathrm{H}\left(c \cdot B\right) \tag{15}$$

4. If any signature $(\sigma_{i,v}, \sigma_{i,c})$ is not correct, the SE cancels the process. Otherwise, the SE computes the session key K_i:

$$K_i = \mathrm{H}\left(b \cdot C\right) \tag{16}$$

   The SE also chooses the profile that defines the allowed attribute disclosure now. There are two cases:
   (a) The verifier is member of a domain and sent $ca_{v,d}$ and D_{id}: the SE checks in the TOFU database if the domain public key D_{id} is already known. If the domain is not in the database or if the certificate $ca_{v,d}$ does not properly validate the verifier key V_{pk}, the process is aborted. If the check is successful, the profile from the TOFU database is chosen.
   (b) The verifier is not member of a domain: the default profile is chosen.
5. The SE and the verifier use the session key K_i for the attribute exchange in an authenticated secure channel now. During this process, the verifier may request different attributes and based on the chosen profile, the SE decides if the attributes are disclosed or not.
6. If the verifier is a valid member of a domain, she may also request for the domain pseudonym $N_{u,d}$. The derivation of it is the same as described in Equations 3-4. However, as the proposed system should not allow single verifiers to trace the activities of the prover, the pseudonym is not directly revealed to the verifier. That is, we assume that not all domain verifiers can be trusted and only the domain manager shall the provers' pseudonym. For that purpose, the SE generates a random value r, encrypts the pseudonym following the elliptic curve integrated encryption scheme [20] (ECIES), and signs the message with the secret key of the pseudonym:

$$K_N = \mathrm{H}\left(r \cdot D_{id}\right) \tag{17}$$
$$\gamma_N = \mathrm{Enc}\left(K_N, N_{u,d}\right) \tag{18}$$
$$\sigma_N = \mathrm{Sign}\left(n_{u,d}, \mathrm{H}\left(\gamma_N\right)\right) \tag{19}$$

The SE outputs $(\gamma_N, \sigma_N, R := r \cdot G)$ to the domain verifier. As the message is encrypted with the public key of the domain manager, the domain verifier cannot acquire the pseudonym $N_{u,d}$. Hence, a single verifier can validate the eID and get domain-specific attributes but cannot by default link and trace the verifications. Even if domain verification of the same eID is requested multiple times, a domain verifier cannot link these verifications due to the randomness of r. Only the manager can decrypt that message, verify the signature and perform appropriate actions on that pseudonym (e.g. add bonus points to the loyalty program account). Note that disclosed attributes may still make verifications linkable to verifiers (see analysis in the next section).

6 Security & Privacy Analysis

Following up on our threat model, the proposed scheme prevents against:

- Forging identities. The security of our scheme relies on the security of the used ABC scheme as well as on the usage of an SE as a tamper-resistant storage for the ABC credentials and the identifier id_u. For that purpose, the SE on the provers' mobile device has a special security compartment (referred to as *security domain* [9]) that is under the control of a trusted eID issuer. Hence, a malicious prover cannot modify or forge sensitive data (i.e. identity, ABC credential, data attributes, etc.) without breaking the security of the SE. This is state-of-the-art technology for protecting sensitive information (e.g. SIM/bank cards) and also protects the integrity of the eID in cases where the mobile device gets stolen or malicious software is able to exploit the operating system of the prover device.
  A malicious prover could also try to establish a secure channel to the verifier and send invalid data attributes. Without the knowledge of an ABC key credential $\mathcal{C}_u$, the malicious prover cannot authenticate the ephemeral public keys in Step 4 of the domain enrollment protocol or in Step 2 of the verification protocol. Hence, without breaking the security of the SE or the used ABC scheme, it is infeasible for an attacker to forge an authenticated secure channel to the verifier and send invalid data attributes.
- Data leakage. The profiles as well as the TOFU database on the SE prevent uncontrolled attribute disclosure. The user stays in control of which data is sent to which verifier.
- Tracing identities. ABC credentials are designed to enable credential holders to attest the existence of certain signed attributes, without revealing the attribute itself. We make use of this mechanism to attest the validity of the eID without revealing any information about the eID holder. Hence, under the assumption that the used ABC mechanism protects against identity tracing, our proposed architecture is also secure against it.
  Note that the disclosed attributes may still make verifications of a user linkable and enable identity tracing. Additional privacy-preserving mechanisms, such as the attribute queries proposed in [12], could reduce the amount of revealed information in this case and further protect the privacy of the eID holders.

– Linking pseudonyms. Our pseudonym derivation relies on the usage of a secure one-way hash function. That is, a hash function that is resistant against preimage, second preimage and collision attacks [19]. Under this assumption, we argue that it is not feasible to link or deanonymize the pseudonyms of the prover without knowledge of the random and secret identifier id_u.

7 Evaluation

In the evaluation we use a 256-bit hash function, 256-bit elliptic curves (EC) for the creation of ECDSA signatures and 128-bit AES encryption. We will focus on the extensibility of the proposed architecture in terms of computation time as well as the required storage space on the SE.

7.1 Required Storage Space

Persistent and volatile memory are highly limited on an SE and therefore a limiting factor for smart card-based eID schemes. Hence, we briefly outline the required storage space of our architecture:

eID The SE of prover has to store the identifier id_u (we assume a size of 128 byte), the ABC key credential $\mathcal{C}_u$ (size depends on the specific ABC implementation and the required security level), and the data attributes of the eID holder da_u.

Domain Each enrolled domain adds one entry to the TOFU database, consisting of the domain public key D_{id} (33 bytes if point compression is supported, 65 bytes otherwise) and a profile that describes the accessible attributes for verifiers of that domain (we assume 4 bytes to control the disclosure of up to 32 attributes). There might also be additional attributes stored for each domain, hence, the exact size depends on the domain and can not be estimated. Nevertheless, with an overhead of 37 bytes (or 69 bytes without point compression) for each domain, we argue that our proposed system is very space efficient and allows the use of many services at the same time.

7.2 Computation Time

In this evaluation we mainly focus on the cryptographic commands that are executed by the SE. We implemented the involved steps of the domain enrollment and verification protocol and measured the computation time on a NFC SIM card and a Yubikey NEO (also a smart card based computing device), both with JavaCard version 3.0.1. The measurements for the NFC SIM where done on an OPPO N1 Mini with Android 4.3 using the Open Mobile API and the Yubikey NEO measurements where done with a Thinkpad T440s over the USB interface.

As the transfer speed between SE and other devices highly depends on the interface [11], we omit the transfer time in this evaluation. For that purpose, we send the required data in a preceding command, store it in temporary memory on the SE, and then execute and measure the actual command.

Table 2: Median computation time of the domain enrollment on the SE.

	NFC SIM	Yubikey NEO
Step 4	$1432 \pm 3\,\mathrm{ms}$	$672 \pm 1\,\mathrm{ms}$
Step 6	$620 \pm 5\,\mathrm{ms}$	$349 \pm 1\,\mathrm{ms}$

Table 3: Median computation time of the verification protocol on the SE.

	NFC SIM	Yubikey NEO
Step 2	$432 \pm 13\,\mathrm{ms}$	$163 \pm 0\,\mathrm{ms}$
Step 4	$970 \pm 5\,\mathrm{ms}$	$153 \pm 1\,\mathrm{ms}$
Step 6	$1048 \pm 7\,\mathrm{ms}$	$437 \pm 1\,\mathrm{ms}$

Domain Enrollment The measurement of the domain enrollment protocol (see Section 5.2) comprises of the following commands:

– Step 4 involves one signature verification, the pseudonym derivation (one hash and an elliptic curve (EC) point multiplication), generation of a new ephemeral key-pair and two signature creations.
– Step 6 involves one signature verification and the creation of the shared secret key (one EC point multiplication and a hash)

Table 2 lists the median results of 25 measurements performed on the two test cards. Overall, the steps involving the SE took 2053 ms for the NFC SIM and 1021 ms for the Yubikey NEO.

Verification The evaluation of the verification protocol (see Section 5.4) comprises of the following commands:

– Step 2 involves the generation of a new ephemeral key-pair and the creation of one signature.
– Step 4 involves two signature verifications and the creation of the shared secret key (one EC point multiplication and a hash)
– Step 6 is executed for domain verifiers and involves the creation of a public/private key-pair, a domain pseudonym derivation (one hash and an elliptic curve point multiplication), one AES encryption with a newly created secret key (one EC point multiplication and a hash) and a signature creation.

Table 3 lists the median computation time of 25 measurements on the test cards. Establishing the secure channel (Step 2 and 4) took overall 1402 ms on the NFC SIM and 315.5 ms on the Yubikey NEO. The derivation and encryption of the domain pseudonym (Step 6) took 1048 ms and 437 ms.

After these steps of the domain enrollment as well as the verification protocol, the SE established an authenticated secure channel with the remote entity. This channel can then be used to efficiently transfer attributes of the currently selected profile. That is, further communication to the SE only requires symmetric and is therefore rather efficient (see performance evaluation of the JCAlgTest project[3]).

Based on these results, we argue that our proposed architecture and the protocols are efficient for the user and can be performed on a computationally

[3]`https://www.fi.muni.cz/~xsvenda/jcalgtest/`

restricted device in reasonable time. That is, we assume that a user is not willing to wait for more then 2 seconds to finish a task. Especially the verification protocol, where two people (e.g. disco bouncer and the guest) are directly interacting with each other, should not exceed this limit. The evaluation shows that the channel establishment on the SE is below this time limit with some time left for the selective disclosure protocol and the data transfer. Only the enrollment on the NFC SIM took more than 2 seconds. However, this enrollment does not require direct human interaction and can therefore be performed in the background.

8 Conclusion

In this paper we proposed an architecture and protocols for a privacy-preserving and extensible mobile eID system for real-world identification. Our system enables eID holders to verify their identity in a similar fashion to regular ID cards and gives them the additional capability to use this eID for many public or private services (e.g. as loyalty card, public transport ticket). We evaluated the proposed architecture and protocols in terms of computation time as well as storage space and demonstrate that it can be executed in reasonable time on a computationally constrained device, such as smart cards.

Acknowledgments

This work has been carried out within the scope of *u'smile*, the Josef Ressel Center for User-Friendly Secure Mobile Environments, funded by the Christian Doppler Gesellschaft, A1 Telekom Austria AG, Drei-Banken-EDV GmbH, LG Nexera Business Solutions AG, NXP Semiconductors Austria GmbH, and Österreichische Staatsdruckerei GmbH.

References

1. G. Alpár and J.-H. Hoepman. A Secure Channel for Attribute-based Credentials: [Short Paper]. In *Proceedings of the 2013 ACM Workshop on Digital Identity Management*, DIM '13, pages 13–18. ACM, 2013.
2. G. Alpár and B. Jacobs. Credential design in attribute-based identity management. In *Bridging distances in technology and regulation, 3rd TILTing Perspectives Conference*, pages 189–204, Apr. 2013.
3. P. Bichsel, J. Camenisch, T. Groß, and V. Shoup. Anonymous Credentials on a Standard Java Card. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 600–610. ACM, 2009.
4. P. Bichsel et al. Architecture for Attribute-based Credential Technologies - Final Version. Deliverable D2.2, Aug. 2014.
5. J. Camenisch and A. Lysyanskaya. An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation. In *Advances in Cryptology – EUROCRYPT 2001*, pages 93–118. Springer Berlin Heidelberg, May 2001.

6. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *International Conference on Security in Communication Networks*, pages 268–289. Springer Berlin Heidelberg, 2002.
7. D. Chaum. Security without identification: Transaction systems to make big brother obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.
8. FIDO Alliance. FIDO UAF Protocol Specification v1.1, Feb. 2017. Implementation Draft.
9. GlobalPlatform. Card Specification v2.3, Oct. 2015. Public Release.
10. J. Hajny and L. Malina. Unlinkable Attribute-Based Credentials with Practical Revocation on Smart-Cards. In *Smart Card Research and Advanced Applications*, pages 62–76. Springer Berlin Heidelberg, Nov. 2012.
11. M. Hölzl, R. Mayrhofer, and M. Roland. Requirements for an Open Ecosystem for Embedded Tamper Resistant Hardware on Mobile Devices. In *Proceedings of International Conference on Advances in Mobile Computing & Multimedia*, MoMM '13, pages 249–252. ACM, 2013.
12. M. Hölzl, M. Roland, and R. Mayrhofer. Real-World Identification: Towards a Privacy-Aware Mobile eID for Physical and Offline Verification. In *Proceedings of the 14th International Conference on Advances in Mobile Computing and Multi Media*, MoMM '16, pages 280–283. ACM, 2016.
13. IBM Research. Specification of the Identity Mixer Cryptographic Library v2.3.0, Apr. 2010. Research Report.
14. J. L. Jensen. Smartphone feasibility analysis. Deliverable D4.4, May 2014.
15. A. Lehmann et al. Survey and Analysis of Existing eID and Credential Systems. FutureID Deliverable D32.1, Apr. 2013.
16. G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger. NFC Devices: Security and Privacy. In *Third International Conference on Availability, Reliability and Security (ARES'08)*, pages 642–647. IEEE, 2008.
17. T. Nyman, J.-E. Ekberg, and N. Asokan. Citizen electronic identities using TPM 2.0. In *Proceedings of the 4th International Workshop on Trustworthy Embedded Devices*, pages 37–48. ACM, 2014.
18. C. Paquin. U-prove technology overview v1.1, Apr. 2013. Technical Report, Microsoft Corporation Draft Revision.
19. P. Rogaway and T. Shrimpton. Cryptographic hash-function basics: Definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In *Fast Software Encryption*, pages 371–388. Springer Berlin Heidelberg, 2004.
20. V. Shoup. A proposal for an ISO standard for public key encryption (version 2.1). IACR E-Print Archive 112, 2001.
21. P. Vullers and G. Alpár. Efficient Selective Disclosure on Smart Cards Using Idemix. In *Policies and Research in Identity Management*, IFIP Advances in Information and Communication Technology, pages 53–67. Springer Berlin Heidelberg, Apr. 2013.