EAI Endorsed Transactions
on Mobile Communications and Applications
Research Article ICST.ORG

A Privacy-Preserving NFC Mobile Pass for Transport Systems

Ghada Arfaoui[1,5], Guillaume Dabosville[2], Sébastien Gambs[3], Patrick Lacharme[4], Jean-François Lalande[3,5,*]

[1]Orange Labs, F-14066 Caen, France
[2]Oberthur Technologies, F-92700 Colombes, France
[3]Université de Rennes 1, Inria, SUPELEC, CNRS, IRISA (UMR 6074), F-35065 Rennes, France
[4]Laboratoire GREYC (Unicaen, Ensicaen, CNRS), UMR 6072, F-14032 Caen, France
[5]INSA Centre Val de Loire, Univ. Orléans, LIFO EA 4022, F-18020 Bourges, France

Abstract

The emergence of the NFC (Near Field Communication) technology brings new capacities to the next generation of smartphones, but also new security and privacy challenges. Indeed through its contactless interactions with external entities, the smartphone of an individual will become an essential authentication tool for service providers such as transport operators. However, from the point of view of the user, carrying a part of the service through his smartphone could be a threat for his privacy. Indeed, an external attacker or the service provider himself could be tempted to track the actions of the user. In this paper, we propose a privacy-preserving contactless mobile service, in which a user's identity cannot be linked to his actions when using the transport system. The security of our proposition relies on the combination of a secure element in the smartphone and on a privacy-enhancing cryptographic protocol based on a variant of group signatures. In addition, although a user should remain anonymous and his actions unlinkable in his daily journeys, we designed a technique for lifting his anonymity in extreme circumstances. In order to guarantee the usability of our solution, we implemented a prototype demonstrating that our solution meets the major functional requirements for real transport systems: namely that the mobile pass can be validated at a gate in less than 300 ms, and this even if the battery of the smartphone is exhausted.

Received on 28 February 2014; accepted on 16 September 2014; published on 28 December 2014

Keywords: DAA, mobile pass, NFC, privacy, smartcard

doi:10.4108/mca.2.5.e4

1. Introduction

Nowadays, smartphones have become essential devices in the life of individuals, as they centralize and gather many personal data through the installed applications. In the future, most of the mobile phones and smartphones will integrate NFC (*Near Field Communication*) capabilities, which is a form of proximity contactless communication in which data are exchanged between a reader and a target device.

The deployment of the NFC technology has been accelerating during the last years and is accompanied by the emergence of new mobile services such as mobile ticketing [1, 2], mobile payment [3], access control [4] or loyalty card [5], just to name a few. On one hand, the widespread adoption of the NFC technology within smartphones brings new challenges in terms of privacy and security. In particular, currently and even more in the future, sensitive and personal data are stored and managed through a smartphone. These personal data are often generated by an individual but are not necessarily under his control. Thus, there is a risk that these data could be used for fraudulent purposes or secondary commercial uses. Moreover, these data may leak due to threats such as malwares, inference attacks or the eavesdropping of communications exchanged between the smartphone and external devices. These

*Corresponding author. Email: jean-francois.lalande@insa-cvl.fr

threats are magnified by the fact that most of mobile devices are also connected to the Internet.

On the other hand, the computational power of smartphones has increased dramatically, now supporting advanced authentication techniques and cryptographic primitives such as the ones provided by *Privacy Enhancing Technologies* (PETs). Most of PETs are based on two fundamental privacy principles: *data minimization* and *data sovereignty*. The data minimization principle states that only the information necessary to complete a particular application should be disclosed (and no more)[1] while the data sovereignty principle considers that the data related to an individual belongs only to him (*i.e.*, he should be able to control how these data are used and for which purpose). This control requires to provide users with transparency on the use of these data as well as the possibility to revoke his consent.

Moreover, smartphones often have access to a tamper-resistant chip, called the *Secure Element*, which ensures that sensitive data like cryptographic keys and credentials are processed and stored in a secure manner (*i.e.*, such that both confidentiality and integrity objectives are achieved). Such a secure element can be directly embedded into a smartphone (see [7] for an example of such a secure element) or be available as an external module like a UICC/SIM card [8].

Using a personal device for deploying a service associated to the identity of the owner with respect to his privacy is a challenging task. For instance, a contactless service may involve a monthly subscription to a public transport system, an electronic ticket for a concert or some personal information stored in the smartphone. If an unauthorized entity is able to follow all the digital traces left behind by a user during these interactions with contactless services, then this entity could potentially build a very detailed profile of this individual, thus causing a privacy breach.

In this paper, we focus on the development of a mobile transport service for NFC-enabled device. This service is very sensitive with regard to privacy, as it requires a proof related to 1) identity of the customer as well as 2) his attributes in relation with the purchased product. In transport systems, these attributes are controlled and certified by the service provider, before being integrated within a user profile stored on his smartphone. In this situation, the user's personal data are directly exposed to surveillance from the service provider. Moreover, there is a high risk that all the actions of the user performed within the transport system will be traced. Therefore, it is extremely important to develop privacy-preserving mobile contactless services enabling a user to control and limit the digital traces left behind by his smartphone.

Our main contributions can be summarized as follows:

- We propose a privacy-preserving contactless mobile service, in which a user's identity cannot be linked to his actions when using the transport system. In particular, an eavesdropper and the transport provider are not able to track the actions of the user. To achieve this objective, we have developed a privacy-preserving protocol for a contactless mobile pass, whose security is based on the combination of a secure element in a smartphone and on a privacy-enhancing cryptographic protocol.

- Even if a user should remain anonymous and his actions unlinkable in his daily journeys, we have designed a technique for lifting his anonymity in extreme circumstances. This exceptional procedure can be called for instance under the order of a judge, in case of a fraud or a murder.

- The efficiency of the developed solution is also of paramount importance as according to the requirements of the current standards, the contactless validation of an electronic mobile pass at a gate should be carried out in less than 300 ms and in an offline manner. In addition, our solution has the additional benefit that it works even if the battery of the smartphone is exhausted.

The remainder of the paper is organized as follows. First in Section 2, we describe the main functionalities of electronic ticketing (e-ticketing) systems, and then in Section 3, we review the state-of-the-art of e-ticketing systems. Studying the existing ticketing systems helps to understand the properties needed to achieve a secure and privacy-preserving mobile transport pass as summarized in Section 4. Afterwards, after introducing the required cryptographic tools in Section 5, we describe the proposed protocol in Section 6 and we discuss its security in Section 7. Finally, we give details about the implementation prototype in Section 8, before concluding.

2. E-ticketing approach to public transport systems

Before describing the state-of-the-art of existing approaches for e-ticketing in Section 3, we review the functional specifications of e-tickets and mobile passes, respectively, in Sections 2.1 and 2.2. Finally in Section 2.3, we summarize the main security and privacy properties of such systems.

[1] This principle is a direct application of the legitimacy criteria defined by the European data protection directive (Article 7, [6]).

2.1. Functional specifications

When using an e-ticket in the context of a public transport system, a user interacts with several entities during the different phases of the process. We briefly review these steps, and refer the reader to [9] for more details. The different entities involved in an e-ticketing system are the following ones:

- The *user* benefits from a transport service personalized according to a set of personal attributes stored in his profile. In this paper, we make the assumption that the user owns an smartphone equipped with NFC capabilities that contains a secure element.
- The *service operator* is in charge of transporting the user across its network.
- The *application retailer* initializes the smartphone (*i.e.*, the applications and the secure element) such that it becomes possible to install further products on it.
- The *product retailer* stores the product data on the user's smartphone, thus personalizing the secure element and the applications.
- The *validator* can initiate a control with the smartphone of the user when he is traveling in the transport system, in order to check the authenticity and the validity of his pass.

Note that the service operator, application retailer and product retailer are usually different entities. For instance in our context, the application retailer could be a telecommunication operator that installs the application into its own secure element, which we refer to as the UICC (for *Universal Integrated Circuit Card*) but is also known as the SIM (for *Subscribe Identity Module*) card. Then, the product retailer could be an external seller of e-tickets that needs the involvement of the application retailer in order to load these tickets into the UICC/SIM card or that uploads itself the e-tickets directly into the UICC/SIM card by relying on cryptographic material provided by the application retailer. In this paper, we assume that the applications and the secure element are setup correctly and managed by these entities.

We now present the different phases of an e-ticketing system.

Application personalization. During this phase, a user gives a proof of his identity (*e.g.*, a student card, an identity card or a passport) to the service operator. If this verification succeeds, the service operator and the application retailer load a personalized application on the smartphone of the user. This application will be used to handle the product.

Product registration. A user chooses a product (*e.g.*, a monthly transport pass or a one time e-ticket) and the related personal attributes (*e.g.*, the age of the user) in order to load the product into his smartphone. Additionally, if such a choice is possible in the current public transport system, the user chooses the geographical area of his pass and the associated validity period, before paying for the product. The product is then loaded by the product retailer on the user's smartphone. During the product distribution phase, no verification is performed on the user identity and attributes. Indeed, as the product may be transferred to another person before its use (*e.g.*, a parent pays for the monthly pass of his child), the verification of attributes should be done later when the product is activated or used for the first time.

Product validation. The user shows his smartphone at the entrance gate of the public transport system, and potentially also at the exit gate, in order to validate the product. To achieve this, the validator initiates a communication with the user. Then, the validator checks the authenticity of the product and grants or denies access to the user. If a validation conflict occurs (two products are eligible to be validated), then the user has to be involved in selecting the adequate product. Finally, the smartphone and the validator record the event. The validator must implement a list of the revoked products in order to reject the users that have been banned as well as an anti-passback feature in order to block two validations of the same product occurring at the same gate within a short period of time. Furthermore to be considered usable, the total transaction time should not exceed 300 ms [10, 11].

Travel control. The service operator may control a user while he travels inside the transport network in order to verify that he possesses the adequate product. Even if it is important, this part of the use case is out of the scope of the current paper.

2.2. Specificities of the mobile pass for transport systems

In one of the first papers about e-ticketing [12], the authors compare the properties of electronic cash solutions to the ones needed for e-ticketing. However, they also show that some of these properties are relevant only for e-tickets and others only for transport passes. For instance, the transferability property applies to e-tickets but not to transport passes (at least once they have been validated). Thus, we believe that taking the approach of e-tickets or the approach of mobile passes for public transport services has a significant impact on the security properties, the design of the proposed solution and the expected performances.

In this paper, we focus on the design of a mobile pass for accessing the public transport system. The mobile pass should meet the following functional requirements:

- The mobile pass is paid for a certain duration of time. Thus it expires after the end of its validity period.
- The mobile pass enables to access specific geographical zones of the transport system.
- Contrary to e-tickets, the mobile pass can be used to perform an infinite number of trips.

2.3. Security and privacy requirements of e-ticketing

Among the major requirements for e-tickets and mobile passes are that the system behaves correctly, in a secure manner and that it should be impossible to forge fake tickets. Other security requirements that can be considered include the non-transferability of a ticket to a third party (which can be desirable or not depending of the use case), the inclusion of an expiration date or the fact that it can be used only once. In addition, respecting the privacy of a user requires that he remains anonymous in his daily use of the transport system. However, under exceptional circumstances (*e.g.*, a fraud or a murder), it should be possible to lift his anonymity, thus providing a form of revocable anonymity.

In addition, a privacy-preserving e-ticketing system should also ensure the unlinkability of the actions of a user. In particular, in the context of transport systems, the service provider might be tempted to link all the trips performed by the same user, even if these trips are "anonymous". If the different actions of the user can be linked, then there is a high risk of profiling. In addition, some of these actions combined together might play the role of *quasi-identifiers*, thus leading to a potential de-anonymization of the user. For instance, existing technologies such as U-prove may fail to provide the unlinkability property if the user does not go online regularly to refresh his pseudonym [13]. This was further confirmed by a study of Vives-Guasch and co-authors [14], which shows that most of current proposals in the literature use pseudonyms in order to achieve revocable anonymity but do not prevent the tracking of these pseudonyms.

3. State-of-the-art

Electronic mobile passes have been seldomly studied in the literature as most of the papers about transport services focus on the use of e-tickets. In this section, we review the state-of-the-art about e-ticketing solutions for transport systems, focusing in particular on the privacy aspects of such solutions. First, we describe the current existing solutions for mobile ticketing in transport systems in Section 3.1, before presenting in Section 3.2 the privacy-preserving solutions that have been recently proposed. This study has helped us to identify the important security and privacy properties needed to build a privacy-preserving mobile pass for transport systems.

3.1. Contactless mobile ticketing

In 2007, the CALYPSO standard [10, 15] (ISO 24014-1:2007 norm) introduced the contactless aspects across a European consortium composed of transport operators such as the Belgium STIB and the French RATP. CALYPSO specifies the details of all the transactions related to e-ticketing for contactless transport services from the purchase of the tickets to their uses. This standard is very precise, in particular when describing operations such as the card authentication, the validator authentication and the messages exchanged between these two entities. Moreover, performance issues such as the computational time are also included in this standard, particularly for contactless smartcards.

In 2008, the German federal railways Deutsche Bahn was one of the first to propose the service Touch and Travel (T&T) [16]. The T&T service is one of the first contactless mobile ticketing services. A user must first subscribe to the service by showing his identity along with the information about his bank account. Afterwards, he gets a user number and a PIN that allows to use the T&T application. To travel from a station A to a station B, the user must start the T&T application upon departure by using a touchpoint if one is available at the station, relying on the GPS as captured by his smartphone or based on the location based on the network cell. Once he has reached his destination (*e.g.*, station B), the user must indicate to the application that this is the end of his trip. When controlled, the T&T application generates and displays a QR code to prove that the payment for the trip has been done.

Regarding the privacy aspect of T&T, Pirker and Slamanig [17] have pointed that T&T stores the list of all the recent trips in a centralized database. Thus, the company running T&T or an adversary that is able to breach the security of the information system can easily trace all the whereabouts of the users of T&T. Moreover, as T&T requires the activation of the GPS to capture the location of the departure or arrival station, there is a risk that this information might leak (*e.g.*, through a malware).

Even more recently, commercial mobile NFC transport and payment solutions have been deployed in Chili [18] and in Honk Kong [19, 20] with the Octopus card. The Octopus system suffers also from the linkability issue, as it enables the tracking of a user with a unique ID that is used at every transaction [21].

To summarize, the introduction of the NFC technology, while easing the use of services such as payment or e-ticketing also brings new challenges regarding the privacy of users. In particular, the contactless transactions can be listened by eavesdroppers and the smartphone of a user can be challenged by a fake validator. Thus, designing a secure protocol that also respects the privacy of users is even more critical.

3.2. Privacy-preserving m-ticketing solutions

In [11], Tamrakar, Ekberg and Asokan propose an e-ticketing scheme for NFC smartphones, which has two variants of the protocol for the verification phase. These protocols are based on short lived-certificates and a challenge/response phase. Both variants assume that all cryptographic operations are executed inside a secure element. This solution complies with the functional requirement that the identity verification has to been done in less than 300 ms. However, the authors have only considered as a threat to privacy an external eavesdropper spying on the communications but not the service provider. Indeed, in their proposal the service provider can easily follow and link the users' actions.

In the follow-up papers [22, 23], Tamrakar and Ekberg propose a contactless m-ticketing transport service. The main idea of this service is that a user, who has already subscribed to the service, can buy some credits before purchasing tickets with them. The ticketing application is authorized to locally generate a ticket before then charging the user account. Like for T&T service, the user has to specify the departure and arrival stations (*i.e.*, pay-as-you-go approach). Depending on the trust assumptions that are made, the authors propose two architectures relying or not on a trusted execution environment. While this solution complies with the main functional requirements of ticketing service, the privacy requirements are still not achieved. Indeed the user personal data, such as his identifier and location, are protected against an external attacker but revealed to the service provider.

In [24], the authors have analyzed the efficiency of a selective disclosure protocol in the latest smartphones. In a nutshell, a selective disclosure protocol allows a user to reveal that he possesses some property linked to his identity, but without revealing his identity itself or any unnecessary information. With respect to the ticketing use case, they used a one time-show protocol, which is the Brands DSA-based protocol [25, 26]. Thanks to this protocol, during the product validation phase, the user is able to prove the validity of his credentials without disclosing them. While this protocol respects the privacy of users, it also introduces an important delay.

In [27], Isern-Deya and his co-authors propose an automatic fare collection system for temporal and spatial services, which is based on the BBS group signature [28, 29]. Thus, they provide revocable anonymity and unlinkability for the users of the service. The main drawback of this solution is that although it has been implemented on a smartphone, the waiting time at the entrance and exit of the transport network is still prohibitive (*i.e.*, in the order of seconds).

In this section, we have given an overview of existing solutions for NFC ticketing services. Most of them do not address the anonymity and unlinkability issues, and the few who do are not yet efficient enough to cope with the constraint of running in less than 300 ms.

4. Security and privacy requirements for m-ticketing

Taking into account the properties described and discussed in [9, 12], we want to design a privacy-preserving NFC mobile pass for transport systems that fulfills the following properties:

- *Security properties*: integrity and authentication (IT in [9]), (ATH in [9]), non-repudiation (NRO in [9]) and unforgeability (UNF in [9]).
- *Privacy properties*: revocable anonymity (S-RAN in [9]) and unlinkability.
- *Usability properties*: efficiency (EFF in [9]), possibility of running offline (OFF in [9]) and contactless communication.

First, the service should meet the standard security requirements of a transport service. In particular, it should not be possible for an adversary to forge a fake pass (IT, UNF). Similarly, a user should be able to prove the validity of his pass and the service provider should not be able to deny having issued the pass (ATH, NRO).

Second, a user should be anonymous in his daily routine when using the transport service but his anonymity may be lift by a dedicated trusted third party under exceptional circumstances (S-RAN). Once a user has been de-anonymized, his rights can be revoked by the transport authority in order to disable his mobile pass. In addition, the user's actions should be unlinkable, especially from the point of view of the service provider.

Finally, the mobile pass should be efficient and always available. Indeed, the validation time should not exceed 300 ms (EFF) and the validation process can be carried out in an offline manner (OFF). Additionally, the pass should be embedded in an NFC enabled smartphone and the validation should be achieved in a contactless manner.

5. Cryptographic building blocks

In this section, we introduce the notations and the cryptographic tools used later in the description of the protocol.

5.1. Notation

We use the notation:

$$x \xleftarrow{\$} X \tag{1}$$

to denote that x is chosen uniformly at random from the set X, while the notation:

$$x \leftarrow y \tag{2}$$

means that the value y is assigned to the variable x.

$\mathbb{Z}_p$ denotes the set of positive integers x less than *p-1*.

$\{0,1\}^*$ denotes the set of all finite length binary strings.

$\{0,1\}^k$ denotes the set of all binary strings of length k.

$\{0,1\}^\gamma$ denotes the set of all binary strings of length γ.

Let $\mathbb{G}$ denote a cyclic group of prime order p. The identity element of $\mathbb{G}$ will be written as $1_{\mathbb{G}}$.

Let $\mathbb{G}_E$ denote an additive cyclic group of order p over an elliptic curve E. The group law of $\mathbb{G}_E$ is denoted by "+". $[n]P$ denotes the operation that takes a positive integer n and a point P on the curve E as input and produces as output another point Q on the curve E, in which:

$$Q = [n]P = \underbrace{P + P + \ldots + P}_{n-1 \text{ times}} \tag{3}$$

The operation satisfies $[0]P = O_E$ (the point at infinity), and $[-n]P = [n](-P)$.

Our mobile pass protocol uses a bilinear mapping $e : \mathbb{G}_1 \text{ x } \mathbb{G}_2 \rightarrow \mathbb{G}_T$, in which $\mathbb{G}_1$ and $\mathbb{G}_2$ denote two additive cyclic groups of order p over an elliptic curve and $\mathbb{G}_T$ a multiplicative cyclic group of order p. In addition of being efficiently computable, the mapping e should satisfy the following two properties :

$$\forall X_1 \in \mathbb{G}_1, X_2 \in \mathbb{G}_2, a, b \in \mathbb{Z}_p,$$

$$e([a]X_1, [b]X_2) = e(X_1, X_2)^{ab} \tag{4}$$

$$\forall X_1 \neq 1_{\mathbb{G}_1}, X_2 \neq 1_{\mathbb{G}_2}, e(X_1, X_2) \neq 1_{\mathbb{G}_T} \tag{5}$$

Being efficiently computable, for e, means that it exists an efficient algorithm for computing $e(P,Q)$ for any $(P,Q) \in G_1 \text{ x } G_2$.

5.2. Cryptographic tools

The basic cryptographic building blocks that are needed for the design of our protocol are *commitment schemes*, *signatures of knowledge*, *group signatures*, *DAA* and *Camenisch-Lysyanskaya signature schemes* (CL-signature in short).

Commitment schemes. In a nutshell, a commitment scheme allows a party to commit to a tuple $(x_1, x_2, \ldots, x_n)$ of *secret* values to another party. Later, the commitment can be opened by having the issuer of the commitment send additional information to the party to which he has committed to. The commitment does not reveal any information on the secret values to the other party (hiding property) and prevents the committing party from changing the values that he has committed to at a later stage (binding property).

Zero-knowledge proofs and signatures of knowledge. In a zero-knowledge proof, a prover convinces interactively a verifier that he knows a witness ω that a predicate P is true without revealing any further information on ω. Zero-knowledge proofs can also be made non-interactive (*i.e.*, thus becoming signatures of knowledge) by using the Fiat-Shamir heuristic [30]. In the sequel, we will use the notation:

$$ZKP[(\alpha, \beta, ...) : P] \tag{6}$$

to denote a signature of knowledge proving that the prover knows a tuple $(\alpha, \beta, ...)$ of secret values satisfying the predicate P. In this notation, the Greek letters correspond to the secret knowledge and the other letters denote public parameters between the prover and the verifier. For example,

$$ZKP[\alpha : Q = [\alpha]P] \tag{7}$$

denotes a non-interactive proof of knowledge of the discrete logarithm of Q in the base P (see [31] for a detailed description of this signature of knowledge which will be implicitly used in our m-pass protocol). Q can be seen as a commitment of the secret value α with respect to the base P.

Group signature and DAA. Group signature schemes have been introduced by Chaum and van Heyst [32] in the nineties. In contrast to classical digital signatures, they provide anonymity to the signer in the sense that the verifier can only tell that a member of the group generated the signature without being able to recover the signer's identity. However in case of a dispute, the anonymity of a group signature can be lifted by one or several designated trusted authorities (called escrow agents).

A variant of group signatures, called list signatures, has been introduced by Canard and his co-authors [33]. With a list signature scheme, it becomes possible, when specific conditions are met, to link signatures produced by a member of the group. In particular, this link becomes possible if the signatures have been produced during a given sequence of time (*i.e.*, a specific time period). A similar technique has later been used in [34] and is called Direct Anonymous Attestation (DAA). The main difference between DAA and list

signatures is that in DAA, the role of the signer is split between a main signer with limited computational and memory capabilities (*e.g.*, a Trusted Platform Module (TPM) or a UICC card), and an assistant signer, who is computationally more powerful but is considered less secure (*e.g.*, a mobile phone containing the UICC).

Camenisch-Lysyanskaya signature schemes. These signature schemes, proposed by Camenisch and Lysyanskaya [35], are equipped with additional protocols. One of these protocols allows a signature to be issued on the messages that are not known to the signer, but to which the signer only knows a commitment. Informally, in a protocol for signing a committed value, we have a signer with public key pk, and the corresponding secret key sk, and a user who queries the signer for a signature. The common input to the protocol is a commitment C on secret values $(x_1, x_2, \ldots, x_n)$ known only by the user. At the end of this protocol, the user obtains a valid CL-signature = $\mathsf{Sign}(x_1, x_2, \ldots, x_n)$ and the signer learns nothing about $(x_1, x_2, \ldots, x_n)$.

Another protocol allows to prove knowledge of a signature on a tuple of messages $(x_1, x_2, \ldots, x_n)$ without releasing any information on the corresponding signature. Each message can either be revealed to the verifier, sent in a committed form or he might have no information about it. For instance, it is possible to prove the knowledge of a CL-signature on committed values. Another appealing feature of some of these CL-signature schemes is that the signatures they produced can be randomized. Indeed, given a valid CL-signature σ on a message m, anyone can compute σ', another valid signature on m. Moreover, σ and σ' are unlinkable under the *Decision Diffie-Hellman assumption* (*i.e.*, given σ and σ' (but not m), it is hard to decide whether these two signatures have been issued on the same message). CL-signatures have been widely used to design anonymous credentials or DAA-like schemes.

6. Anonymous and untraceable mobile pass

We propose an m-pass application providing the same functional service as a classical transport pass while preserving the privacy of users. The architecture of the m-pass application is illustrated in Figure 1. This architecture consists of two components: an m-pass cardlet and an m-pass user application. The m-pass cardlet, which is running inside the secure element, manages the attributes and credentials of the user and communicates through the NFC controller. In our setting, the secure element used is a smartcard (*e.g.*, the UICC). The m-pass user application is an Android application dealing with the interactions with the smartphone owner.

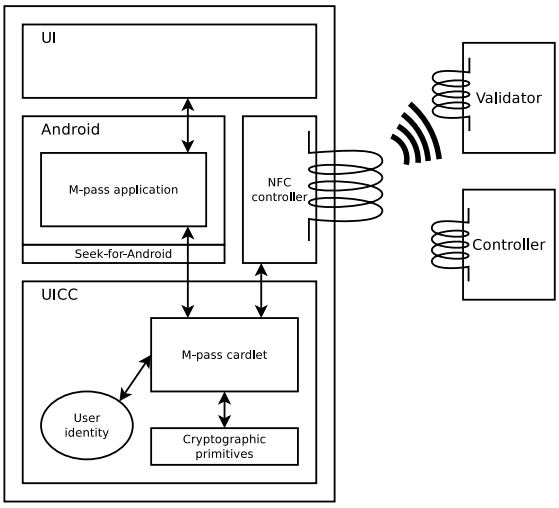

Figure 1. Overview of an m-pass architecture.

6.1. Privacy-preserving protocol for mobile pass

The main idea of our solution is to use a DAA scheme in order to enable an enrolled user to be anonymous inside the group of other subscribers of the m-pass service. More precisely, our solution is based on the DAA scheme [34], which we have adapted in order to handle revocation features. For the sake of simplicity in this section, we define our protocol by taking into account only one group: the group of persons having an m-pass for using the public transport system. However, we treat the generic case of several groups in Section 6.5.

Three actors are involved in this protocol: the user who uses the m-pass application during his journeys, the transport authority who is the manager of the transport service and the opening authority who is the only entity able to retrieve the identity of an anonymous user. The opening authority is completely independent of the transport authority. If one does not trust a single authority, the role of the opening authority may be split between several authorities in such a way that they should all agree and cooperate before unlocking the identity of an anonymous user.

The protocol is composed of three phases:

- *Setup*. In this phase, the transport authority initializes the public parameter of the group and all the entities generate their keys.
- *User registration*. In this phase, a user contacts the transport service for joining the group of users. In addition, the user must also pay for his m-pass (the payment procedure itself is out of the scope of this paper).
- *M-pass validation*. During this phase, the user has to interact with the validator in order to prove the authenticity of his m-pass before accessing the transport service.

Setup. The following two steps occur during the initialization.

Initialization of the public parameters of the group. The transport authority considers two hash functions, $\mathcal{H}$ and $\mathcal{H}_1$, modeled as random oracles:

$$\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^k, \quad (8)$$
$$\mathcal{H}_1 : \{0,1\}^* \rightarrow \mathbb{G}_1. \quad (9)$$

Then, he picks up randomly G_1 (respectively G_2) from $\mathbb{G}_1$ (respectively $\mathbb{G}_2$):

$$G_1 \xleftarrow{\$} \mathbb{G}_1, \quad (10)$$
$$G_2 \xleftarrow{\$} \mathbb{G}_2. \quad (11)$$

The public parameters of the group are gpp such that:

$$gpp \leftarrow (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathcal{H}, \mathcal{H}_1, e, G_1, G_2). \quad (12)$$

(recall that e is an elliptic curve and $\mathbb{G}_T$ is a multiplicative cyclic group of order p, *cf.* Section 5).

Initialization of the cryptographic keys of the other entities. The transport authority generates his public and private keys (pk_t, sk_t) to sign messages such that:

$$sk_t \leftarrow (x, y) \xleftarrow{\$} \mathbb{Z}x\mathbb{Z}, \quad (13)$$
$$pk_t \leftarrow (X, Y) \leftarrow ([x]G_2, [y]G_2). \quad (14)$$

Similarly, the opening authority generates his pair of public and private signature keys (pk_o, sk_o). In consequence, the group public key pk_g is set up such that:

$$pk_g = (gpp, pk_t, pk_o) \text{ and} \quad (15)$$

where gpp is the public parameter of the group. The keys pk_t and pk_o) are used to verify signatures issued respectively by the transport authority and the opening authority.

Regarding the user, his m-pass cardlet generates his private key sk_u, which is randomly picked from $\mathbb{Z}$.

User registration. During the registration phase, described in Table 1, the m-pass cardlet establishes two secure channels with the opening and transport authorities.

First, the registration of the user to the opening authority consists of two steps.

1) The m-pass cardlet sends two commitments (C_1, C_2) on the private key sk_u of the user such that:

$$C_1 = [sk_u]G_1, \quad (16)$$
$$C_2 = [sk_u]G_2. \quad (17)$$

Registration within the database of the opening authority		
Smartcard cardlet		**Opening authority**
Public input: pk_g Private Input: sk_u		Public Input: pk_g Private Input: sk_o, DB_o
$(C_1, C_2) \leftarrow$ $([sk_u]G_1, [sk_u]G_2)$	$\xrightarrow{C_1, C_2}$	If $e(C_1, G_2) = e(G_1, C_2)$ Then:
	$\xleftarrow{\mu}$	$\mu \leftarrow Signature(sk_o, C_1)$ **Store** (C_1, C_2, μ) **in** DB_o
Registration within the database of the transport authority		
Smartcard cardlet		**Transport authority**
Public input: pk_g, μ Private Input: sk_u		Public Input: pk_g Private Input: sk_t, DB_t
	$\xrightarrow{C_1, \mu}$	If verify(pk_o, μ) then computes a signature on C_1: $a \xleftarrow{\$} \mathbb{Z}_p$ $A \leftarrow [a]G_1$ $B \leftarrow [y]A$ $C \leftarrow [x](A + D)$
	$\xleftarrow{A,B,C,D}$	$D \leftarrow [a.y]C_1$ **Store** $(user, C_1)$ **in** DB_t

Table 1. Protocols for the registration phase.

2) The opening authority verifies that these computations are correct. More precisely, the opening authority checks if $e(C_1, C_2) = e(G_1, G_2)$. If this verification succeeds, it means that C_1 and C_2 are commitments on the same secret value (sk_u here). In this case, the opening authority sends back a signature μ of C_1 and stores the triplet (C_1, C_2, μ) in the database DB_o, which has to remain private and secure.

Then, upon receiving the signature μ, the m-pass cardlet sends (C_1, μ) to the transport authority, which then checks the validity of the signature using pk_o. If the signature is valid, the transport authority computes and sends a CL-certificate [35] (A, B, C, D) on the private key of the user (recall that CL-signature schemes allow a signature to be issued on the messages that are not known to the signer, but to which the signer only knows a commitment). The transport authority also saves (ID_{user}, C_1) inside his database DB_t.

The databases DB_o and DB_t will be used only for de-anonymizing or blacklisting of users as detailed later in Sections 6.3 and 6.4. Once the registration phase is finished, the user is registered in both the transport and opening authorities databases. In addition, he is now able to use his m-pass application to access the transport service.

M-pass validation. The validation of the m-pass, described in Table 2, is used by the user to prove his legitimacy to use the transport service without disclosing his identity. The process is split into two parts: the first part (called precomputation in Table 2) can be performed before the second part, which is the

Precomputations: randomization of the CL-Certificate		
Smartcard cardlet		**Smartphone**
Public input: pk_g Private Input: sk_u		Public Input: pk_g
	$\xrightarrow{A,B,C,D}$	$l \xleftarrow{\$} \mathbb{Z}_p$ $(R, S) \leftarrow ([l]A, [l]B)$
$k \xleftarrow{\$} \mathbb{Z}_p$ $R_2 \leftarrow [k]S$	$\xleftarrow{R,S,T,W}$	$(T, W) \leftarrow ([l]C, [l]D)$

M-pass validation		
Smartcard cardlet		**Gate**
Public input: pk_g, R, S, T, W, R_2 Private Input: sk_u, k		Public Input: pk_g, bsn
	$\xleftarrow{rc,bsn}$	$\text{rc} \xleftarrow{\$} \{0,1\}^\gamma$
Compute the signature σ $J \leftarrow \mathcal{H}_1(bsn)$ $R_1 \leftarrow [k]J$ $K \leftarrow [sk_u]J$ $c \leftarrow \mathcal{H}(J, K, R, S, T, W, R_1, R_2, rc)$ $s \leftarrow k + c.sk_u \ mod p$		
	$\xrightarrow{\sigma=(R,S,T,W,J,K,c,s)}$	Verify the signature σ: $J' \leftarrow \mathcal{H}_1(bsn)$ $R_1 \leftarrow [s]J - [c]K$ $R_2 \leftarrow [s]S - [c]W$ $e(R, Y) \stackrel{?}{=} e(S, G_2)$ $e(T, G_2) \stackrel{?}{=} e(R + W, X)$ $c \stackrel{?}{=}$ $\mathcal{H}(J', K, R, S, T, W, R_1, R_2, rc)$ If the user is black-listed, then access is denied Otherwise, access is granted

Table 2. Protocols for the validation phase.

validation itself at the gate (called m-pass validation in Table 2).

We now give an overview of the challenge/response protocol. After the registration phase, the m-pass cardlet of the user stores as secrets, the private key of the user sk_u and a CL-certificate on this user's secret, (A, B, C, D). The DAA scheme enables to start partially the computation of the DAA before receiving the random challenge. First, the m-pass cardlet sends the CL-certificate (A, B, C, D) to the m-pass application of the smartphone. Then, the m-pass application randomizes the CL-certificate (A, B, C, D) (see Section 5.2 on CL-signatures). To this end, it randomly picks an integer l and computes $(R, S, T, W) = ([l]A, [l]B, [l]C, [l]D)$ a new valid CL-certificate on sk_u. Obviously, the m-pass user application can pre-compute several randomized versions of the certificate (A, B, C, D). During the m-pass validation, the m-pass cardlet will pick up one of these randomized CL-certificates.

When the validation at the gate occurs, the validator sends a random challenge rc and a time slot bsn to the m-pass cardlet that will respond with a DAA σ of these challenges. The bsn message is constant during a fixed period of time and enables the implementation of the anti-pass back feature described in Section 6.2. During the validation phase, the m-pass cardlet has very few computations to perform, only the ones that involve the secret key sk_u, the basename bsn and the challenge rc. The m-cardlet first computes $J = \mathcal{H}_1(bsn)$ and a commitment K on sk_u with respect to the base J: $K = [sk_u]J$. It then computes a signature of knowledge (c, s) proving that it knows the discrete logarithm of K with respect to the base J. Afterwards, it communicates the DAA signature σ on rc and bsn to the gate. This DAA σ consists of a randomized CL-certificate (R, S, T, W), pre-computed by the m-pass application, along with J, K, c, and s. Roughly speaking, the DAA signature σ is a signature of knowledge (*cf.* Section 5.2) on rc and bsn, proving that (R, S, T, W) is a CL-certificate on the secret value committed in K and that the m-pass cardlet knows this committed value. Then, the validator verifies that the signature σ is valid and is not blacklisted (*cf.* Section 6.4). If the verification is successful, the validator grants access to the user to the transport network.

6.2. Anti-passback

The anti-passback feature of a transport service denies the access to a user that has already used the m-pass during the previous minutes or seconds depending on how the system is configured. For example, the entrance gate should deny the access if two users try to enter consecutively using the same transport pass.

During the validation phase, the validator stores the signature σ_1 for a time slot bsn. If the user tries to re-use his m-pass during this time slot, the m-pass cardlet will compute and send a new signature σ_2. The validator is able to detect that σ_1 and σ_2 are computed by the same m-pass cardlet by comparing the two parameters K of the signatures σ_1 and σ_2. If they are the same, this means that the two signatures σ_1 and σ_2 originate from the same m-pass cardlet.

When the gate changes the value of bsn, two DAA signatures cannot be linked any more using K. Thus, renewing bsn frequently is mandatory in order to guarantee that the actions of users cannot be linked by the transport authority.

6.3. De-anonymization

The de-anonymization procedure, described in Table 3, allows the transport authority to identify the m-pass (and consequently the holder of this pass) that has generated a particular signature. For instance, in case of a fraud, the logs contained in a gate can be used to

Transport authority		Opening authority
Public input: pk_g, DB_t, σ Private Input: sk_t		Public Input: pk_g Private Input: sk_o, DB_o
	σ ⟶	Find (C_1, C_2, μ) in DB_o such that: $e(J, C_2) = e(K, G_2)$
Knowing C_1 Find (ID_{user}, C_1) in DB_t	⟵ C_1	

Table 3. Protocol for the de-anonymization of a user.

Transport authority		Opening authority
Public input: pk_g, DB_t, ID_{user} Private Input: sk_t		Public Input: pk_g Private Input: sk_o, DB_o
Knowing ID_{user} Find (ID_{user}, C_1) in DB_t	C_1 ⟶	Knowing C_1 Find (C_1, C_2, μ) in DB_o $\forall bsn$, Compute
	⟵ $e(\mathcal{H}_1(bsn), C_2)$	$e(\mathcal{H}_1(bsn), C_2)$

Table 4. Protocol for blacklisting a user.

identify the corresponding malicious users. Of course the transport authority cannot de-anonymize users on his own, which would otherwise harm their privacy. Thus, the revocation of the anonymity of a signature cannot be done without the consent and the cooperation of the opening authority.

First, the transport authority sends the targeted signature $\sigma = (R, S, T, W, J, K, c, s)$ to the opening authority. Based on this signature and the secret database DB_o, the opening authority searches for the triplet (C_1, C_2, μ) such that $e(J, C_2)$ matches with $e(K, G_2)$. If the equality holds, this means that C_2 and K are two commitments on the same secret value (*i.e.*, sk_u in this case). Then, it replies with the corresponding C_1. Finally, based on the database DB_t and the received C_1, the transport authority finds out the identity of the user ID_{user}.

6.4. Blacklist management

The blacklisting procedure, described in Table 4, allows the transport authority to revoke an m-pass such that it cannot access anymore to the transport service. For example, a user that has lost his smartphone can ask the transport authority to blacklist his m-pass, or in case of a fraud the de-anonymized m-pass can be blacklisted by the transport authority.

If the transport authority wants to exclude a user, the authority seeks for the relevant commitment, C_1, of the user in the database DB_t and sends it to the opening authority. Upon receiving C_1, the opening authority retrieves the corresponding commitment C_2 and computes, for all the possible bsn starting from the time the m-pass is blacklisted, $e(\mathcal{H}_1(bsn), C_2)$. The number of computed $e(\mathcal{H}_1(bsn), C_2)$ could appear to be large, but it stays limited as we propose to renew regularly the group public keys to implement the validity duration of all m-passes, as explained in Section 6.5. Thus, depending on the frequency of changes of bsn and of group public keys, the size of the computed set can vary.

Finally, the set of computed $e(\mathcal{H}_1(bsn), C_2)$ is sent back to the transport authority that forwards it to all the validators of the transport system. Thus at the validation, the validator will receive a signature $\sigma = (R, S, T, W, J, K, c, s)$ from an m-pass. It will check the validity of the signature and that the m-pass is not blacklisted. To this end, the validator will check whether $e(\mathcal{H}_1(bsn), C_2)$ matches with $e(K, G_2)$. If the equality holds, this means that C_2 and K are two commitments on the same secret value (*i.e.*, sk_u in our context), and thus the user is blacklisted and the validator denies access to this user. The validator will then typically send a specific message to the m-pass cardlet to lock it. Clearly, this locking message has to be authenticated to thwart the denial-of-service attacks. This kind of countermeasure is very classic and is out of scope of the current paper.

6.5. Groups

A group of users is defined by a group public key pk_g. We suggest to regularly update this key to ensure that this key has only a limited validity period. Indeed, the users have to update their CL-certificate every time the group public key is updated. A user, who did not update his certificate accordingly, will not be considered to be anymore a member of the group and thus he will not be able to generate a valid signature at the validation phase. Moreover, a regular update of the group public key will limit the size of the blacklist at the end of the validation phase.

For the sake of simplicity, we have considered so far a single group of users (*i.e.*, a unique group public key), in order to distinguish a legitimate user from an illegal one. Nevertheless, the ability to define several groups is necessary in order to guarantee the general aspect of the architecture. In practice, a group is characterized by a set of attributes such as for instance, the type of the m-pass, (*e.g.*, student, senior, ...), and the accessible areas (*e.g.*, area1-2, area1-5, area2-3,...). Thus, potentially at a validator, many public group keys can be used.

For example, in the situation in which 1) three groups of users exist: {student, area1-2}, {student, area1-5} and {student, area2-3} and 2) a validator is located in a station of area 3, the users of the groups {student, area1-5} and {student, area2-3} are both legitimate to access the subway at this validator. Thus, to verify the signature of a user, the validator has to browse through

the list of available keys and perform the verification algorithm for each of them.

If the number of groups available in the validator list is important, this approach will be inefficient. Thus, we propose that, at the validation phase, the m-pass cardlet sends to the validator the group it belongs to, which enables to have a constant verification time even if the size of the list of groups is variable.

7. Security analysis

In this section, the security of the proposed protocol is analyzed from two different perspectives. First, the classical security requirements like integrity, non-repudiation and unforgeability are considered. Then in a second time, the requirements regarding user's privacy are taken into account. Finally, we discuss the possible attacks we identified against our protocol and we describe the possible countermeasures.

7.1. Security requirements

The common security requirements of the protocol are the integrity, the non repudiation and the unforgeability. These security requirements are fulfilled by both the classical security properties of the group signature scheme used in the protocol and the tamper-resistance of the secure element processing the sensitive assets.

The attributes of a user, his secret keys and m-pass features such as the geographical zone and the validity date, are embedded into the secure element that, by assumption, ensures their integrity. Thus, it is assumed that it is not possible to modify the sensitive assets stored and processed by the secure element.

The non-repudiation property is directly inherited from the CL certificate and the Schnorr signature. The CL certificate is a 4-tuple (A, B, C, D) in which the three last values are computed from $sk_t = (x, y)$ (*i.e.*, the secret key of the transport authority) and sk_u (*i.e.*, the user's secret key). The Schnorr signature computation involves the user's secret key sk_u while the validation roughly consists in checking that (1) the CL certificate has been provided by the valid transport authority and (2) that it is bound to the same secret key sk_u that has been used in the Schnorr signature computation. Finally, in case of a dispute, the opening authority can disclose the link between a suspicious CL certificate (A, B, C, D) and a user so that this latter cannot repudiate a group signature he computed.

The unforgeability of the m-pass is guaranteed by both the signature verification and the tamper-resistance of the secure element on which the signature is computed. Indeed, it is not possible to compute a valid signature without the knowledge of the secret key sk_u and the CL certificate (A, B, C, D). Thus, the unforgeability property is directly related to the security of the signature scheme which is considered to be secure. Moreover, the secure element protects the confidentiality of the secret key sk_u used in the Schnorr signature computation.

7.2. Privacy requirements

In a nutshell, the anonymity of a user is achieved thanks to the use of a group signature. The user, once he belongs to a group of authorized people, can compute group signatures without revealing any information related to his identity. In particular in our protocol, this is possible due to the structure of the CL certificate that can be randomized each time the user enters the transport network. As a consequence, the "root" CL certificate delivered by the transport authority to the user at the registration time is never directly used during the validation.

The anonymity can be revoked by the opening authority. More precisely, from the signature σ (obtained for example in the log files of the gate), it is possible to recover the corresponding data C_2 of the user by the opening authority. After that, C_1 is recovered and the transport authority has the corresponding identity in his database.

Unlinkability is only achieved partially. Indeed the value $K = [sk_u]H_1(bsn)$, which is involved in the signature computation, is a deterministic function of the bsn value sent by the validator. Unlinkability is therefore only ensured across periodically refreshed bsn values. Consequently, if the validator is malicious and always sends the same value bsn during the validation, the m-pass can be tracked with the value J. This malicious validator could then use the logs from those readers and track the entries and the exits of a user thanks to the occurrence of the value $H_1(bsn)$.

Note that the m-pass cardlet could detect such an attack by a simple comparison between the previously received value (or possibly the list of the previous values of bsn if enough memory is available) and the current one. However, there is still the possible risk of a periodic usage of the same set of values bsn. In this situation, the m-pass cardlet would not be able to detect it if the period is greater than its storage capabilities. Moreover a secure element does not necessarily have access to an internal secure clock. Thus, we cannot expect that the m-pass cardlet can compare the current bsn to the current time. To solve this issue, a solution would consist in reporting the bsn used to an Android application that would be responsible to detect any attempt of user tracking.

7.3. Denial-of-service attacks

When a handset is turned off or when the battery of a handset is drained, the smartcard cardlet has only a limited number of precomputed anonymous

attestations to validate at the gates of the transport network. A malicious user could interact with the victim's pass by simulating a fake gate and then consume the limited amount of those precomputed anonymous credentials available in the smartcard. Once consumed, the victim can no longer enter or exit the transport network. Such an attack has both a low impact on the usability of our proposal and a low benefit for the attacker. Moreover, this attack is difficult to realize, as it requires to target a smartphone whose battery is drained and to succeed to send fake challenges several times.

Another DOS attack would consist in infecting the smartphone of a victim with a malicious Android application that makes the pass always unavailable for a contactless transaction with the validator. This can be achieved by continuously interacting with the pass through the ISO7816 interface that has a higher priority.

Finally, another simple DOS attack would consist in disrupting the Android application responsible of precomputing the anonymous credentials. Such attacks would be of low benefit for an attacker.

7.4. Impersonating a validator

Because of the use of the NFC technology, an attacker can impersonate the validator and challenge the cardlet of a legitimate user. A simple countermeasure consists in implementing an authentication step in the protocol to verify the authenticity of the validator. Nevertheless, such a step would impact the total validation time. Without any authentication phase, an attacker may be able to challenge the mobile pass and discover the authorized zones and validity date. This attack slightly breaches the victim's privacy but such information could already been obtained by following physically the user.

Another interesting attack exploiting the NFC interface is to build a relay attack between the validator and a targeted user. In this scenario, the attacker tries to use the accreditation of a user to unlock the transit gate. To conduct such an attack, the attacker has to build a communication tunnel between the victim's smartphone V and its own smartphone A. Then, when the attacker is challenged by the validator, the received APDUs are forwarded to the smartcard of V and the resulting signature is sent back to A and transmitted to the validator.

The difficulties of such an attack are (1) to transmit efficiently the APDUs and the signature between A and V and (2) to communicate with the smartcard transparently from the smartphone V (using for example a malware). Two countermeasures can defeat such an attack. First, by measuring the elapsed time during a transaction, the smartcard cardlet can detect an abnormal communication speed. Second, by controlling the source of the incoming APDUs (Android, NFC controller), the cardlet can distinguish a malware from an external validator device.

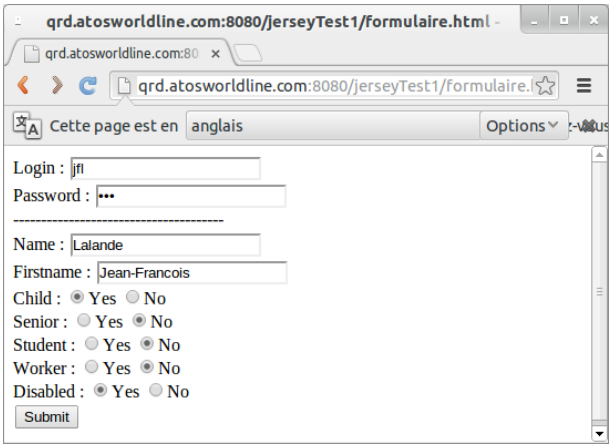

Figure 3. User account creation.

8. Implementation

In this section, we describe the prototype implementing the proposed protocol. We implemented all the steps of a transport service including the user registration, the product registration and the contactless validation.

The software components involved in the transport service are represented in Figure 2. The transport authority is a Tomcat application deployed on a web server. The validator is simulated by a Java swing application connected to an NFC reader using the ISO14443B protocol. The cardlet is embedded in an smartcard that supports javacard applications. We used a regular Galaxy S3 smartphone running Android 4.1.2. The Android system does not need any modification as the Seek for Android patch [36] has been already added by Samsung in order to access secure elements.

8.1. Prototype description

User account creation. First, a user registers himself using the Tomcat application and sets up his attributes. This step is achieved online using a regular browser, as shown in Figure 3. Registering the identity of the user could be achieved by an authority different from the transport operator. In this case, the attributes of the user would be written in an independent cardlet. Nevertheless, in order to simplify developments, we implemented the management of the identity and user attributes in the m-pass cardlet.

Loading pass credentials into the smartcard. The next step consists in loading the attributes (cryptographic credentials, identity and attributes of the user) from the

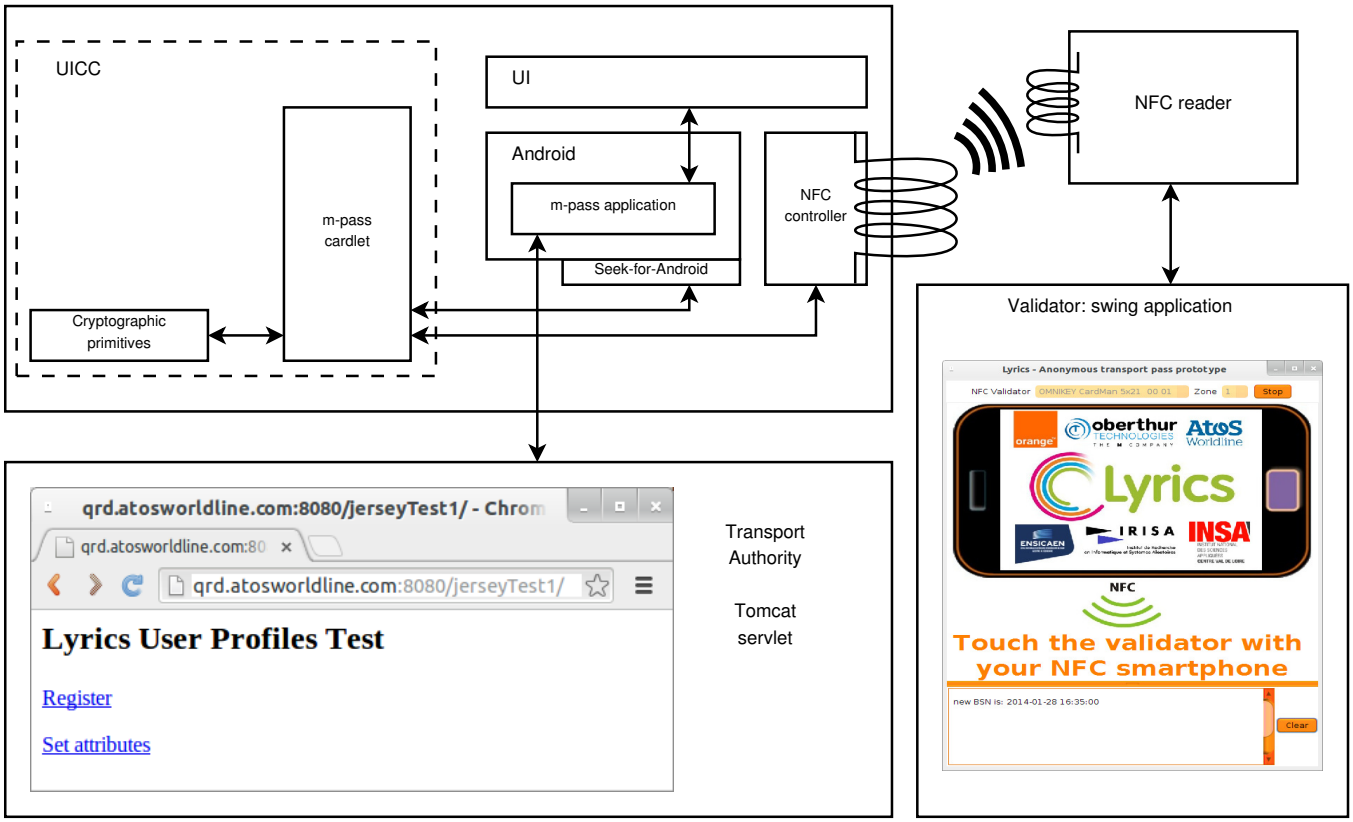

Figure 2. Architecture of the m-pass prototype and validator

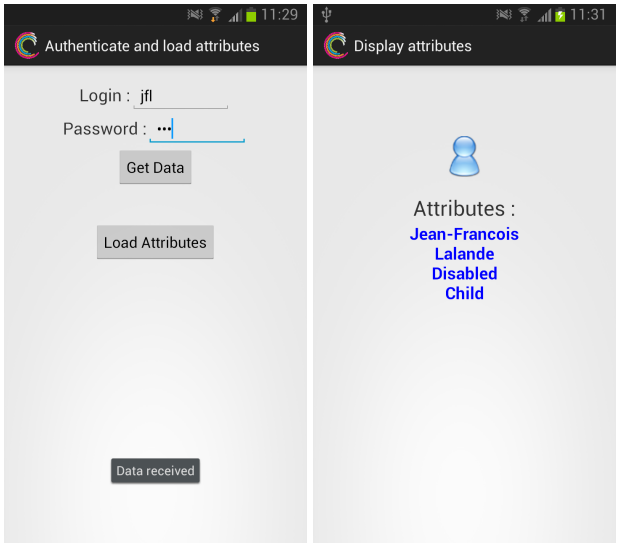

Figure 4. Loading attributes into the cardlet.

server into the smartcard. Using the transport authority Android application, the user can authenticate himself on the server. The application builds an HTTP request with the login/password that are sent to the Tomcat server in order to receive the attributes and write

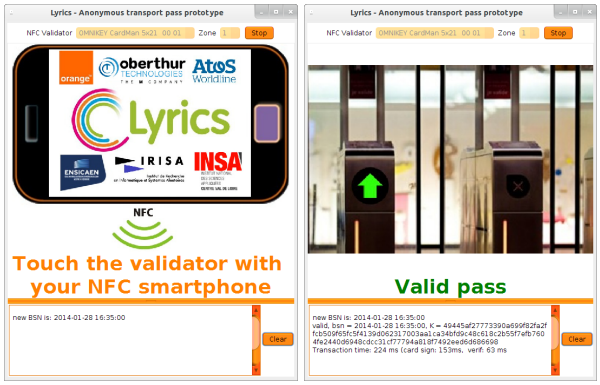

Figure 5. Validation result

them to the smartcard (first screen of Figure 4). These attributes can be read from the smartcard upon user request (second screen of Figure 4). Finally, the m-pass can be bought and stored in the smartcard. In the current prototype, there is no exchange with the server and the online payment is simulated. The chosen m-pass is written to the smartcard: it contains the type of m-pass, the geographical zones of validity and the validity duration.

```
1  GroupMemberCandidate memCand = new GroupMemberCandidate(params);
2  LRSW1JoinMess1ToOpeningAuth mess1 = memCand.joinStep1(rand);
3  // mess1 must be send to opening authority
4
5  LRSW1JoinMess2OASign mess2 = OA.joinStep2(mess1);
6  // mess2, containing mu, is sent back by the opening authority
7
8  LRSW1JoinMess3ToGM mess3 = memCand.joinStep3(mess2);
9  // mess3, containing mu and C1, is sent to the transport authority
10
11 // Getting A, B, C, D from the transport authority
12 memCert = gM.JoinStep4(mess3);
```

Listing 1: Registration protocol implementation.

Validation. The validation step occurs when the user taps the validator, represented by the Swing application connected to the NFC reader on the left part of Figure 5, with his smartphone. After detecting the smartphone and selecting the m-pass cardlet, the Swing application sends the basename *bsn* and the challenge *rc* to the m-pass cardlet. The signature, computed by the cardlet, is sent back to the validator that checks it and displays the result, as shown on the right part of Figure 5.

8.2. Protocol implementation details

Registration with authorities. In order to simplify the development of our prototype, the two authorities (opening and transport) have not been deployed on a real server. The registration phase described in Table 1 is simulated by a unique Java program that builds the exchanged messages in order to represent the communications. The four exchanges of the registration protocol are represented by four messages in Listing 1. These messages involve the three entities of Table 1: the m-pass cardlet, the opening authority and the transport authority.

First, the m-pass carldet generates G_1 and G_2, represented by *memCand*, and builds C_1 and C_2, stored in the message *mess*1. The call to *joinStep*1 executes the multiplication C1 = params.bigG.multiply(sku);. Such calls are directly accelerated by the smartcard using native libraries, as described later in Section 8.3. Then, the opening authority sends back μ, represented by message *mess*2.

In a second phase, at Line 8 of Listing 1, the signature μ and C_1 are sent to the transport authority across the *mess*3 message. The transport authority sends back A, B, C and D that are stored in the *memCert* object.

Precomputation operations. Before the validation step, some precomputations operations must be achieved by the smartphone after receiving A, B, C and D from the cardlet. The smartphone application then generates a pool of precomputed tokens by multiplying each variable A, B, C and D by a random l, as shown in the upper part of Listing 2. These multiplications are performed on the smartphone side by calling the multiply function, which is in this case a pure Java code. The pool of generated tokens is sent back to the cardlet using an adapted compression algorithm.

The bottom part of Listing 2 shows the decoding phase of the message contained in *data* in which

```
1  public LRSW1Token generateToken(Random rand) {
2    BigInteger l = new BigInteger(params.r.bitLength(), rand).mod(params.r);
3    return new LRSW1Token(certif.bigA.multiply(l), certif.bigB.multiply(l), certif.
         bigC.multiply(l), certif.bigD.multiply(l)); }
```

```
1  public byte setToken(byte[] data, short offset) {
2          short encSize = (short) (GroupParameters.p.length + (short) 1);
3          if(((short) (data.length – offset)) < (short)((encSize + (short)1) * (
               short) 4)) return (short) 4;
4
5          short ind= offset;
6          ECUtils.decodeCompressedPointY(data, ind, encSize, bigR_x, bigR_y);
7          bigR_enc = data[ind]; // Retrouver le signe lors de la decompression
8          ind += encSize;
9          ECUtils.decodeCompressedPointY(data, ind, encSize, bigS_x, bigS_y);
10         bigS_enc = data[ind];
11         // Idem for T and W
12         ...
13
14         // Generate random k in Zp
15         rand.generateData(k, (short) 0, (short) k.length);
16
17         // R2 = k.S
18         GroupParameters.ec_curve.multiply(k, (short) 0, (short) k.length,
               bigS_x, (short) 0, bigS_y, (short) 0, sizeP, bigR2_x, (short)
               0, bigR2_y, (short) 0); }
```

Listing 2. Precomputations (smartphone/cardlet).

```
1  public byte sign(byte [] bsn, byte[] rc) {
2    short indToken = searchUnusedToken();
3
4    if(indToken < 0 ) return (byte) 1;
5
6    // J = Hash(bsn)
7    ECUtils.hashToRTorsionPoint(bsn, bigJ_x, bigJ_y);
8
9    // R1 = k.J
10   byte[] k = tokens[indToken].k;
11   GroupParameters.ec_curve.multiply(k, (short) 0, (short) k.length, bigJ_x, (
         short) 0, bigJ_y, (short) 0, (short) bigJ_x.length, bigR1_x, (short)
         0, bigR1_y, (short) 0);
12
13   // K = sku.J
14   byte sku [] = MemberPrivateKey.sku;
15   GroupParameters.ec_curve.multiply(sku, (short) 0, (short) sku.length, bigJ_x,
         (short) 0, bigJ_y, (short) 0, (short) bigJ_x.length, bigK_x, (short)
         0, bigK_y, (short) 0);
16
17   // c = H(J, K, R, S, T, W, R1, R2, m) mod r
18   digestUpdate(bigJ_x, bigJ_y);
19   digestUpdate(bigK_x, bigK_y);
20   digestUpdate(tokens[indToken].bigR_x, tokens[indToken].bigR_y);
21   digestUpdate(tokens[indToken].bigS_x, tokens[indToken].bigS_y);
22   digestUpdate(tokens[indToken].bigT_x, tokens[indToken].bigT_y);
23   digestUpdate(tokens[indToken].bigW_x, tokens[indToken].bigW_y);
24   digestUpdate(bigR1_x, bigR1_y);
25   digestUpdate(tokens[indToken].bigR2_x, tokens[indToken].bigR2_y);
26   digest.doFinal(rc, (short) 0, (short) rc.length, c, (short) 0);
27   // mod r
28   ECUtils.mod(c, GroupParameters.r, c);
29
30   // s = k + c.sku mod r
31   byte r[] = GroupParameters.r;
32   modular.multiply(c, sku, r, (short) r.length, tmp);
33   modular.add(k, tmp, r, (short) r.length, s);
34   ... }
```

Listing 3. Signature computation (cardlet).

each *token*, indexed by $offset$, is extracted and uncompressed. For example, $BigR\_x$ and $BigR\_Y$ are computed from *data*, which corresponds to the received point R in the protocol. At the end, for the considered token, $R2$ is computed.

Validation of the m-ticket. Listing 3 shows the final computation of the signature μ using the challenge parameters transmitted by the validator *bsn* and *rc*. When computing the signature, the code uses one of the precomputed tokens by accessing $tokens[indToken]$ and the associated random value of k.

8.3. Smartcard acceleration

The smartcard that has been used in this prototype is a Java Card compliant with:

- Global Platform version 2.2.1,
- Javacard version 2.2.2 and
- Javacard virtual Machine version 3.0.1 classic.

Those features allow to develop so-called cardlets, to load them on the Java Card and to execute them thanks to the Java Card virtual machine. This execution environment is typical of a modern UICC/SIM card. In order to provide a very efficient signature computation, we chose a high performance UICC equipped with an arithmetic coprocessor.

Thanks to the arithmetic coprocessor, we were able to meet the timing performances required by the transport operators (typically less than 300 ms), by offering non standardized Java Card APIs to the signature cardlet (see Listing 3). Those APIs allow to compute efficiently:

- modular operations on large integers like addition, subtraction, multiplication, exponentiation, etc., but also
- arithmetic operations on elliptic curves like scalar multiplication and point addition.

Those customized APIs are mapped to assembly subroutines that drive the arithmetic coprocessor directly.

For instance, the cardlet in Listing 3 performs the following function calls that are supported by the coprocessor:

- `modular.multiply` performs a modular multiplication,
- `modular.add` performs a modular addition,
- `ec_curve.multiply` performs a scalar multiplication,

in which the size of the involved operands is 256 bits.

Protocol step	Measured time
Total transaction time	186 ms
Cardlet selection	12 ms
Cardlet signature	140 ms
Verification	34 ms

Table 5. Execution times of our prototype.

8.4. Performances

In this section, we first present the performance results of the accelerated customized APIs implemented in the smartcard. Then, we give the performance of our protocol by measuring the total transaction time when the smartphone is on or off.

Smartcard performances. We stress the need for an hardware acceleration to achieve the required performances. Indeed, we have benchmarked two implementations for the modular multiplication and the modular exponentiation (this latter is not used in the protocol). To multiply two 1024 bit operands, a pure software implementation takes around 10 ms whereas it runs in less than a 1 ms when supported by the arithmetic coprocessor. It is even more eloquent when talking about the modular exponentiation of a 1024 bit base (same bit size as the module) by a 1024 bit exponent that takes around 1300 ms with a pure software implementation and falls under 100 ms when the API is supported by the arithmetic coprocessor.

Smartphone switched on. The performance of our solution is critical for the validation of the m-ticket. The whole validation process takes 186 ms, which is quite efficient. It enables to add the authentication of the validator into a window of time of 300 ms. Table 5 gives the different execution time of the protocol measured by the terminal control GUI:

- Transaction time (total time): the time from the detection of the smartcard by the validator until the validator announces the result of the signature verification.
- Cardlet selection: the time needed, after detecting the smartcard by the validator, for selecting the cardlet into the smartcard.
- Smartcard cardlet signature: the time to get a signature from the cardlet (including NFC communications).
- Verification: the time to verify the signature into the validator.

Smartphone switched off / drained battery. It is possible to use the service even if the smartphone has run out of battery. Indeed, the ticketing application hosted in

the Android is involved only in the pre-computation phase. Once this step is done, the validator interacts with the smartcard via the NFC controller and sends the energy via the NFC magnetic field. In this case, the smartcard runs the signature with lower performances. We measured a total transaction time of 484 ms.

9. Conclusion

In this paper, we propose a privacy-preserving solution for e-ticketing in transport systems that can be embedded into an NFC smartphone. This solution enables users to use the transport service in an anonymous and unlinkable manner, preventing in particular the risk of being tracked by the service provider. However, in exceptional circumstances (*e.g.*, under the injunction of a judge), the identity of the user associated to an anonymous validation of the mobile pass can be de-anonymized and his rights to access the transport network revoked. Thus, this solution is flexible enough to ensure the standard security properties while providing a high level of privacy to users.

The cryptographic protocol behind our solution is based on group signatures that are used when authenticating with the m-pass at one of the entrance gates of the transport network. When a user authenticates to the validator, he will sign on a challenge on behalf of a group, thus hiding his identity from the transport operator. The unlinkability of the different actions of the user is obtained by changing regularly the challenge sent by the validator. In the situation in which a malicious service operator deliberately refuses to renew the challenge, we suggest to detect such an attack by collecting the received challenge using an Android application for implementing the detection.

The prototype that has been developed in order to evaluate the performance of our protocol shows that the total transaction time is of 186 ms on a Samsung Galaxy S3 equipped with a Javacard smartcard. To obtain such an efficient validation time, the protocol has been split between the smartphone and the smartcard, which enables the precomputation of a part of the group signature in the smartphone before the validation step. The information generated during the precomputation step is not critical in terms of security, in particular from the point of view of an adversary abusing the operating system. Indeed, the private signature key of the user for the group remains protected in the smartcard. The current implementation has the additional benefit that even if the battery is out of energy, the service can still be used several times by the user.

In the future, we would like to broaden the applicability of the proposed approach to other situations linked to the mobile identity of a user. For instance, we plan to design a generic authentication protocol for NFC devices that would allow or refuse an access based on a combination of the certified attributes of the user as stored on his smartphone. One of the difficulties of this use case is that the user's privacy could be threatened by a leak of one of the attributes of the user when allowing or denying an access. We also envision the possibility of computing in a distributed and secure manner a function that takes as input the attributes of several users and outputs a global property of the population of the users considered (statistics, survey, ...).

Acknowledgement. This work has been supported by French National Research Agency (ANR) through LYRICS program (ANR-11-INS-0013 LYRICS project).

References

[1] Ghiron, S.L., Sposato, S., Medaglia, C.M. and Moroni, A. (2009) NFC ticketing: A prototype and usability test of an NFC-based virtual ticketing application. In *2009 First International Workshop on Near Field Communication* (Hagenberg, Austria: IEEE Computer Society): 45–50. doi:10.1109/NFC.2009.22.

[2] Madlmayr, G. and Kleebauer, P. (2008) Secure communication between web browsers and NFC targets by the example of an e-ticketing system. In Psaila, G. and Wagner, R. [eds.] *9th International Conference on E-Commerce and Web Technologie* (Turin, Italy: Springer Berlin / Heidelberg): 1–10. doi:10.1007/978-3-540-85717-4_1.

[3] Smart Card Alliance (2007), Proximity mobile payments: Leveraging NFC and the contactless financial payments infrastructure. URL http://www.smartcardalliance.org/resources/lib/Proximity_Mobile_Payments_200709.pdf.

[4] Dmitrienko, A., Sadeghi, A.R., Tamrakar, S. and Wachsmann, C. (2012) Smarttokens: Delegable access control with NFC-enabled smartphones. In Katzenbeisser, S., Weippl, E., Camp, L., Volkamer, M., Reiter, M. and Zhang, X. [eds.] *Trust and Trustworthy Computing* (Vienna, Austria: Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **7344**, 219–238. doi:10.1007/978-3-642-30921-2_13.

[5] Cavoukian, A. (2011) *Mobile Near Field Communications (NFC) "Tap 'n Go" Keep it Secure & Private*. Tech. rep., Information and Privacy Commissioner, Ontario, Canada. URL http://www.ipc.on.ca/images/Resources/mobile-nfc.pdf.

[6] European Union (1995), Directive 95/46/EC of the European Parliament and of the Council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31995L0046:EN:HTML.

[7] Business Wire (2013), OT's PEARL embedded secure element is certified by american express. URL http://www.businesswire.com/news/home/20131204005514/en.

[8] Business Wire (2013), Oberthur technologies' dragonfly nfc sim card certified by mastercard and visa. URL http://www.businesswire.com/news/home/20130522005727/en.

[9] Mut-Puigserver, M., Payeras-Capellà, M.M., Ferrer-Gomila, J.L., Vives-Guasch, A. and Castellà-Roca, J. (2012) A survey of electronic ticketing applied to transport. *Computers & Security* **31**(8): 925–939. doi:10.1016/j.cose.2012.07.004.

[10] Calypso Networks Association (2010), Calypso handbook v1.1. URL http://www.calypsostandard.net/index.php/documents/specifications/public-documents/79-100324-calypso-handbook.

[11] Tamrakar, S., Ekberg, J.E. and Asokan, N. (2011) Identity verification schemes for public transport ticketing with nfc phones. In *The sixth ACM workshop on Scalable trusted computing*, STC '11 (Chicago, IL, USA: ACM): 37–48. doi:10.1145/2046582.2046591.

[12] Fujimura, K. and Nakajima, Y. (1998) General-purpose digital ticket framework. In *3rd conference on USENIX Workshop on Electronic Commerce* (Boston, Massachusetts, USA: USENIX Association): 15–24. URL https://www.usenix.org/legacy/events/ec98/full_papers/fujimura/fujimura.pdf.

[13] Baldimtsi, F. and Lysyanskaya, A. (2013) Anonymous credentials light. In Sadeghi, A.R., Gligor, V.D. and Yung, M. [eds.] *ACM Conference on Computer and Communications Security* (Berlin, Germany: ACM): 1087–1098. doi:10.1145/2508859.2516687.

[14] Vives-Guasch, A., Payeras-Capellà, M., Puigserver, M.M., Castellà-Roca, J. and Ferrer-Gomila, J.L. (2012) A secure e-ticketing scheme for mobile devices with near field communication (NFC) that includes exculpability and reusability. *IEICE Transactions* **95-D**(1): 78–93. URL http://search.ieice.org/bin/summary.php?id=e95-d_1_78&category=D&year=2012&lang=E&abst=.

[15] Calypso Networks Association (2010), Functional card application v1.4. URL http://www.calypsostandard.net/index.php/documents/specifications/public-documents/78-010608-functional-card-application.

[16] The project Touch and Travel (2008). URL http://www.touchandtravel.de/.

[17] Pirker, M. and Slamanig, D. (2012) A framework for privacy-preserving mobile payment on security enhanced arm trustzone platforms. In *11th International Conference on Trust, Security and Privacy in Computing and Communications* (Liverpool, United Kingdom: IEEE Computer Society): 1155–1160. doi:10.1109/TrustCom.2012.28.

[18] Business Wire (2013), Oberthur technologies' contactless payment solution selected by movistar chile and banco santander chile. URL http://www.businesswire.com/news/home/20130901005026/en.

[19] Sony Corp. Info (2013), Mobile payment services using nfc sims equipped with sony felica™ technology to begin in hong kong. URL http://www.sony.net/SonyInfo/News/Press/201310/13-137E/.

[20] Gematlo (2013), Gemalto enables commercial mobile nfc transport and payment roll-out in hong kong. URL http://www.gemalto.com/php/pr_view.php?id=1685.

[21] Lee, A., Lui, T. and Leung, B. (visited the 31th January 2013), Security analysis of the octopus system. URL http://www.proxmark.org/files/Documents/13.56%20MHz%20-%20Felica/security_analysis_of_octopus_smart_card_system.pdf.

[22] Tamrakar, S. and Ekberg, J.E. (2013) Tapping and tripping with nfc. In Huth, M., Asokan, N., Čapkun, S., Flechais, I. and Coles-Kemp, L. [eds.] *Trust and Trustworthy Computing* (London, United Kingdom: Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **7904**, 115–132. doi:10.1007/978-3-642-38908-5_9.

[23] Ekberg, J.E. and Tamrakar, S. (2012) Mass transit ticketing with NFC mobile phones. In Chen, L., Yung, M. and Zhu, L. [eds.] *Trusted Systems* (Beijing, China: Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **7222**, 48–65. doi:10.1007/978-3-642-32298-3_4.

[24] Derler, D., Potzmader, K., Winter, J. and Dietrich, K. (2011) Anonymous ticketing for NFC-enabled mobile phones. In Chen, L., Yung, M. and Zhu, L. [eds.] *The Third International Conference on Trusted Systems* (Beijing, China: Springer Berlin Heidelberg), **7222**: 66–83. doi:10.1007/978-3-642-32298-3_5.

[25] Brands, S. (2000) *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy* (Cambridge: MIT Press).

[26] Glenn, A., Goldberg, I., Légaré, F., Stiglic, A. (2001), A description of protocols for private credentials. URL http://eprint.iacr.org/2001/082.

[27] Isern-Deya, A.P., Vives-Guasch, A., Mut-Puigserver, M., Payeras-Capella, M. and Castella-Roca, J. (2012) A secure automatic fare collection system for time-based or distance-based services with revocable anonymity for users. *The Computer Journal* **56**(10): 1198–1215. doi:10.1093/comjnl/bxs033.

[28] Boneh, D. and Boyen, X. (2004) Short signatures without random oracles. In Cachin, C. and Camenisch, J. [eds.] *Advances in Cryptology - EUROCRYPT 2004* (Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **3027**, 56–73. doi:10.1007/978-3-540-24676-3_4.

[29] Boneh, D., Boyen, X. and Shacham, H. (2004) Short group signatures. In Franklin, M. [ed.] *Advances in Cryptology - CRYPTO 2004* (Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **3152**, 41–55. doi:10.1007/978-3-540-28628-8_3.

[30] Fiat, A. and Shamir, A. (1986) How to prove yourself: Practical solutions to identification and signature problems. In Odlyzko, A. [ed.] *Advances in Cryptology, CRYPTO'86* (Santa Barbara, CA, USA: Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **263**, 186–194. doi:10.1007/3-540-47721-7_12.

[31] Schnorr, C. (1989) Efficient identification and signatures for smart cards. In Quisquater, J.J. and Vandewalle, J. [eds.] *Advances in Cryptology - EUROCRYPT'89* (Houthalen, Belgium: Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **434**, 688–689. doi:10.1007/3-540-46885-4_68.

[32] Chaum, D. and Van Heyst, E. (1991) Group signatures. In *Proceedings of the 10th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'91 (Berlin, Heidelberg: Springer-Verlag):

257–265. URL http://dl.acm.org/citation.cfm?id=1754868.1754897.

[33] Canard, S., Schoenmakers, B., Stam, M. and Traoré, J. (2006) List signature schemes. *Discrete Applied Mathematics* **154**(2): 189–201. doi:10.1016/j.dam.2005.08.003.

[34] Brickell, E., Camenisch, J. and Chen, L. (2004) Direct anonymous attestation. In *Proceedings of the 11th ACM Conference on Computer and Communications Security*, CCS '04 (New York, NY, USA: ACM): 132–145. doi:10.1145/1030083.1030103.

[35] Camenisch, J. and Lysyanskaya, A. (2004) Signature schemes and anonymous credentials from bilinear maps. In Franklin, M. [ed.] *Advances in Cryptology - CRYPTO 2004* (Santa Barbara, CA, USA: Springer Berlin Heidelberg), *Lecture Notes in Computer Science* **3152**, 56–72. doi:10.1007/978-3-540-28628-8_4.

[36] Seek for Android (2013). http://code.google.com/p/seek-for-android/.