CONTRIBUTIONS TO THE SECURITY AND PRIVACY OF ELECTRONIC TICKETING SYSTEMS

Arnau Vives Guasch

Dipòsit Legal: T.1028-2013

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO THE SECURITY AND PRIVACY OF ELECTRONIC TICKETING SYSTEMS
Arnau Vives Guasch
Dipòsit Legal: T.1028-2013

Universitat Rovira i Virgili

Department of Computer Engineering and Mathematics

Ph.D. Dissertation

Contributions to the Security and Privacy of Electronic Ticketing Systems

Author:

Arnau Vives-Guasch

Thesis Advisor:

Dr. Jordi Castellà-Roca

Dissertation submitted to the Department of Computer Engineering and Mathematics in partial fulfillment of the requirements of the degree of Doctor of Philosophy in Computer Science

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO THE SECURITY AND PRIVACY OF ELECTRONIC TICKETING SYSTEMS
Arnau Vives Guasch
Dipòsit Legal: T.1028-2013

Arnau Vives-Guasch

Contributions to the Security and Privacy of Electronic Ticketing Systems

Ph.D. Dissertation

Directed by Dr. Jordi Castellà-Roca

Department of Computer Engineering and Mathematics

Universitat Rovira i Virgili

Tarragona

2013

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO THE SECURITY AND PRIVACY OF ELECTRONIC TICKETING SYSTEMS
Arnau Vives Guasch
Dipòsit Legal: T.1028-2013

UNIVERSITAT ROVIRA I VIRGILI
CONTRIBUTIONS TO THE SECURITY AND PRIVACY OF ELECTRONIC TICKETING SYSTEMS
Arnau Vives Guasch
Dipòsit Legal: T.1028-2013

UNIVERSITAT ROVIRA I VIRGILI

DEPARTMENT OF COMPUTER ENGINEERING AND MATHEMATICS

I STATE that the present study, entitled "Contributions to the Security and Privacy of Electronic Ticketing Systems", presented by Arnau Vives-Guasch for the award of the degree of Doctor, has been carried out under my supervision at the Department of Computer Engineering and Mathematics of this university, and that it fulfils all the requirements to be eligible for the European Doctorate Award.

Tarragona, May 15, 2013

Dr. Jordi Castellà-Roca, Doctoral Thesis Supervisor

Approved by the University Committee on Graduate Studies:

Acknowledgements

Many thanks to my director Jordi for his great patience and support. I also want to thank Macià Mut, Magdalena Payeras, Andreu Pere Isern and Pep Lluís Ferrer from the Universitat de les Illes Balears for their help and collaboration. I am also very grateful to my father, my mother, Joan, Úrsula, and the numerous people who have helped me during the elaboration of this thesis. Finally I want to thank Xavier Pérez Costa and the whole company NEC Laboratories Europe, Ltd. in Heidelberg (Germany) for their hospitality and cooperation during my stay there during the spring of 2012.

Resum

Un bitllet electrònic és un contracte en format digital entre dues parts, l'usuari i el proveïdor de serveis, on hi queda reflectit l'acord entre ambdós per tal que l'usuari rebi el servei que desitja per part del proveïdor. Els bitllets són emprats en diferents tipus de serveis, com esdeveniments lúdics o esportius, i especialment en l'àmbit del transport. En aquest cas permet reduir costos donat l'alt volum d'usuaris, a més de facilitar la identificació del flux de viatges. Aquesta informació permet preveure i planificar els sistemes de transport de forma més dinàmica.

La seguretat dels bitllets electrònics és clau perquè es despleguin a l'entorn real, com també ho és la privadesa dels seus usuaris. La privadesa inclou tant l'anonimitat dels usuaris, és a dir, una acció no s'ha de poder atribuir fàcilment a un determinat usuari, com també la no enllaçabilitat dels diferents moviments d'un determinat usuari.

En aquesta tesi proposem protocols de bitllets electrònics que mantinguin les propietats dels bitllets en paper juntament amb els avantatges dels bitllets digitals. Primerament fem un estat de l'art amb les propostes relacionades, analitzant-ne els requisits de seguretat que compleixen. Presentem un protocol de bitllets electrònics que incorpora els nous requisits de seguretat d'exculpabilitat i reutilització, diferents dels que haviem analitzat, tot complint també la privadesa pels usuaris. Posteriorment, presentem una proposta de bitllets electrònics adaptada als sistemes de pagament depenent de l'ús, bàsicament enfocat al transport, que incorpora tant l'anonimat pels usuaris, com també la enllaçabilitat a curt termini, és a dir, complint la no enllaçabilitat dels diferents moviments del mateix usuari, però permetent la enllaçabilitat de les accions relacionades amb el mateix trajecte (p.ex. entrada i sortida). Finalment, mitjançant una evolució de la mateixa tècnica criptogràfica utilitzada en el sistema de pagament per ús, millorant-ne el temps de verificació per a múltiples bitllets alhora (verificació en "batch"), presentem una proposta que pot ser útil per a varis sistemes de verificació massiva de missatges, posant com a cas d'ús l'aplicació a sistemes de xarxes vehiculars.

Abstract

An electronic ticket is a digital contract between two parties, that is, the user and the service provider. An agreement between them is established in order that the user can receive the desired service. These tickets are used in different types of services, such as sports or entertainment events, especially in the field of transport. In the case of transport, costs can be reduced due to the high volume of users, and the identification of the travel flow is facilitated. This information allows the forecast and planification of transport systems more dynamically.

The security of electronic tickets is very important to be deployed in the real scenarios, as well as the privacy for their users. Privacy includes both the anonymity of users, which implies that an action cannot be easily attributed to a particular user, and also the unlinkability of the different movements of that user.

This thesis presents protocols which keep the same security requirements of paper tickets while offering the advantages of digital tickets. Firstly, we perform a state of the art with the related proposals, by analysing the security requirements considered. We then present an electronic ticketing system that includes the security requirements of exculpability and reusability, thus guaranteeing the privacy for users. We later present a proposal of electronic ticketing systems adapted to use-dependant payment systems, especially focused on transport, which includes both the anonymity of users and the short-term linkability of their movements. The related actions of a journey of a determined user can be linkable between them (i.e. entrance and exit of the system) but not with other movements that the user performs. Finally, as an extension of the previous use-dependant payment system solution, we introduce the case of mass-verification systems, where many messages have to be verified in short time, and we present a proposal as a vehicular network use case that guarantees privacy for users with short-term linkability and can verify these messages efficiently.

Contents

CHAPTER 1

Introduction

This chapter introduces the main issues we face in this dissertation. Moreover, it briefly describes the solutions that have been adopted. Finally, the structure and organisation of this thesis are defined.

Contents

We start with the motivation in §1.1, followed by the main contributions performed in §1.2, and the organisation of this PhD dissertation in §1.3.

1.1 Motivation

The use of Information and Communication Technologies (ICT) is increasing every day in our common operations, since ICT are replacing classic paper systems by digital ones, i.e. the case of electronic tickets.

An electronic ticket is a contract in digital format between two parties, the user and the service provider, which reflects the agreement between them so that the user receives the desired service from the provider. This ticket is used in different types of transportation systems, entertainment and special events, etc. In any of these cases, this electronic ticket contains information about the terms and conditions of the associated service, and their use limitations, such as an established number of uses or a validity time.

In order to complete the transition from paper to digital format, the same security requirements provided by the paper tickets have to be guaranteed in the new digital scenario. Digital information is known to be easily copied or modified, which enables the attempts of forgery or the duplication of information, allowing

fraudsters to exploit them, for example, in order to use the service more times than the preset in the ticket.

Actually, some cryptographic techniques such as the electronic signature are used in order to ensure that the issuer of a ticket is the authorised one (authenticity), that the ticket has not been modified since its issue (integrity), and that any entity that has generated a ticket cannot deny it (non-repudiation).

However, security is not the only requirement of those systems, because users also require privacy for their transactions. Nevertheless, the design of secure and efficient protocols that provide privacy to users is a difficult challenge that also depends on the devices that the users handle.

The mobile industry is taking profit of the rise of the newest smartphones, as they give usability and portability to the user. These devices offer both high computation and storage capabilities, together with a wide variety of communication technologies (Wi-Fi, Bluetooth, NFC, etc.). These devices thus become valuable tools that could be key in the future due to their multiple applications. Moreover, the high acceptance rates of these devices by the user community are making this mobile industry grow at an outstanding rate.

In this line, transportation seen as ticketing systems over mobile devices is one of the main applications that could be deployed at the present or in the very close future, so the interest on these systems requires the design of cryptographic protocols to protect their transactions and also their users. They also need to be efficient for their deployment to mobile devices. We classify these systems in which we have contributed as follows:

- *Electronic ticketing systems.* Such systems allow a *user* to get an electronic ticket from the authorised *issuer*, and a *service provider* can verify it in order to provide the according service. Note that, in this scenario, the user has already paid for the service, so she receives the ticket as a grant to further receive that service. We can consider them as prepaid ticketing systems.

- *Automatic Fare (or Electronic Toll) Collection systems.* Such systems allow a *user* to use a payment transportation system in which the fare to be paid depends on its use (i.e. tolls, subway, etc.). In such scenario, users receive an entrance ticket. The roles involved in the system are *entrance station* and *exit station*, which they are assigned in order to receive the according tickets, and finally a *payment manager* as an entity to manage the payment calculated at the exit.

In this scenario, the user has to pay at the system exit, so we can see them as postpayment ticketing systems.

- *Mobile and Vehicular Ad-Hoc Networks (MANETs and VANETs).* These networks are formed by mobile nodes which are connected in a non-hierarchical way. Usually, these devices are connected with wireless communications forming this ad-hoc network. VANETs are an specialization of MANETs which are specially oriented to transport with vehicles, by using different types of devices for communication. They use On-Board Units (OBUs), small computation units inside vehicles which have wireless communication, and Road-Side Units (RSUs), entities of the system that can feed information from trusted sources. These kind of communications are useful for special purposes, as they could enable the exchange of real-time data, from accident alarms, traffic jams to any other information related to the journey. Mass-verification systems is still an open problem that involves all types of ticketing and transport systems.

Privacy is an important requirement in these use cases, since having new communication and computation capabilities does not necessarily mean that users can be identified –anonymity–. Likewise, their journeys or habits can be tracked in any case –unlinkability– in order to avoid profile generation.

In this thesis we propose security and privacy solutions by means of cryptography, which could be applied to multiple use cases, such as electronic ticketing systems, use-dependant payment systems, such as Automatic Fare Collection or Electronic Toll systems, and finally inter-device communication such as Mobile or Vehicular Ad-Hoc Networks (MANETs and VANETs). Moreover, these protocols are designed taking into account the latest trends, such as the use of mobile devices for users, in order to allow an easy deployment for possible real scenarios.

1.2 Contributions

The main contributions of this dissertation are the following:

1. **State of the Art in Electronic Ticketing Systems.** In this thesis we present an analysis of the existing proposals related to the field. Moreover, these proposals are classified depending on the degree of anonymity for the users of those systems.

2. **Secure e-Ticketing system for mobile devices, which includes exculpability as a security requirement.** Designing secure and efficient e-ticketing protocols that preserve anonymity of users is a challenge, and more if mobile devices are used. In those conditions, all the movements of a same user must not be tracked or recognised. Under these assumptions, we propose a scheme for mobile devices that protects anonymity for users and introduces exculpability as a novel security requirement. We finally show its efficiency with experimental results.

3. **Secure Automatic Fare Collection system for different transport services.** Automatic Fare Collection systems (also known as Electronic Toll Collection) calculate the fare to be paid according to the use of the system. We present a solution that protects security and also privacy of their users, regarding the unlinkability of the different journeys of a determined user, but also enabling to link the entrance and the exit of a same journey.

4. **Short-term linkable group signatures with categorized batch verification.** Regarding the previous works, we propose a system with short-term linkability between group signatures, performed by the same user, which improves the results specially in the verification phase. This proposal is based on the use case of Vehicle Ad-hoc Networks, but they can also be applied as a generic solution for mass-transport systems such as electronic ticketing, toll collection, etc.

1.3 Organisation

This thesis is organised as follows:

- Chapter 2 provides an overview of e-ticketing systems, and describes the major trends concerning the involved participants, the information included into the ticket, and the main services which can be used. Some definitions are presented to describe the main requirements, differring between security or functional requirements. Finally, the main security proposals in the field are described and categorized depending on the level of anonymity.

- Chapter 3 introduces cryptographic background on which our proposals are

based. We describe the notation to be further used, and the main assumptions and definitions.

- Chapter 4 presents our first contribution to e-ticketing systems thought for mobile devices for users. In more detail, it focuses on the privacy to users, by ensuring revocable anonymity, and we introduce the requirement that both parties (user and service provider) can verify whether the processes have been performed successfully or not.

- Chapter 5 introduces a contribution to Automatic Fare (or Electronic Toll) Collection (AFC) systems, in which the fare, either time- or distance-dependant, is calculated depending on the use of a transportation system. The electronic tickets exchanged between user and provider have to link one entrance ticket of a certain user to its corresponding exit ticket, in order to solve confabulation fraud attacks. However, other electronic tickets different from that movement cannot be linkable with other ones of the same user, what could enable generation of profiles. We call short-term linkability to this requirement, which is achieved by using our novel adaptation to the group signatures scheme that we present.

- Chapter 6 presents an approach based on the batch verification of group signatures, which allows to verify a set of group signatures in just one operation, as an extension of the work made in the previous chapter, so we preserve security and privacy, namely anonymity for users and short-term linkability of movements. The solution that we present suits especially in systems where a significant amount of messages/tickets are sent in a high frequency (mass-verification systems).

- Finally, Chapter 7 summarises our contributions and describes possible future research lines.

CHAPTER 2

State of the Art

This chapter presents the existing proposals related to electronic ticketing systems.

Contents

2.1 Electronic ticketing systems

This section includes the analysis of the e-ticketing systems, first defining the involved participants, the related phases, the most suitable services related to public transport of these systems and the information to be included in the e-tickets.

2.1.1 Participants

We introduce the participants who are involved in an electronic ticketing system, according to the authors [Fuji 99a, Fuji 99b, Mana 01, Mihl 02, Mats 03, Quer 05]:

- User: receives the electronic ticket and sends it for its verification in order to use the service.

- Issuer: issues the electronic ticket to the user. E-tickets can be issued by both service providers and intermediaries [Siu 01b].

- Service provider: receives the e-ticket from the user and verifies it. If correct, then it provides access to the service.

These are the general and main participants in e-ticketing systems, but some systems include other participants, for example, an *intermediary* or a *broker* [Fuji 99a, Fuji 99b, Kura 02, Wang 04a]. Moreover, if public key cryptography systems [Pate 97, Serb 08] are used, a *Certification Authority* (*CA*) is also included. In some cases, the e-ticketing system is based on the use of Smart-Cards, so the *Smart-Card issuer* is also included in the system [Siu 01a]. The scheme presented in [Chen 07] includes a *user agent* and the *network access service provider*. The system proposed in [Jorn 07] includes user localization, as well as information related to this location. In order to provide this service while preserving user anonymity, the *network provider* is added as a trusted participant. Other systems also consider the possibility to pay for the e-ticket, so that the *payment service provider*, the *bank* and the *credit card issuer* are also participants involved in the system.

2.1.2 Phases

According to most authors, an electronic ticketing system consists of three main phases: *e-ticket payment*, *issue* and *verification* [Elli 99, Fuji 99a, Siu 01a, Siu 01b, Kura 02, Mats 03, Bao 04, Quer 05, Chen 07]. However, these three phases are not strictly defined. Some authors [Pate 97, Pedo 00, Mana 01, Mihl 02] group the payment and issue phases, making it a two-phase system consisting of e-ticket issue and verification. Other proposals [Wang 04b, Arna 06, Chan 06] add a previous registration phase in which users must be identified and authenticated in order to give them permission to use the service. In [Heyd 06], as well as the previous phases, service *start* and *end* are also considered. This lack of standardization in the definition of the electronic ticket phases is mainly due to the great diversity of services where e-tickets can be used [Fuji 98, Bao 04].

2.1.3 Services

The existing proposals have been evaluated depending on the services that can be offered with these systems. Regarding our overview (see Table 2.1), we can say that electronic ticketing systems are mainly oriented to public transport services. Most of these transport services are rail transport [Pate 97, Elli 99, Hane 02, Vald 03, Hane 04, Heyd 06, Jorn 07, Caro 07, Lutg 07, Hane 08], followed by air travel [Bao 04, Wang 04a, Gran 07, Caro 07, Dorn 07, Serb 08], bus transport [Pate 97, Elli 99, Heyd 06, Caro 07, Lutg 07] subway [Pate 97, Elli 99, Vald 03, Heyd 06, Caro 07, Lutg 07], and finally taxi transport [Caro 07]. It is clear then that our focus is especially put on transport e-ticketing systems. In 2006, in Germany, more than 25 e-ticketing projects were intended or in testing phases for public transport [Hane 08], and most of them were thought for short distance journeys.

We can find systems which are running nowadays and which are applied to tolls [McDa 93, Mats 03, Vald 03, Caro 07, Lutg 07], which compare electronic payment systems with electronic ticketing systems. Users pay for the service when they have used it depending on some usage factor and charging the amount of money directly to the credit card accounts. These kinds of services can be implemented by applying Automatic Fare Collection systems (AFC). A similar payment system using e-tickets is applied to location-based services in [Amol 10]. Also a generic e-ticketing system is used in [Kunt 07] as a method for service access control in a trusted computing environment. The rest of the proposals are not related to transport; furthermore, they are oriented to the leisure sector [Pate 97, Kura 00, Kura 02, Bao 04, Caro 07, Bald 10], such as sports or cultural events.

2.1.4 Information

Like paper tickets, electronic tickets must include some basic information for their practical use. In this section, information fields that electronic tickets can include are briefly described:

- Serial number (SN): unique identification of an e-ticket.
- Issuer (IS): entity who is responsible for issuing the e-ticket. This issuer can also be the service provider, or an intermediary.
- Service provider (SP): entity who offers the service to the user.

SERVICES	Air travel	Rail	Bus	Subway	Taxi	Tolls	No transport
[McDa 93]						✓	
[Pate 97]		✓	✓	✓			✓
[Elli 99]		✓	✓	✓			
[Kura 00]							✓
[Kura 02]							✓
[Hane 02]		✓					
[Mats 03]						✓	
[Vald 03]		✓		✓		✓	
[Bao 04]	✓						✓
[Wang 04a]	✓						
[Hane 04]		✓					
[Heyd 06]		✓	✓	✓			
[Gran 07]	✓						
[Jorn 07]		✓					
[Lutg 07]		✓	✓	✓		✓	
[Kunt 07]							✓
[Caro 07]	✓	✓	✓	✓	✓	✓	✓
[Dorn 07]	✓						
[Hane 08]		✓					
[Serb 08]	✓						
[Bald 10]			✓				✓
[Amol 10]							✓
TOTAL	6	10	6	6	1	5	8

Table 2.1: Services for electronic ticketing systems

- User (US): information about the e-ticket owner. In case this field exists in the e-ticket, user anonymity cannot be achieved.
- Service (SV): description of the service contract.
- Terms and conditions (TC): definition of the e-ticket terms and conditions, or alternatively an external link to enable consultation.
- Type of e-ticket (TT): e-ticket includes a field describing its type.

  - Transferability (TF): if this field is permitted, transferability to another user is allowed.
  - Number of uses (NU): information about the allowed number of e-ticket uses.

- Destination (DT): this field is used for transport services in order to have user destination information.
- Attributes (AT): other attributes of the e-ticket that depend on the service (e.g. theatre seat).
- Validity time (VT): it includes two timestamps, start and expiration dates.
- Date of issue (DI): the e-ticket date of issue. Validity time field could be set by including this field together with the terms and conditions.
- Issuer's digital signature (DS): the e-ticket issuer has a public key cryptosystem key pair, which is able to digitally sign the e-ticket.
- Device identification (DV): e-ticket is linked to a specific device.

2.2 Requirements

We have classified the requirements of the electronic tickets into two categories, security and functional requirements.

2.2.1 Security requirements

Definition 2.1 (Authenticity, ATH)**.** *A ticket is authentic when any party can verify that the e-ticket information has been generated by its legitimate issuer.*

The fulfillment of this requirement will help users to verify if the issuer is the legitimated one, avoiding then some kind of fraud. This security requirement is directly related to unforgeability, its inverse property, that is, an e-ticket cannot be forged by an unauthorised party.

Definition 2.2 (Integrity, IT)**.** *An electronic ticket cannot be modified without being detected by any party.*

All the participants have to be able to verify if an e-ticket has been modified. The e-ticket, then, must be issued by its corresponding issuer, and any party can verify that the content inside the ticket was agreed by the issuer.

Definition 2.3 (Non repudiation, NRP)**.** *Any party that sends or generates a message cannot deny its transmission a posteriori.*

In fact, this requirement comprehends authenticity and integrity requirements: if the user cannot deny the emission of an e-ticket, then it is verifiable that she did issue the e-ticket (authenticity) and nobody did modify the content of the e-ticket (integrity). As an example, this requirement is necessary when a user requests an e-ticket issue, and the issuer tries to deny it (especially if there is a payment for the e-ticket).

Definition 2.4 (Fairness, FR)**.** *At the end of an exchange between two or more parties, either everybody achieves the expected items, or nobody can stand in a privileged situation.*

This requirement is closely related to non-repudiation, but goes a step further because it does not only seek to ensure that the parties cannot deny having participated in a transaction *a posteriori*, but also that the parties are committed, in relation to a particular exchange, with fairness. This requirement can be useful for multiple processes related to e-ticket management:

- issue: if the costumer pays the amount that the e-ticket is worth, then she should receive a valid e-ticket from the issuer, and vice versa, if the customer receives a valid e-ticket, she has to pay the corresponding amount or must provide a proof that she has received the e-ticket. We can think of some exceptions: donations (between users), free e-tickets (for some events), etc.
- use: the service provider must provide the service linked to the e-ticket if the client delivers a valid ticket, and vice versa.
- compensation: if the service provider has a valid e-ticket (received from a client) then it must receive, if applicable, the corresponding compensation (typically economic), and if the service provider has received such compensation, then she must provide a proof that she has received it.

A protocol for those exchanges will have to be designed, and some properties achieved. We are in front of a kind of fair exchange of values (an e-ticket for a

payment, a service for an e-ticket), and so, some of the following properties will have to be met: fairness, abuse-freeness, timeliness, verifiability of the TTP, etc. These properties can be found, for instance, in [Ferr 10].

Definition 2.5 (Non-overspending, NOV)**.** *E-tickets can only be used as agreed in the contract between the issuer and the user.*

This requirement is closely related to the reusability requirement. Non-reusable e-tickets cannot be reused once they have been already spent. Reusable e-tickets can be used exactly the number of times agreed in the moment of issue, or in the case it is a time-limitation, they cannot be used once their validity time has expired. Period and usable times can be combined in the same e-ticket (see reusability requirement). Mechanisms to control overspending can affect the requirement of anonymity when attempting to identify fraudsters. Overspending can be prevented or detected. If overspending is detected in the verification phase, overspending will not be allowed (prevention). If it is detected afterwards, some way to identify the overspender(s) will be necessary for any kind of extra charge or penalty if required.

This requirement is also related to the uniqueness requirement of paper-based tickets: they are unique documents. It means we can distinguish original and copy (although some copies are difficult to identify). At least in those cases where making a copy becomes easy, system security is based on the fact that it is difficult to falsify tickets, and so it is difficult to duplicate them. Note that here another requirement is related with uniqueness: forgery. Some authors [Siu 01b] call this property "duplication".

In the electronic world we cannot distinguish between two identical strings of bits. Any accessible electronic document can be duplicated as many times as we want. When we want to talk about non-usable copies of electronic documents, we have to use some technique in order to achieve this requirement:

- tamper-resistant devices (e.g., Smart-Cards), prevent a document stored in that device from being manipulable, so the distribution of these unique documents will be possible among this kind of devices. Then the security in this device is based on the fact that the manipulation cost has to be higher than the benefits that an attacker could obtain from that. When the value of the information stored in the device is high (e.g. bank accounts data), the value can be high and it then becomes a threat.

- some entity keeps track of the used e-tickets in a centralized way, so the uniqueness of the document is not guaranteed, but the uniqueness of the use can be guaranteed. What matters is the information on the central register.

- active proofs of knowledge. Any attacker could obtain a copy of a certain volume of data, namely tickets or messages. But there are techniques to perform verifications in order to ensure that the holder of the ticket is the authorised one (e.g. verify that the receiver in the issue phase is the same that the user in the verification phase).

The controller entity is able to know that an overspending occurs. We can distinguish two different techniques to do so: prevention (the overspending attempt is detected and not allowed, typically with online transactions with that entity), and detection *a posteriori* (fraud could be allowed but assuring further actions like extra-charges or penalysing fraudsters).

Whatever technique(s) used, as we have said previously, only one valid copy of an e-ticket must be legally used.

Definition 2.6 (Non Anonymity, NAN)**.** *Identity of the e-ticket owner must be verifiable.*

Not all the paper-tickets present the same requirements regarding anonymity, so we have to distinguish between some possible scenarios for e-tickets. The first scenario comprehends non-anonymous e-tickets, where the service requires user identification and authentication. It means that the user identity has to be embedded in the e-ticket in some way, in order that the service provider can verify that the user is authorized to spend that e-ticket. It is the case of plane e-tickets.

Definition 2.7 (Revocable Anonymity, RAN)**.** *Anonymity of users has to be guaranteed, but it could be revoked in case of misbehaviour.*

Identity of users is embedded in some way (pseudonyms, encrypted identity), in e-tickets. Only one or a reduced number of trusted authorities are typically able to reveal this identity, mainly when some misbehaviour is detected during some process, which means that the same e-ticket could be used more times than desired. For anonymous e-tickets, anonymity has to be revocable in order to identify the fraudster. Obviously, honest users should remain anonymous or, at least, they should be able to prove they are honest users.

Definition 2.8 (Anonymous e-tickets, AN)**.** *A user of an e-ticket has to remain anonymous.*

Some paper tickets allow that users remain anonymous in front of the issuer, verifier and service provider. Therefore, e-tickets will have to maintain the requirement. This requirement deals with the e-ticket, the way of issuing and the way of spending it. The anonymity remains during the life cycle of the e-ticket. However, depending on the kind of used payment method, the user could be identified in this phase. But, in any case, the user has to be able to spend the e-ticket without any kind of identification. Even colluded issuers and service providers should not be able to break anonymity of consumers. Some kind of e-tickets have to be anonymous, and in no case should it be possible to know the identity of the user, even if it's known that somebody is trying or has tried to overspend the e-ticket.

Definition 2.9 (Unlinkability, UNL)**.** *Several tickets from the same user cannot be attributted to a solely user.*

This requirement is closely related to anonymity. A user could be anonymous in all of her movements (e.g. the information that is self-generated and sent), but these could be traceable, which means that any party could know that this information comes from the same user. This could allow the generation of profiles, despite being anonymous (or not). Untraceability avoids the linkage between the movements performed by the same user.

Definition 2.10 (Short-term linkability, STL)**.** *Several tickets from the same user cannot be linkable between them, except in determined occasions, when some moves need to be linked in a very delimited scenario.*

This requirement is a specification of the previous definition of unlinkability. This definition maintains the avoidance of linkage between the movements that are performed by the same user, except in some special cases where a limited linkage is needed (e.g. entrance and exit of a system, payment and further use of a service), in order to prove the owner's credentials. Apart from these cases, unlinkability with other moves or different payments has to be guaranteed.

Definition 2.11 (Exculpability, EXC)**.** *The service provider cannot falsely accuse an honest user of e-ticket overspending, and the user is able to demonstrate that she has already validated the e-ticket before using it.*

An honest user has to be able to prove that she has validated the e-ticket, and therefore the service provider cannot falsely accuse him.

On the other hand, we have claimed that after the detection of overspending the provider has to be able to identify the overspender or possible overspenders. The ideal technical solution provides the overspender, but if the technical solution provides a set of possible overspenders, those kinds of solutions have to provide some way in order honest users can prove they have used e-ticket according to the issuing conditions, and so they cannot be accused of overspending.

Definition 2.12 (Reusability, REU)**.** *The e-ticket can be used more than once.*

An e-ticket could be used once (non-reusable, can only be spent once) or many times (reusable). In both cases, e-ticket overspending has to be prevented or detected. E-tickets can be used more than once as is the case of some urban transport, where a transport pass can be used for several travels (and a counter is decreased in every travel) or it can be used over a period of time. Even, sometimes, the same e-ticket can be used in different services (for instance, bus and underground in the same city). E-tickets have to incorporate security measures that allow using the e-ticket in the valid period of time or for the number of uses agreed (or a combination of both, time and uses). Some authors name divisibility to this requirement (probably influenced by the similarities between e-ticket and e-money).

Definition 2.13 (Transferability, TF)**.** *One user can transfer her e-ticket to other users.*

Some paper tickets can be transferred to other people (show tickets, bus tickets, etc.). Obviously, it is not the case of non-anonymous tickets, e.g. plane tickets. People receiving an e-ticket in a transfer (not directly from an authorized issuer) has to be able to verify that this e-ticket is valid (it will be easy if non-repudiation, integrity and authenticity are met) and not spent by the transferor entity (or previous transferors). When we are in front of gifts or donations by confident people (a friend, familiar, etc.), no special measures have to be taken. It is a personal matter if afterwards an overspending occurs.

But perhaps e-tickets can be resold, or e-tickets (show entrances) can be a present from a third company (in exchange of buying some product from this company). The receiving entity has to make sure that the e-ticket is valid and has not been spent, although the transferring user may try to overspend the e-ticket, and the transferor entity will therefore need to be able to prove she has not

reused the e-ticket. This problem should be specially handled when anonymity is revocable. Transferability will sometimes make the fairness requirement necessary.

Given the previous explanation, two additional definitions of transferability must be provided.

Definition 2.14 (Weak-Transferability, W-TF)**.** *The e-ticket is transferable but overspending cannot be verified in the transfer phase.*

It means that the e-ticket can be used by a user other than the first owner of the e-ticket, but the receiver of the e-ticket will not be able to verify whether that e-ticket has been provided to multiple users or whether it has been used previously, once she receives the e-ticket. When using the e-ticket, the user will know if it is valid (and perhaps it is too late): the provider will inform the user if the e-ticket has been previously used or not. This drawback can be softened if the recipient is provided some evidence of misuse by the transferor.

Definition 2.15 (Strong-Transferability, S-TF)**.** *The e-ticket is transferable and the receiver can verify that it is a valid e-ticket.*

It means that the receiver can be sure that she will be the only one to be able to use the e-ticket: the e-ticket has not been spent, and the originator will not be able to transfer the same e-ticket to other users.

2.2.2 Functional requirements for e-tickets

There are some other requirements that are not directly related to security, yet they are as important as those explained previously.

Definition 2.16 (Expiry date, EXD)**.** *An e-ticket is only valid during an interval of time.*

The fulfillment of this requirement can be useful in order to limit the size of database containing information of used e-tickets.

Definition 2.17 (Offline verification, OFF)**.** *E-ticket verification can be done without any external connection.*

In some scenarios it will not be possible to contact external databases or Trusted Third Parties in order to verify whether an e-ticket is valid or not. Perhaps it will not be the general case, but a solution for this problem needs to be thought. This requirement is quite related to the adopted security mechanisms.

Definition 2.18 (Online verification, ON)**.** *E-ticket verification requires a persistent connection with a trusted centralized system.*

Typically, the offline option is preferred, alleging costs, possible bottleneck, etc.; but in a e-world where millions of transactions with credit card are made online (with "heavy" SSL connections), and with companies working with great computational power (Google, Facebook, etc.), it seems that this argument is no longer valid. In terms of security, online verification is better for active overspending control.

Definition 2.19 (Portability, PT)**.** *E-tickets must be able to be stored in mobile devices.*

E-tickets, as paper tickets, have to be portable for users. So, a laptop or a personal computer are not necessary to handle e-tickets. Mobile phones, smart cards, etc. will have to be able to store and process e-tickets.

Definition 2.20 (Reduced size, RS)**.** *E-tickets must be as short as possible.*

Typically, e-tickets will be stored in mobile devices (a mobile terminal as a mobile phone, a smart card, etc.), and sometimes these devices will have a limited memory. Therefore, e-tickets have to be as reduced in size as possible.

Definition 2.21 (Flexibility, FX)**.** *E-tickets can be used in multiple environments.*

We can think of a lot of different tickets (plane tickets, bus tickets, concert tickets, museum tickets, etc.). We can either design a specific e-ticket, or adapt a general e-ticket for each application. The adaptation is obviously preferred in order to economize the solution, it becomes standardised and it allows a more generic security analysis.

Definition 2.22 (Ease of use, EU)**.** *The learning of the use of e-tickets must be easy.*

We are thinking on e-ticketing as a solution for general public (using paper tickets nowadays, and not especially confident in electronic means necessarily). Ease of use of e-tickets must be as simple as it is in paper format, and avoiding the generation of new problems for users.

Definition 2.23 (Efficiency, EFF)**.** *Processing an e-ticket must not be resource-consuming.*

We can think about efficiency from two different points of view. Mobile terminals could be limited in terms of computational power (although they are everyday decreasing). Therefore, the cryptographic operations of the protocol should be reduced to the necessary ones. The diversity of communication technologies could also be limited (as an example, in 2013 not all phones still have NFC), and then the protocol needs to be designed with this constrain in mind. Any delay due to verification of e-ticket validity must be reasonable for practicability.

Definition 2.24 (Payment openness, PYO)**.** *Electronic tickets should be paid by the most common payment platforms.*

When designing an e-ticketing system, a payment system embedded to the ticketing system should be used in order to obtain the e-ticket once the payment is made. For this reason, every e-ticketing system should be flexible and accept payments from different (and common) payment platforms in order to bring practicability, and thus promoting acceptability of such systems for their users. However, this "ideal" ecosystem could be not as ideal as expected, since a conflict of interests between companies could arise, as in the case of the NFC ecosystem, in which the phone manufacturers, Mobile Network Operators (MNOs) and banks compete in order to take the main role of the business, even producing a halt on its development.

Definition 2.25 (Globally spendable, GS)**.** *Costumers should be able to spend their e-tickets at any appropriate service provider.*

An electronic ticket must be able to be used when attempting to get a service which is considered in the ticket.

Definition 2.26 (Availability, AV)**.** *E-tickets must be usable when needed.*

This requirement could be seen not only as a security requirement, but also as a generic requirement, since we can detail major issues such as denial of service attacks (difficult to handle), disastrous events (more difficult to handle) or temporal malfunction of the infrastructure (e.g. a power failure). In that case, e-tickets could not be verified, and sometimes the event could not be posponed (a concert, plane, etc.). A procedure to handle these situations needs to be designed.

2.3 Existing security proposals

In this section we classify the e-ticketing proposals by focusing on their privacy. In an e-ticketing system, anonymity is the closest related property to the privacy of users, since this property deals with the secrecy of the identity of the user and it guarantees that that user will not be identified.

In the following sections, the studied proposals have been classified depending on the anonymity compliance and according to the given definitions in §2.2 (definitions 2.6, 2.7 and 2.8). Firstly, we describe anonymity-compliant schemes in §2.3.1. Section §2.3.2 describes the e-ticketing proposals in which anonymity can be guaranteed but revoked in case of a fraudulent user (by overspending or law enforcement). Finally, non-anonymous schemes are detailed in §2.3.3.

Table 2.2 summarises the classification of the proposals that we have examined, as well as their requirements. The basic security requirements of authenticity, non-repudiation and integrity are fulfilled in the majority of the proposals, followed by the control of non-overspending. The table shows that online verification beats offline verification. Thus, either a central authority or synchronisation between the providers are usually needed. Regarding anonymity, there are more non-anonymous proposals than revocable anonymous proposals, as well as a few fully anonymous proposals, which are detailed in the following sections. Finally, there are some which allow transferability of tickets, reusability, and only a few which include expiry date specifically in their proposals.

2.3.1 Anonymous schemes (AN)

The following schemes provide anonymity to e-ticketing users. Most are based on Chaum's blind signature [Chau 83] in order to achieve anonymity.

Patel and Crowcroft [Pate 97] define the security requirements are defined, where anonymity is achieved, as well as the offline mode, although central authority intervention is needed in order to prevent overspending.

In [Fan 98], Fan and Lei make an e-ticketing system proposal for electronic voting purposes. They use Chaum's blind signatures in order to achieve anonymity. Only two types of participants take part in the system: the authority and a group of voters.

Song and Korba [Song 03] propose a system for service payment, providing strong privacy (anonymity) and non-repudiation. This system achieves overspend-

PROPERTIES	ATH	NRP	IT	AN	NAN	RAN	TF	NOV	REU	ON	OFF	EXD
[Pate 97]	✓		✓	✓				✓		✓		
[Fuji 98]	✓	✓	✓			✓	✓		✓	✓		
[Naka 99]	✓	✓	✓			✓		✓			✓	
[Elli 99]	✓				✓		✓	✓		✓		
[Pedo 00]	✓		✓		✓			✓	✓	✓		
[Mana 01]	✓	✓	✓		✓		✓	✓	✓		✓	
[Siu 01a]	✓		✓		✓		✓	✓	✓	✓		
[Siu 01b]	✓	✓	✓		✓	✓	✓	✓		✓	✓	
[Mihl 02]	✓				✓			✓		✓		
[Kura 02]	✓		✓		✓		✓	✓	✓	✓		
[Wang 04a]	✓	✓	✓		✓			✓		✓		
[Wang 04b]	✓				✓			✓		✓		
[Bao 04]	✓	✓	✓	✓	✓		✓	✓	✓	✓		
[Hane 04]	✓	✓	✓	✓				✓		✓	✓	
[Kref 05]	✓	✓	✓	✓				✓		✓		
[Quer 05]	✓	✓	✓			✓	✓	✓			✓	
[Chan 06]	✓	✓			✓		✓			✓		✓
[Heyd 06]	✓					✓	✓		✓	✓	✓	
[Chen 07]	✓	✓	✓			✓		✓			✓	
[Jorn 07]						✓	✓	✓		✓	✓	✓
[Amol 10]	✓	✓	✓			✓				✓		✓

Table 2.2: Comparison of e-tickets' security requirements (the codes are taken from section 2.2)

ing control, protection against ticket loss or stealing, without transferability option. Anonymity is achieved by using Chaum's blind signatures.

Haneberg et al. [Hane 04] present an electronic on-board ticketing scheme, by using a PDA connected to the system through Bluetooth and using Java for all applications. PDAs are chosen for their short-range wireless communications and the display. Anonymity is achieved in this proposal as no personal data is included, and anonymity then only depends on the payment method used.

In [Bao 04], Bao states that either the user or the e-ticket should be identified in order to prevent problems such as malicious attacks. So, depending on the

application, the classification of the scheme presented in this paper could change. So, for some applications (e.g. cinema), the e-ticket may not include the identity of the user and the scheme is anonymous. In other cases, the information of the ticket holder may be inside the electronic ticket, and then the protocol becomes non-anonymous. There is a real relationship between anonymity and transferability in this scheme because they do not need the user identification in the e-ticket. Reusability concerns to other ticket information, such as the user's destination. Online mode is used in this scheme for security reasons: the authors states that offline systems show weaknesses to malicious attacks.

The previous schemes become preferable when total anonymity is considered an essential requirement.

2.3.2 Revocable anonymous schemes (RAN)

In this section, we expose solutions that provide revocable anonymity. If all parties behave correctly, then the anonymity of users can be guaranteed, but if some party misbehaves, users may be identified by a trusted party which would charge them a penalty fee or even take legal actions.

In the Fujimura et al. proposal [Fuji 98], anonymity, transferability and reusability are required. Pseudonyms are proposed if anonymity is required, and overspending is controlled by a central database (online verification).

In [Quer 05], Quercia and Hailes' e-ticketing system proposal is based on Chaum's e-cash blind signatures, providing revocable anonymity to the user (the anonymity is revoked in case of overspending), but there is a severe communication cost that could probably slow down the system. Apart from revokable anonymity, non-repudiation, offline verification and portability are achieved in this proposed system.

In [Heyd 06], Heydt-Benjamin et al. make a proposal using latest advances in e-cash to improve privacy in electronic ticketing systems for public transit. One-time pseudonyms are used in order to achieve anonymity.

Chen et al. [Chen 07] propose the use of mobile devices (mobile phones, smart phones or PDAs) in e-ticketing systems, by taking advantage of their wireless communications. They focus on the compliance of several security requirements such as (revocable) anonymity, non-repudiation, as well as efficient verification. The ticket process is defined in 3 phases in their paper: request, issue and verification. Anonymity is achieved with the use of pseudonyms.

The system defined by Jorns et al. [Jorn 07] is aimed to transport services, as the ticket includes information about the route. The system uses GPS technologies to show the location of the user, and it is used with mobile phones and PDAs. Authencity, non-repudiation and integrity are not fulfilled in that proposal, in which pseudonyms are used in order to achieve revocable anonymity.

Kuntze and Schmidt's proposal [Kunt 07] presents a ticketing system with a pseudonym created by the Trusted Agent (TA) (i.e., the user of a ticket system and associated services operating with her trusted platform), using the identities embodied in the trusted device, and a private Certification Authority (PCA). The system achieves anonymity thanks to the pseudonyms, although the PCA knows the identity of the TA. This can be used to perform a charging for the ticket and the PCA is therefore able to de-anonymise misbehaving participants. In order to protect privacy, the authors point out that the pseudonym can identify a group of many TAs in the system and only the PCA can potentially resolve the individual identity of a TA.

Serban et al. [Serb 08] present an e-ticketing system oriented to air travel e-tickets. A certification authority (CA) is needed to authenticate all participants in the system (sellers, airlines, banks, reputation server) except for users. Users in the system have not to be authenticated then, but credit card payment information is only sent to the bank, as anonymity could not be guaranteed to the user if overspending has been attempted.

Amoli et al. [Amol 10] propose a Location-Based Services (LBS) protocol based on one-time tickets. The ticket lets the user prove that she has been authorized to access to a LBS. The protocol provides anonymity of location as well as the ability of revoking anonymity on the ticket overspending. The ticket disconnects the relation between the location of the mobile user and its identity. The protocol is based on blind signatures and elliptic curve cryptography. There is no need for the parties to trust each other in order that the protocol operates correctly; i.e. it is not possible to make collusion even if the service provider and the ticket issuer cooperate to disclose the identity and location of the user. However, in this protocol, the user gains the trust of the ticket issuer by going through a set of cut-and-choose operations and receives the signed ticket. This technique results in a communicational and computational overhead and may be an open door to the fraud.

The majority of the studied proposals use pseudonyms in order to achieve

revocable anonymity. If pseudonyms are used, real identity information is not included into the ticket, except for its pseudonym. However, if the issuer links every pseudonym to its real identity, then anonymity may be compromised. For that reason, only revocable anonymity for the user could be achieved. In this case, user traceability could be easily performed if the user does not change her pseudonym regularly, because the same pseudonym would be used for different tickets. A certain volume of data could allow some of the involved participants to create user profiles if there are no pseudonym controls.

2.3.3 Non-anonymous schemes (NAN)

The following schemes do not provide anonymity to users, but we have to bear in mind that some services require identified e-tickets, so non-anonymous schemes are not always a drawback.

In Elliott [Elli 99], anonymity is not considered for travel services. The proposal is mainly focused on the use of smartcards to store and manage the electronic tickets.

Pedone [Pedo 00] applies atomic broadcast to e-ticket validation system, where distributed databases could reply to user requests more rapidly, improving server availability and avoiding bottleneck problems, as information is replicated in the distributed servers. Two phases are defined in this paper: e-ticket reception and verification.

Kuramitsu et al. [Kura 00] present an electronic ticketing system that allows transferability between two tamper-proof devices (smart-cards, or alternatively mobile devices that have an internal smart-card). This transfer process guarantees atomicity, which means that the ticket will be totally transferred or not transferred. No digital signature is used to sign the ticket; there is protection only when the e-ticket is transferred by using a secure channel between the two devices.

Siu et al. [Siu 01a] propose an e-ticketing system that uses a smartcard (SIM card of the mobile phone), which defines four participants (merchant, customer, card issuer and service provider) and three process phases (ticket issue, transfer and verification). The ticket is digitally signed, and its verification is done online. Transferability is also allowed through a TTP.

Siu et al. in [Siu 01b] present another system with two options: the ticket can include the identity of the user or not. Obviously, in the first case, the scheme is not anonymous, but in the second case, each user has a wallet to store the e-tickets

and the e-tickets have the identification of the wallet. In this case, the issuer of the wallet knows the link of the wallet and the user, so it can revoke its anonymity.

Maña et al. [Mana 01] present a system in which e-tickets are stored in the SIM card of the mobile phone. They achieve offline verification, non-anonymity, transferability and portability. The ticket is linked to a user identification, so anonymity is not achieved.

Kuramitsu and Sakamura [Kura 02] present a system that uses contactless smart-cards to store e-tickets. The system accesses the database (access control), and checks ticket validity. If the ticket is valid, the user is authorized to access the event by updating the database. This paper introduces severe limitations in smart-card storage capacity, and it describes problems in contactless communication disconnections (causing inconsistency). Moreover, the need to use standarized formats in order to solve the management of specific tickets from different applications is also addressed. This proposal provides transferability, but not anonymity.

Matsuo and Ogata [Mats 03] present an e-ticketing system that can fit with Automatic Fare Collection systems. It consists of a prepaid system, where the ticket has been already received. Then, the user only has to send the ticket for its validation. Smart-cards are used in this scenario for their tamper-proof properties. Wireless communication technologies are used for the transaction. Space and time synchronization is also taken into account for the AFC system, as it uses GPS. This paper considers the existence of three phases: issue, spend, recharge; it also considers three participants in the system: issuer, user, and the shop. Instead of the use of digital signatures for e-ticket verification, the system uses hash functions to minimize verification delays, although several security properties could not be achieved.

The proposal by Wang et al. [Wang 04a] presents an air ticket booking scheme where air travel companies delegate their issue digital signatures to a proxy. This proxy is responsible to sign the ticket. Users could verify integrity and authenticity, as well as the verification of the e-ticket's issue delegation from the air travel companies to the proxy. In this paper, only basic requirements are considered; anonymity and other security requirements are not taken into account.

Wang et al. in [Wang 04b] presented a system that is non-anonymous, where the authentication method is made by the use of a smart-card.

Chang, Wu and Lin [Chan 06] present an online e-ticketing system for mobile users, considering security aspects like ticket theft and verification of the ticket

owner. The tickets are digitally signed, and can also be transferred to another user always with the participation of the TTP. Anonymity is not achieved as every ticket has its serial number, and overspending is controlled by searching on the central database. It also has information of the ticket's expiration date.

Haneberg [Hane 08] presents an application for railway tickets (transport), taking into account advantages and disadvantages in the properties that smart-cards, PDAs and mobile phones have, considered as not-tamper-proof devices. Overspending is controlled by a central server (online mode), and anonymity is not considered in this system.

The SIESTA [Bald 10] is a research project co-funded by Tuscany Region in Italy, which provides automated services to visiting tourists. Concerning the electronic tickets, the developed application allows tourists to use their own phone as a ticket for museums, theaters, public transports or car parks. The protocol is divided into two phases: the acquisition phase, in which the ticket is purchased and downloaded in the internal NFC memory of the phone, and the access phase, in which the user uses the ticket. In this e-ticketing system, the identification number of the NFC device identifies the user. The user information is stored in a database along with the number of her device. So, the authentication is very simple and there is no privacy. Moreover, the method to avoid overspending is quite straightforward: the NFC reader of the system provider deletes the ticket from the user's device.

In these systems, anonymity cannot be granted due to different reasons. Some proposals are addressed to services in which anonymity could not be provided to the user, or simply, these systems are not conceived to achieve anonymity. Some systems consider e-ticket transferability, and in many cases, anonymity cannot be granted because the ticket is already signed, without the possibility to modify the e-ticket information.

2.4 Related publications

This survey of proposals has been published as an ISI-JCR Journal. We show the reference as follows:

- M. Mut-Puigserver, M.M. Payeras-Capellà, J.L. Ferrer-Gomila, A. Vives-Guasch, and J. Castellà-Roca. "A survey of electronic ticketing applied to transport". *Computers & Security, Vol.31, Issue 8, pp. 925–939,*

doi: 10.1016/j.cose.2012.07.004, 2012.

CHAPTER 3

Cryptographic Background

This chapter introduces the cryptographic background, assumptions and definitions in which our proposals are based.

Contents

3.1 Notation

We describe several cryptographic techniques that are used in our proposal, so we unify the notation used.

$x \| y$	concatenation of values x and y
$x \leftarrow value$	assign $value$ to variable x
$x \xleftarrow{R} \mathbb{D}$	assign to x a random value from a domain $\mathbb{D}$. This domain could be $\mathbb{Z}, \mathbb{Z}_p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \ldots$, or even $\{0,1\}^*$ for indicating a random string of data.
$x \stackrel{?}{=} y$	verify if the variable x is equal in value with y

$x \blacktriangleleft y$	replace operation in the internal database of the entity that performs that operation, in which the value represented by x is replaced by y
$hash(x)$	hash function of the value x. This function $hash()$ is a public cryptographic one-way summarizing function that achieves collision-resistance. The notation $hash^n(x)$ is used to describe that the hash function is applied n times over the item x as a chain (i.e. $hash^n(x) = hash^{n-1}(hash(x)) = hash^{n-2}(hash(hash(x))) = \dots$)
h_x	value of the calculation of $hash(x)$
$\mathsf{h}_{(x,n)}$	value of the calculation of $hash^n(x)$
$enc_{sk_\mathcal{E}}(x)$	encryption of the plain content x with the private key of the entity $\mathcal{E}$
$enc_{pk_\mathcal{E}}(x)$	encryption of the plain content x with the public key of the entity $\mathcal{E}$
$dec_{sk_\mathcal{E}}(x)$	decryption of the encrypted content x with the secret key of the entity $\mathcal{E}$
$dec_{pk_\mathcal{E}}(x)$	decryption of the encrypted content x with the public key of the entity $\mathcal{E}$
$sig_\mathcal{E}(x)$	signature of the content x with the private key of the entity $\mathcal{E}$. This notation would be equal to performing the hash function of the content x and encrypting it with the secret key of $\mathcal{E}$, that is $enc_{sk_\mathcal{E}}(hash(x))$
$ver_\mathcal{E}(x)$	verification of the signed content x with the public key of the entity $\mathcal{E}$. This notation would be equal to decrypting the signature with the public key of $\mathcal{E}$, that is $dec_{pk_\mathcal{E}}(sig_\mathcal{E}(x)) = dec_{pk_\mathcal{E}}(enc_{sk_\mathcal{E}}(hash(x)))$ in order to obtain $hash(x)$ and compare it to the hash image of the received content
$op \rightarrow x$	remark that a determined operation op gives a determined value x (e.g. $dec_{sk_\mathcal{E}}(enc_{pk_\mathcal{E}}(x)) \rightarrow x$)

3.2 Bilinear maps

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of the same order p, that is, $|\mathbb{G}_1| = |\mathbb{G}_2| = |\mathbb{G}_T| = p$, where g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$. A bilinear map from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$ is a function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ such that for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_p$, then $e(u^a, v^b) = e(u, v)^{ab}$. Bilinear maps are also called pairings because they link pairs of elements from $\mathbb{G}_1$ and $\mathbb{G}_2$ with elements in $\mathbb{G}_T$. Useful bilinear maps have these three properties:

Bilinearity: for all $u \in \mathbb{G}_1, v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, then $e(u^a, v^b) = e(u, v)^{ab}$.

Non-degeneracy: $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$.

Computability: e is efficiently computable.

Pairings are classified into three different types according to [Galb 08].

Type 1: if $\mathbb{G}_1 = \mathbb{G}_2$, that is, the pairing is symmetric.

Type 2: if $\mathbb{G}_1 \neq \mathbb{G}_2$, that is, the pairing is asymmetric, but there is an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ which is efficiently computable, but not inversely.

Type 3: if $\mathbb{G}_1 \neq \mathbb{G}_2$, that is, the pairing is asymmetric, and there is no efficiently computable isomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$.

3.3 Intractability assumptions

A number of cryptographic protocols rely on the computational intractability assumptions. Below, we detail the main assumptions that our protocols are based on in terms of security.

3.3.1 Factoring related assumptions

Definition 3.1. ***Factoring problem.*** *Given a positive integer $n \in \mathbb{N}$, find its prime factorisation $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$ where the p_i are pairwise distinct primes and $e_i > 0$.*

There is no known algorithm to solve this problem, which is considered computationally intractable. Several cryptographic protocols such as RSA rely on the security of this assumption.

Definition 3.2. ***Strong RSA assumption****. Given a modulus n of unknown factorization and a ciphertext c, it is computationally intractable to find any pair (m, e) such that $c \equiv m^e \pmod{n}$.*

The task can be described as finding the e^{th} roots of a random number modulo n, where $n \in \mathbb{N}$ is a large semiprime (i.e. a product of two large prime numbers $p \in \mathbb{N}$ and $q \in \mathbb{N}$: $n = pq$), and where $2 < e < n$ and is coprime to $\phi(n)$. For large RSA key sizes (more than 1024 bits), no efficient method for solving this problem is known. If an efficient algorithm is ever developed, it would threaten the current or eventual security of RSA-based cryptosystems, namely both public key encryption and digital signatures. The security of RSA [Rive 83] depends on factoring, but some authors argue that there is a difference in the complexity of these problems [Bone 98b].

3.3.2 Discrete logarithm related assumptions

Definition 3.3. ***The Discrete logarithm problem (DLP)****. Consider a cyclic group $\mathbb{G}$ of order p. Given g, g^x for a randomly chosen generator g and random $x \in \mathbb{Z}_p$, it is computationally intractable to recover x.*

Many cryptosystems are designed by taking the DLP assumption as the basis of their security, such as Schnorr signatures or DSA signatures. We can find variants from the same DLP assumption. We detail the two most important ones.

Definition 3.4. ***The Computational Diffie-Hellman problem (CDH)****. Consider a cyclic group $\mathbb{G}$ of order p. Given g, g^a, g^b for a randomly chosen generator g and random $a, b \in \mathbb{Z}_p$, compute the value g^{ab}. It is considered a computationally intractable problem.*

This assumption is a variant of DLP, and it is applied for the Diffie-Hellman key exchange protocol [Diff 76]. The best known algorithm for solving CDH is to actually solve the DLP.

Definition 3.5. ***The Decisional Diffie-Hellman problem (DDH)****. Consider a cyclic group $\mathbb{G}$ of order p. Given g, g^a, g^b for a randomly chosen generator g and random $a, b \in \mathbb{Z}_p$, the probability distributions (g^a, g^b, g^{ab}) and (g^a, g^b, g^c), where c is independent and random $c \in \mathbb{Z}_p$, are computationally indistinguishable.*

In other words, decide whether $c = ab$ or not. This assumption is stronger than CDH, it is a variant of DLP, and is applied for the Diffie-Hellman key exchange and variants, and ElGamal encryption protocol [ElGa 85, Bone 98a] and variants.

Sometimes these assumptions may not apply. This is the case of bilinear maps, where the DDH is easy in $\mathbb{G}_1$ [Bone 98a]. Determine $v_1 = e(g^a, g^b)$ and $v_2 = e(g, g^c)$. If $v_1 = v_2$, we assume $c = ab$, and then:

$$e(g^a, g^b) = e(g, g)^{ab} = e(g, g^{ab}) = e(g, g^c)$$

So if we know that the mapping e is non-degenerate, the equality $e(g^a, g^b) = e(g, g^c)$ is equivalent to $ab = c$. Then an adversary can have significant advantage in deciding DDH given the mapping e. We need other assumptions such as the ones are following.

Definition 3.6. ***The q-Strong Diffie-Hellman problem (q-SDH).*** *Occasionally also known as SDH, omitting then the 'q' parameter. Given two cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order p, two randomly chosen generators $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$ of their respective groups, with an isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ where $g_1 = \psi(g_2)$, the q-SDH problem is a hard computational problem where the (q+2)-tuple $(g_1, g_2, g_2^{\gamma}, g_2^{\gamma^2}, ..., g_2^{\gamma^q}) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$ is the input and the pair $(g_1^{\frac{1}{x+\gamma}}, x) \in \mathbb{G}_1 \times \mathbb{Z}_p$ is the output, for some $x \in \mathbb{Z}_p^*$ such that $x + \gamma \neq 0$.*

This assumption was first presented in [Bone 04a] and also in [Bone 04b]. The best known algorithm to solve the q-SDH is to solve the DLP.

Definition 3.7. ***The Decision Linear Diffie-Hellman problem (DLIN).*** *Given a cyclic group $\mathbb{G}_1$ of order p, and taking $u, v, h, u^a, v^b, h^c \in \mathbb{G}_1$ as input, where $u, v, h \in \mathbb{G}_1$ randomly chosen generators, and random $a, b, c \in \mathbb{Z}_p$, and output yes if $a + b = c$ and no otherwise, as detailed in [Bone 04b]. In other words, it is hard to distinguish (h, h^{a+b}) from (h, h'), being $h' \in \mathbb{Z}_p$ a random independent value.*

This assumption is used when the DDH is easily solved. This is the case of bilinear maps, and was first presented in [Bone 04b]. The best known algorithm to solve the Decision Linear DH problem is to solve the DLP.

3.3.3 Pairing assumptions

Definition 3.8. ***The Bilinear Diffie-Hellman problem (BDH).*** *Given two cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of prime order p, a randomly chosen generator $g_1 \in \mathbb{G}_1$, and a bilinear*

map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$*, the Bilinear Diffie Hellman problem is a hard computational problem as follows. Given values* $\{g_1, g_1^a, g_1^b, g_1^c\} \in \mathbb{G}_1^4$ *for some* $a, b, c \in \mathbb{Z}_p^*$*, compute* $e(g_1, g_1)^{abc} \in \mathbb{G}_T$ *is the output.*

The best known algorithm to solve BDH is to solve DLP. This assumption first appeared in [Joux 00] and later in [Bone 01].

The BDH problem can be generalised to asymmetric bilinear groups [Bone 11], where $\mathbb{G}_1$ is not the same as $\mathbb{G}_2$, and taking as input $(g_1, g_1^a, g_1^b, g_2, g_2^a, g_2^b) \in \mathbb{G}_1^3 \times \mathbb{G}_2^3$, compute $e(g_1, g_2)^{ab} \in \mathbb{G}_T$ as the output.

3.4 Group signatures

We use the short group signature (BBS) scheme [Bone 04b] in order to verify that a user is a correct member of a certain group of users. Next, we introduce the main definitions related to the BBS signature and the group signatures scheme.

We follow the BBS notation for the concept of bilinear maps: $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are multiplicative cyclic groups of a prime order p. Then, g_1 is a generator of $\mathbb{G}_1$, g_2 is a generator of $\mathbb{G}_2$ and ψ is an isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$ where $\psi(g_2) = g_1$. Finally e is a computable bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with the properties of bilinearity, non-degeneracy and computability.

Suppose that the SDH assumption (Definition 3.6) holds on $(\mathbb{G}_1, \mathbb{G}_2)$, and that the Decision Linear assumption (Definition 3.7) holds on $\mathbb{G}_1$. The scheme uses a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ and a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$.
The public values are $g_1, u, v, h \in \mathbb{G}_1$ and $g_2, w \in \mathbb{G}_2$. Here $w = g_2^\gamma$ for some secret $\gamma \in \mathbb{Z}_p$.

3.4.1 Procedures of the group signature scheme

The group signature scheme consists of a tuple of algorithms or procedures $(KeyGen_G, Sign_G, Verify_G, Open_G)$ that are constructed from the same BBS scheme, and they are detailed as follows:

$KeyGen_G(n)$

This algorithm takes a parameter n as input, which is the number of members of the group. The algorithm then has the following steps:

1. select $h \xleftarrow{R} \mathbb{G}_1 \backslash \{1_{\mathbb{G}_1}\}$ and compute $gmsk = (\xi_1, \xi_2)$ as the group manager secret key, where $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p^*$, and set $u, v \in \mathbb{G}_1$ such that $u^{\xi_1} = v^{\xi_2} = h$;

2. select $\gamma \xleftarrow{R} \mathbb{Z}_p^*$ and set $w = g_2^{\gamma}$; and

3. generate for each user $\mathcal{U}_i$, $1 \leq i \leq n$, an SDH tuple (A_i, x_i) by performing: select $x_i \xleftarrow{R} \mathbb{Z}_p^*$ and set $A_i \leftarrow g_1^{\frac{1}{\gamma + x_i}}$. The parameter γ has to remain secret.

$Sign_G(gpk, gsk[i], M)$

Given a group public key $gpk = (g_1, g_2, h, u, v, w)$, a private user's key $gsk[i] = (A_i, x_i)$ and a message $M \in \{0,1\}^*$, compute and output a signature of knowledge $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$. Note that the tuple (T_1, T_2, T_3) is the linear encryption of A, that is: $(T_1, T_2, T_3) = (u^\alpha, v^\beta, Ah^{\alpha+\beta})$ for $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p$. There are also some helper values $\delta \leftarrow x\alpha$ and $\mu \leftarrow x\beta$. The parameter c is the self-generated challenge (hash of the information in the commit information of the proof of knowledge). Finally, $(s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$ are the response values of the proof of knowledge.

1. select $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p$;

2. compute the linear encryption of A: $(T_1, T_2, T_3) \leftarrow (u^\alpha, v^\beta, Ah^{\alpha+\beta})$;

3. compute the helper values $\delta \leftarrow x\alpha$ and $\mu \leftarrow x\beta$;

4. select $r_\alpha, r_\beta, r_x, r_\delta, r_\mu \xleftarrow{R} \mathbb{Z}_p$;

5. compute the values R_1, R_2, R_3, R_4, R_5. For computational simplicity, note that 2 out of the 3 pairings which were needed to calculate R_3 can be already precomputed in the setup, namely $p_2 = e(h, w)$ and $p_3 = e(h, g_2)$, as their value is computed from the group public parameters. Then, only the first pairing $p_1 = e(T_3, g_2)$ needs to be computed when signing. This is a note from a computational point of view, so we maintain the original notation.

$$\begin{aligned} R_1 &\leftarrow u^{r_\alpha}, \qquad & R_2 &\leftarrow v^{r_\beta}, \\ R_3 &\leftarrow e(T_3, g_2)^{r_x} \cdot e(h, w)^{-r_\alpha - r_\beta} \cdot e(h, g_2)^{-r_\delta - r_\mu}, & & \\ R_4 &\leftarrow T_1^{r_x} \cdot u^{-r_\delta}, \qquad & R_5 &\leftarrow T_2^{r_x} \cdot v^{-r_\mu}. \end{aligned} \tag{3.1}$$

6. self-compute the challenge: $c \leftarrow H(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5)$

7. compute the values:

$$\begin{aligned} s_\alpha &\leftarrow r_\alpha + c\alpha, & s_\beta &\leftarrow r_\beta + c\beta, & s_x &\leftarrow r_x + cx, \\ & & s_\delta &\leftarrow r_\delta + c\delta, & s_\mu &\leftarrow r_\mu + c\mu. \end{aligned} \tag{3.2}$$

8. output $\sigma \leftarrow (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$.

$Verify_G(gpk, M, \sigma)$

Given a group public key $gpk = (g_1, g_2, h, u, v, w)$, a message M and a group signature $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$, verify that σ is a valid signature of the message.

1. re-derive R_1, R_2, R_3, R_4, R_5:

$$\begin{aligned} \tilde{R_1} &\leftarrow u^{s_\alpha} / T_1^c, & \tilde{R_2} &\leftarrow v^{s_\beta} / T_2^c, \\ \tilde{R_3} &\leftarrow e(T_3, g_2)^{s_x} \cdot e(h, w)^{-s_\alpha - s_\beta} \cdot e(h, g_2)^{-s_\delta - s_\mu} \cdot (e(T_3, w) / e(g_1, g_2))^c, \\ \tilde{R_4} &\leftarrow T_1^{s_x} / u^{s_\delta}, & \tilde{R_5} &\leftarrow T_2^{s_x} / v^{s_\mu}. \end{aligned} \tag{3.3}$$

2. check that $c \stackrel{?}{=} H(M, T_1, T_2, T_3, \tilde{R_1}, \tilde{R_2}, \tilde{R_3}, \tilde{R_4}, \tilde{R_5})$.

$Open_G(gpk, gmsk, M, \sigma)$

This algorithm is used in order to trace a signature to a concrete signer inside the group. It is only available for the group manager, as it is the holder of the $gmsk$ group manager secret key and knows all the pairs (A_i, x_i). Given a group public key $gpk = (g_1, g_2, h, u, v, w)$, the group manager secret key $gmsk = (\xi_1, \xi_2)$, a message M and a signature $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$, it proceeds as follows.

1. First, recover the user's A by performing: $A \leftarrow T_3 / (T_1^{\xi_1} \cdot T_2^{\xi_2})$;
2. If the elements $\{A_i\}$ of the user's private keys are given to the group manager, then it can look up the user index corresponding to the identity A recovered from the signature.

3.4.2 Zero-knowledge proof procedures of the group signature scheme

During the signature generation, the verifier does not take part of that protocol, but only receives the signature. The need of a active proof of knowledge then appears in order to verify that the signer of the previous message is the one who is performing the proof. We detail the procedures $ZKP_G Commit$, $ZKP_G Response$ and $ZKP_G Verify$:

$ZKP_G Commit(M^*)$

This procedure is performed by the user (prover) who wants to demonstrate to other user (verifier) that she is the right holder of the signed message. Given a public group key $gpk = (g_1, g_2, h, u, v, w)$, a group private key for the user $gsk[i] = (A_i, x_i)$ and a signed message $M^* = (M, \sigma)$ where $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$, it generates the commitment $m' = (T_1, T_2, T_3, R'_1, R'_2, R'_3, R'_4, R'_5)$ as output.

1. we have to demonstrate the ownership of the values $(\alpha, \beta, x, \delta, \mu)$ that have been generated by the signature of M^*, keeping then the resulting values with the linear encryption of A: $(T_1, T_2, T_3) = (u^\alpha, v^\beta, Ah^{\alpha+\beta})$;
2. the values ${r_\alpha}', {r_\beta}', {r_x}', r'_\delta, r'_\mu \overset{R}{\leftarrow} \mathbb{Z}_p$ are selected and then these values are generated:

$$\begin{aligned} R'_1 &\leftarrow u^{{r_\alpha}'}, \qquad R'_2 \leftarrow v^{{r_\beta}'}, \\ R'_3 &\leftarrow e(T_3, g_2)^{{r_x}'} \cdot e(h, w)^{-{r_\alpha}'-{r_\beta}'} \cdot e(h, g_2)^{-r'_\delta - r'_\mu}, \\ R'_4 &\leftarrow T_1^{{r_x}'} \cdot u^{-r'_\delta}, \qquad R'_5 \leftarrow T_2^{{r_x}'} \cdot v^{-r'_\mu}. \end{aligned} \tag{3.4}$$

3. the output $m' = (T_1, T_2, T_3, R'_1, R'_2, R'_3, R'_4, R'_5)$ is generated.

$ZKP_G Response(m', c')$

Given a commitment m' where $m' = (T_1, T_2, T_3, R'_1, R'_2, R'_3, R'_4, R'_5)$ and a challenge c' given by the verifier, the prover generates the response $s' = ({s_\alpha}', {s_\beta}', {s_x}', s'_\delta, s'_\mu)$ where their values are given by:

$$\begin{aligned} {s_\alpha}' &\leftarrow {r_\alpha}' + c'\alpha, \qquad {s_\beta}' \leftarrow {r_\beta}' + c'\beta, \qquad {s_x}' \leftarrow {r_x}' + c'x, \\ s'_\delta &\leftarrow r'_\delta + c'\delta, \qquad s'_\mu \leftarrow r'_\mu + c'\mu. \end{aligned} \tag{3.5}$$

$ZKP_G Verify(m', c', s')$

Given a commitment m' where $m' = (T_1, T_2, T_3, R'_1, R'_2, R'_3, R'_4, R'_5)$, a challenge c' given by the verifier, and the response $s' = ({s_\alpha}', {s_\beta}', {s_x}', s'_\delta, s'_\mu)$ provided by the prover, the verifier checks:

$$\begin{aligned} u^{{s_\alpha}'} &\overset{?}{=} T_1^{c'} \cdot R'_1, \qquad v^{{s_\beta}'} \overset{?}{=} T_2^{c'} \cdot R'_2, \\ e(T_3, g_2)^{{s_x}'} \cdot e(h, w)^{-{s_\alpha}'-{s_\beta}'} \cdot e(h, g_2)^{-s'_\delta - s'_\mu} &\overset{?}{=} (e(g_1, g_2)/e(T_3, w))^{c'} \cdot R'_3, \\ T_1^{{s_x}'} \cdot u^{-s'_\delta} &\overset{?}{=} R'_4, \qquad T_2^{{s_x}'} \cdot v^{-s'_\mu} \overset{?}{=} R'_5. \end{aligned} \tag{3.6}$$

CHAPTER 4

Secure Electronic Ticketing system with Exculpability and Reusability

This chapter presents our contribution to electronic ticketing systems, by detailing the desired security requirements and adding exculpability and reusability. Moreover, we show the experimental results performed on a mobile device scenario for users.

Contents

We present our proposal in §4.1, describing the security requirements to be guaranteed, the involved participants and the different phases. In §4.2 we evaluate the security and privacy of the presented system. We have performed some implementation of this scheme in a mobile phone; we detail these implementation tests and their obtained results in §4.3. Finally, we explain the conclusions and related publications in §4.4.

4.1 Description of the e-ticketing scheme

This e-ticketing scheme is designed for mobile devices, reducing the computational requirements in the user side, as well as providing some security requirements. We detail these requirements in §4.1.1, and we introduce the new exculpability and reusability properties. We enumerate the involved participants and phases of the system in §4.1.2. For simplicity, the scenario is designed taking into account that one service can be given by a determined service provider; then the scenario with multiple providers is then discussed in §4.1.3. In table 4.1, we define the details of our proposal, as well as some notation that is used in the scheme.

4.1.1 Security requirements

The security requirements in digital format have to fulfill, at least, the same security requirements that are fulfilled in paper format in order to be successful. We start presenting these security requirements, and we also introduce *exculpability* and *reusability*. We make a brief description of these requirements seen from the particular case of this scheme, and also reference the general definitions that are stated in §2.2 which are fulfilled in the scheme.

The electronic ticketing scheme pretends to guarantee the following security requirements:

- Authenticity (Def. 2.1). A ticket is authentic when any party can verify that the e-ticket information has been generated by its legitimate issuer.
- Integrity (Def. 2.2). An electronic ticket cannot be modified without being detected by any party.
- Non-repudiation (Def. 2.3). The requirement of non repudiation comprehends the fulfillment of the non repudiation of origin and receipt. That means neither the issuer nor the receiver of an e-ticket can deny its emission.
- Non-overspending (Def. 2.5). E-tickets can only be used as agreed in the contract between the issuer and the user.
- Revocable anonymity (Def. 2.7). Anonymity of users has to be guaranteed, but it could be revoked in case of misbehaviour.

- Exculpability (Def. 2.11). None of the analyzed proposals deals with exculpability; that is, the service provider cannot falsely accuse the user of ticket overspending, and the user is able to demonstrate that she has already validated the ticket before using it. Exculpability is a key requirement in our proposal, as the e-ticketing scheme should ensure that either both parties (users and provider) receive their desired data (e-ticket and the validated e-ticket) from each other or none of them do (fair exchange). The parties agree to reveal their data only if the other party also agrees. If any party deviates from the scheme, it then can be identified by the Trusted Third Party (TTP) as the misbehaver. Our scheme defined in §4.1 takes exculpability as a security requirement for an e-ticketing system, as the first step to include this security requirement in future works.

- Reusability (Def. 2.12). A ticket could be used once (non-reusable) or many times (reusable). In both cases, ticket overspending has to be prevented. Tickets can be used more than once as it is the case of some urban transport, where a transport pass can be used for several journeys (and a counter is decreased in every journey) or it can be used over a period of time. The same ticket can sometimes even be used in different places (for instance, bus and underground in the same city). E-tickets have to incorporate security measures that allow using the ticket in the valid period of time or for the number of uses agreed (or a combination of both, time and uses). Some authors call this property divisibility (probably influenced by the similarities between e-ticket and e-cash).

- Expiry date (Def. 2.16). A ticket is valid only during a determined time interval.

- Online/Offline (Def. 2.17, 2.18). Ticket verification does not require a persistent connection with a centralised system by default. In case that there is only one provider giving service, the service can then be offline. However, a case of multiple providers giving one service could be resolved by the use of a persistent connection to a centralised (and trusted) system, or communication between all the providers that can offer the same service.

Table 4.1: Details of the proposal: security requirements, participants, ticket and receipt information

Security Requirements			
Authenticity		Non-repudiation	
Integrity		Revocable anonymity	
Non-overspending		Offline verification	
Expiry date		Exculpability	
Reusability			
Participants			
User	$\mathcal{U}$	Service Provider	$\mathcal{P}$
Ticket Issuer	$\mathcal{I}$	Trusted Third Party	$\mathcal{T}$
Ticket Information (T)			
Serial number	Sn	Issuer	Is
Service	Sv	Terms and conditions	Tc
User pseudonym	$Pseu_{\mathcal{U}}$	Attributes	At
Type of ticket	Ty	Verification data	$\delta_{\mathcal{T},\mathcal{P}}$
Validity time	Tv	Date of issue	Ti
Exculpability ($\mathcal{U}$)	$\mathsf{h}_{(r_{\mathcal{U}},n)}$	Exculpability ($\mathcal{P}$)	$\mathsf{h}_{(r_{\mathcal{I}},n)}$
Digital signature of $\mathcal{I}$	$sig_{\mathcal{I}}(\mathsf{T})$		
Receipt Information (R)			
Exculpability ($\mathcal{P}$)	$A_{\mathcal{P}}$	Timestamp	τ_i
Ticket serial number	T.Sn	Digital signature of $\mathcal{P}$	$sig_{\mathcal{P}}(\mathsf{R})$

4.1.2 Participants and phases

The scheme has the following participants: the user $(\mathcal{U})$, the ticket issuer $(\mathcal{I})$, the service provider $(\mathcal{P})$, and the TTP $(\mathcal{T})$. The phases of our system consist of the traditional phases (ticket purchase and verification), and another phase is added in order to register and obtain temporal pseudonyms without linkage to the identity of users (if they behave correctly) in order to achieve anonymity. We can see in Figure 4.1 the diagram of the entire protocol, with all its participants and phases. Then, the resulting phases are:

- *Pseudonym renewal*, where the user obtains a new temporal pseudonym to be

used in the system;

- *Ticket purchase*, which consists of the payment of the service and reception of the ticket; and

- *Ticket verification*, where the user shows the ticket to the service provider in order to be checked and verified.

Other phases considered in the system are claims. These claims should be only executed in case of controversial situations during the *Ticket verification* phase:

- *Claim* m_2 *not received*, when $\mathcal{U}$ sends the first step of the verification m_1 but does not receive m_2 by $\mathcal{P}$, or the information is not correct;

- *Claim* m_3 *not received*, when $\mathcal{P}$ sends the second step of the verification m_2 but does not receive m_3 by $\mathcal{U}$, or the information is not correct; and

- *Claim* m_4 *not received*, when $\mathcal{U}$ sends the third step of the verification m_3 but does not receive m_4 by $\mathcal{P}$, or the information is not correct.

These situations will be explained in the following sections.

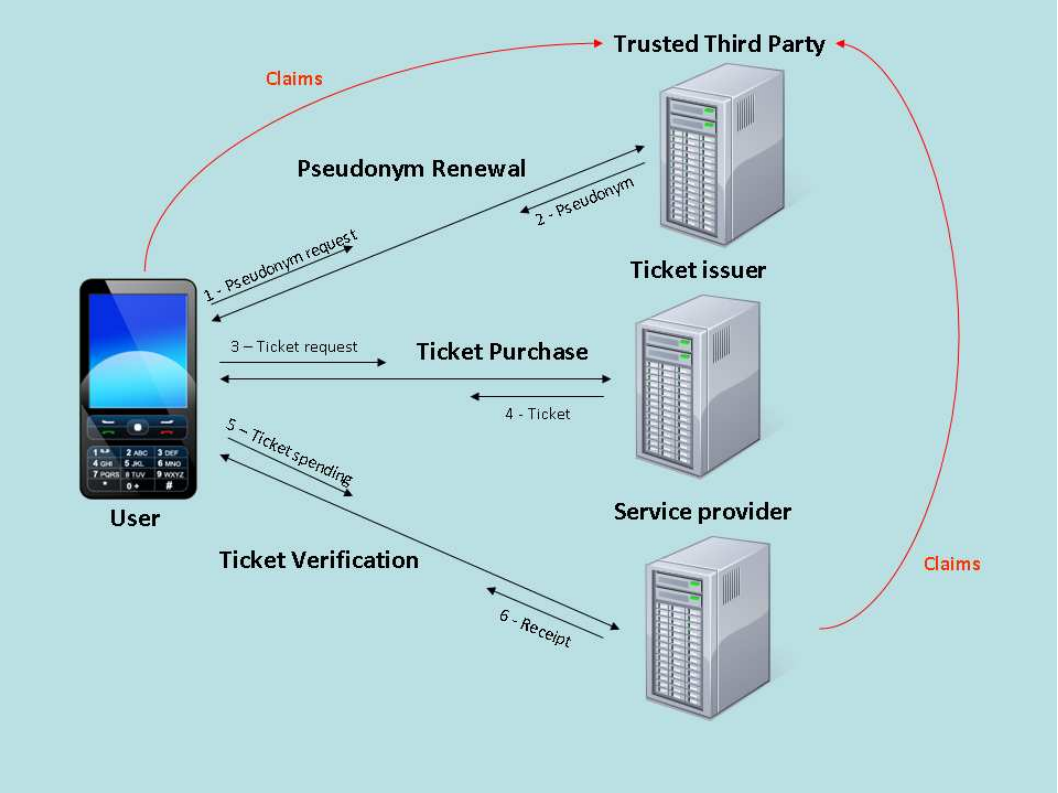

Figure 4.1: Diagram of the entire protocol

Pseudonym renewal

The user $\mathcal{U}$ contacts the pseudonym manager $\mathcal{T}$ in order to renew the assigned pseudonym. Users have a digital credential ($Cert_{\mathcal{U}}$) for authentication to the TTP only, as the system is anonymous, and all further movements in the system are tracked only with the assigned temporal pseudonym ($Pseu_{\mathcal{U}}$). The certificate $Cert_{\mathcal{U}}$ identifies $\mathcal{U}$ through a secure connection established between the two parties. The system has the public parameters (α, p, q), where α is a generator of the group $\mathbb{G}$ with order p, being p and q large primes achieving $p = 2q + 1$. $\mathcal{U}$ generates a random value $x_{\mathcal{U}} \xleftarrow{R} \mathbb{Z}_q$ and computes $y_{\mathcal{U}} \leftarrow \alpha^{x_{\mathcal{U}}} \pmod{p}$ in order to receive a valid signed pseudonym $Pseu_{\mathcal{U}}$ from $\mathcal{T}$. $\mathcal{U}$ and $\mathcal{T}$ have their own pair of keys used for signature and encryption of the transmitted data between them.

authenticateUser User $\mathcal{U}$ follows the next steps:

1. generates $x_{\mathcal{U}} \xleftarrow{R} \mathbb{Z}_q$, and computes $y_{\mathcal{U}} \leftarrow \alpha^{x_{\mathcal{U}}} \pmod{p}$;
2. computes the signature $sig_{\mathcal{U}}(y_{\mathcal{U}}) = enc_{sk_{\mathcal{U}}}(\mathsf{h}_{y_{\mathcal{U}}})$ where $\mathsf{h}_{y_{\mathcal{U}}} = hash(y_{\mathcal{U}})$
3. concatenates and encrypts the information to be sent $(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})$with the $\mathcal{T}$'s public key as a digital envelope: $enc_{pk_{\mathcal{T}}}(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})$;
4. sends $enc_{pk_{\mathcal{T}}}(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})$ to $\mathcal{T}$;

generatePseudonym Pseudonym Manager $\mathcal{T}$ executes:

1. decrypts $dec_{sk_{\mathcal{T}}}(enc_{pk_{\mathcal{T}}}(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})) \rightarrow (y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})$;
2. verifies $y_{\mathcal{U}}$: $ver_{pk_{\mathcal{U}}}(sig_{sk_{\mathcal{U}}}(\mathsf{h}_{y_{\mathcal{U}}})) \rightarrow (\mathsf{h}_{y_{\mathcal{U}}}) \stackrel{?}{=} hash(y_{\mathcal{U}})$;
3. if correct, then computes the signature of $sig_{\mathcal{T}}(y_{\mathcal{U}}) = enc_{sk_{\mathcal{T}}}(\mathsf{h}_{y_{\mathcal{U}}})$;
4. encrypts the signature with the $\mathcal{U}$'s public key: $enc_{pk_{\mathcal{U}}}(sig_{\mathcal{T}}(y_{\mathcal{U}}))$; and
5. sends $enc_{pk_{\mathcal{U}}}(sig_{\mathcal{T}}(y_{\mathcal{U}}))$ to $\mathcal{U}$.

verifyPseudonym $\mathcal{U}$ computes:

1. decrypts $dec_{sk_{\mathcal{U}}}(enc_{pk_{\mathcal{U}}}(sig_{\mathcal{T}}(y_{\mathcal{U}}))) \rightarrow (sig_{\mathcal{T}}(y_{\mathcal{U}}))$;
2. verifies the TTP signature of $y_{\mathcal{U}}$: $ver_{pk_{\mathcal{T}}}(sig_{sk_{\mathcal{T}}}(\mathsf{h}_{y_{\mathcal{U}}})) \rightarrow (\mathsf{h}_{y_{\mathcal{U}}}) \stackrel{?}{=} hash(y_{\mathcal{U}})$;

Note that the $\mathsf{h}_{(item,n)}$ is used as the value of the calculation of $hash^n(item)$ as depicted in Figure 4.2.

User ($\mathcal{U}$)	**Pseudonym Manager ($\mathcal{T}$)**
Pseudonym Renewal	
$x_{\mathcal{U}} \xleftarrow{R} \mathbb{Z}_q$	
$y_{\mathcal{U}} \leftarrow \alpha^{x_{\mathcal{U}}} \pmod p$	
$\mathsf{h}_{y_{\mathcal{U}}} \leftarrow hash(y_{\mathcal{U}})$	
$sig_{\mathcal{U}}(y_{\mathcal{U}}) = enc_{sk_{\mathcal{U}}}(\mathsf{h}_{y_{\mathcal{U}}})$	
$enc_{pk_{\mathcal{T}}}(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})$	
$\xrightarrow{enc_{pk_{\mathcal{T}}}(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})}$	
	$dec_{sk_{\mathcal{T}}}(enc_{pk_{\mathcal{T}}}(y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})) \rightarrow (y_{\mathcal{U}} \| sig_{\mathcal{U}}(y_{\mathcal{U}}) \| Cert_{\mathcal{U}})$
	$ver_{\mathcal{U}}(sig_{\mathcal{U}}(\mathsf{h}_{y_{\mathcal{U}}})) \rightarrow (\mathsf{h}_{y_{\mathcal{U}}}) \stackrel{?}{=} hash(y_{\mathcal{U}})$
	$sig_{\mathcal{T}}(y_{\mathcal{U}}) = enc_{sk_{\mathcal{T}}}(\mathsf{h}_{y_{\mathcal{U}}})$
	$enc_{pk_{\mathcal{U}}}(sig_{\mathcal{T}}(y_{\mathcal{U}}))$
$\xleftarrow{enc_{pk_{\mathcal{U}}}(sig_{\mathcal{T}}(y_{\mathcal{U}}))}$	
$dec_{sk_{\mathcal{U}}}(enc_{pk_{\mathcal{U}}}(sig_{\mathcal{T}}(y_{\mathcal{U}}))) \rightarrow (sig_{\mathcal{T}}(y_{\mathcal{U}}))$	
$ver_{\mathcal{T}}(sig_{\mathcal{T}}(\mathsf{h}_{y_{\mathcal{U}}})) \rightarrow (\mathsf{h}_{y_{\mathcal{U}}}) \stackrel{?}{=} hash(y_{\mathcal{U}})$	

Table 4.2: Pseudonym Renewal subprotocol

Ticket purchase

The user establishes a connection with the ticket issuer $\mathcal{I}$ in order to receive the ticket. This connection could be established through an anonymous channel like TOR [Ding 04], thus guaranteeing user's privacy. There are current contributions [1] that have implemented TOR for mobile devices with Android. $\mathcal{I}$ has a key pair and its public key certificate ($Cert_{\mathcal{I}}$). Users do not use their personal keys (it would cause loss of anonymity); they use the temporal pseudonyms and authenticate through the Schnorr's Zero-Knowledge Proof (ZKP) [Schn 91]. The payment method is considered as out of scope in this proposal as we focus on the privacy given to user when joining/exiting the system, and using the service.

$\mathcal{I}$ generates the ticket with the information and its digital signature, together with the secret value $r_{\mathcal{I}}$ and the secret shared key (they are decryptable only by $\mathcal{P}$ and $\mathcal{T}$) in order to let the provider show the secret value $r_{\mathcal{I}}$ later, in the verification phase. The ticket issuer $\mathcal{I}$ and the user $\mathcal{U}$ follow this protocol:

getService $\mathcal{U}$ executes:

1. selects and pays for the desired service Sv;
2. generates a random value $r_{\mathcal{U}} \overset{R}{\leftarrow} \mathbb{Z}_q$, and computes $\mathsf{h}_{(r_{\mathcal{U}},n)} \leftarrow hash^n(r_{\mathcal{U}})$, where n is the predefined maximum number of times that the e-ticket can be spent;
3. computes $H_{\mathcal{U}} \leftarrow \alpha^{r_{\mathcal{U}}} \pmod{p}$;
4. generates two more random values $a_1, a_2 \overset{R}{\leftarrow} \mathbb{Z}_q$ to be used in the Schnorr proof;
5. computes $A_1 \leftarrow \alpha^{a_1} \pmod{p}$;
6. computes $A_2 \leftarrow \alpha^{a_2} \pmod{p}$;
7. sends $(Pseu_{\mathcal{U}} \| H_{\mathcal{U}} \| A_1 \| A_2 \| \mathsf{h}_{(r_{\mathcal{U}},n)} \| \mathsf{Sv})$ to the ticket issuer $\mathcal{I}$.

getChallenge $\mathcal{I}$ follows the next steps:

1. generates and sends a challenge $c \overset{R}{\leftarrow} \mathbb{Z}_q$ for $\mathcal{U}$;
2. asynchronously, for optimization, pre-computes ${y_{\mathcal{U}}}^c \pmod{p}$;
3. asynchronously, for optimization, pre-computes ${H_{\mathcal{U}}}^c \pmod{p}$;

solveChallenge $\mathcal{U}$ computes:

[1] http://sourceforge.net/apps/trac/silvertunnel/wiki/TorJavaOverview

1. computes $w_1 \leftarrow a_1 + c \cdot x_{\mathcal{U}} \pmod{q}$;
2. computes $w_2 \leftarrow a_2 + c \cdot r_{\mathcal{U}} \pmod{q}$;
3. encrypts $(w_1 \| w_2)$ and sends it to $\mathcal{I}$: $enc_{pk_{\mathcal{I}}}(w_1 \| w_2)$;
4. pre-computes the shared session key used in the ticket verification: $K \leftarrow hash(w_2)$;

getTicket $\mathcal{I}$ follows the next steps:

1. decrypts $dec_{sk_{\mathcal{I}}}(enc_{pk_{\mathcal{I}}}(w_1 \| w_2)) \rightarrow (w_1 \| w_2)$;
2. computes $\alpha^{w_1} \pmod{p}$;
3. computes $\alpha^{w_2} \pmod{p}$;
4. verifies $\alpha^{w_1} \stackrel{?}{=} A_1 \cdot {y_{\mathcal{U}}}^c \pmod{p}$;
5. verifies $\alpha^{w_2} \stackrel{?}{=} A_2 \cdot {H_{\mathcal{U}}}^c \pmod{p}$;
6. computes the shared session key: $K \leftarrow hash(w_2)$;
7. obtains a unique serial number Sn, and a random value $r_{\mathcal{I}} \stackrel{R}{\leftarrow} \mathbb{Z}_p$;
8. computes $\mathsf{h}_{(r_{\mathcal{I}},n)} \leftarrow hash^n(r_{\mathcal{I}})$;
9. composes $\kappa \leftarrow (K \| r_{\mathcal{I}})$ and signs it $\kappa^* \leftarrow (\kappa \| sig_{\mathcal{I}}(\kappa))$;
10. encrypts κ^* with a digital envelope which is decryptable by the TTP $\mathcal{T}$ and the provider $\mathcal{P}$ for possible future controversial situations during the ticket verification: $\delta_{\mathcal{T},\mathcal{P}} \leftarrow enc_{pk_{\mathcal{T},\mathcal{P}}}(\kappa^*)$. This is a mechanism that prevents $\mathcal{I}$ from forging $r_{\mathcal{I}}$, because $\mathcal{T}$ can check that information and, demonstrate that $\mathcal{I}$ is the culprit;
11. fills out the ticket information $\mathsf{T} \leftarrow (\mathsf{Sn} \| \mathsf{Sv} \| Pseu_{\mathcal{U}} \| \mathsf{Tv} \| \mathsf{Ti} \| \mathsf{h}_{(r_{\mathcal{I}},n)} \| \mathsf{h}_{(r_{\mathcal{U}},n)} \| \delta_{\mathcal{T},\mathcal{P}})$;
12. digitally signs the ticket T, and obtains the signed ticket, $sig_{\mathcal{I}}(\mathsf{T}) \leftarrow sig_{sk_{\mathcal{I}}}(hash(\mathsf{T}))$, and $\mathsf{T}^* \leftarrow (\mathsf{T} \| sig_{\mathcal{I}}(\mathsf{T}))$;
13. sends T^* to the user $\mathcal{U}$;

receiveTicket $\mathcal{U}$ executes:

1. verifies the digital signature $sig_{\mathcal{I}}(\mathsf{T})$ of the ticket T using the issuer's certificate;
2. verifies that ticket T data and the performed request match;
3. verifies the ticket validity $(\mathsf{T.Ti}, \mathsf{T.Tv})$;
4. verifies $\mathsf{T}.Pseu_{\mathcal{U}}$;
5. stores $(\mathsf{T}^*, r_{\mathcal{U}}, j = 0)$ in the device. We set up j to 0 because represents the times that the e-ticket has been used. The e-ticket will be totally

consumed when $j = n$.

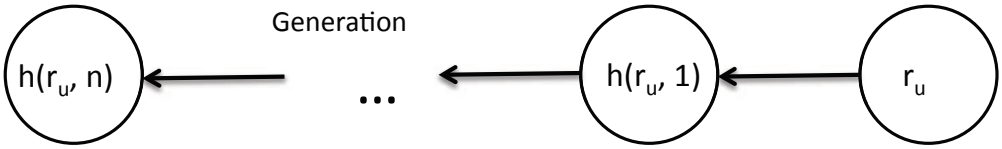

Figure 4.2: Hash Chain

User ($\mathcal{U}$)		**Issuer ($\mathcal{I}$)**
	Ticket Purchase	
select Sv		
$r_{\mathcal{U}} \xleftarrow{R} \mathbb{Z}_q$		
$\mathsf{h}_{(r_{\mathcal{U}},n)} \leftarrow hash^n(r_{\mathcal{U}})$		
$H_{\mathcal{U}} \leftarrow \alpha^{r_{\mathcal{U}}} \pmod p$		
$a_1, a_2 \xleftarrow{R} \mathbb{Z}_q$		
$A_1 \leftarrow \alpha^{a_1} \pmod p$		
$A_2 \leftarrow \alpha^{a_2} \pmod p$		
	$(Pseu_{\mathcal{U}} \Vert H_{\mathcal{U}} \Vert A_1 \Vert A_2 \Vert \mathsf{h}_{(r_{\mathcal{U}},n)} \Vert \mathsf{Sv}) \longrightarrow$	
		$c \xleftarrow{R} \mathbb{Z}_q$
	$\longleftarrow c$	
$w_1 \leftarrow a_1 + c \cdot x_{\mathcal{U}} \pmod q$		${y_{\mathcal{U}}}^c \pmod p$
$w_2 \leftarrow a_2 + c \cdot r_{\mathcal{U}} \pmod q$		${H_{\mathcal{U}}}^c \pmod p$
$enc_{pk_{\mathcal{I}}}((w_1 \Vert w_2))$		
	$enc_{pk_{\mathcal{I}}}((w_1 \Vert w_2)) \longrightarrow$	
$K \leftarrow hash(w_2)$		$dec_{sk_{\mathcal{I}}}(enc_{pk_{\mathcal{I}}}(w_1 \Vert w_2)) \rightarrow (w_1 \Vert w_2)$
		$\alpha^{w_1} \pmod p$
		$\alpha^{w_2} \pmod p$
		$\alpha^{w_1} \stackrel{?}{=} A_1 \cdot {y_{\mathcal{U}}}^c \pmod p$
		$\alpha^{w_2} \stackrel{?}{=} A_2 \cdot {H_{\mathcal{U}}}^c \pmod p$
		$K \leftarrow hash(w_2)$
		obtains a unique serial number Sn
		$r_{\mathcal{I}} \xleftarrow{R} \mathbb{Z}_p$
		$\mathsf{h}_{(r_{\mathcal{I}},n)} \leftarrow hash^n(r_{\mathcal{I}})$
		$\kappa \leftarrow (K \Vert r_{\mathcal{I}})$
		$\kappa^* \leftarrow (\kappa \Vert sig_{\mathcal{I}}(\kappa))$
		$\delta_{\mathcal{T},\mathcal{P}} \leftarrow enc_{pk_{\mathcal{T},\mathcal{P}}}(\kappa^*)$
		$\mathsf{T} \leftarrow (\mathsf{Sn} \Vert \mathsf{Sv} \Vert Pseu_{\mathcal{U}} \Vert \mathsf{Tv} \Vert \mathsf{Ti} \Vert \mathsf{h}_{(r_{\mathcal{I}},n)} \Vert \mathsf{h}_{(r_{\mathcal{U}},n)} \Vert \delta_{\mathcal{T},\mathcal{P}} \Vert ...)$
		$sig_{\mathcal{I}}(\mathsf{T}) = sig_{sk_{\mathcal{I}}}(hash(\mathsf{T}))$, and $\mathsf{T}^* \leftarrow (\mathsf{T} \Vert sig_{\mathcal{I}}(\mathsf{T}))$
	$\longleftarrow \mathsf{T}^*$	
verifies the digital signature $sig_{\mathcal{I}}(\mathsf{T})$		
verifies that ticket T data		
verifies the ticket validity $(\mathsf{T.Ti}, \mathsf{T.Tv})$		
verifies $\mathsf{T}.Pseu_{\mathcal{U}}$		
stores $(\mathsf{T}^*, r_{\mathcal{U}}, j = 0)$ in the device		

Table 4.3: Ticket Purchase subprotocol

Ticket verification

When the user wants to use the service, she must verify the ticket in advance. For simplicity, we present the ticket verification with only one provider, so the service provider $\mathcal{P}$ never needs permanent communication with the ticket issuer. Nonetheless, the protocol can be extended for multiple providers. In that case, all the service providers should be connected to a central repository of spent tickets in order to control ticket overspending. In all these situations, when a ticket starts its verification process, the database has to lock its item (keyed with the unique serial number of the ticket) in order to allow concurrent accesses to the database for different tickets; in this case, if another user tried to verify the same ticket in another provider concurrently must get an error. The user only interacts with the service provider, but in controversial situations, she and/or the service provider could interact directly with the TTP through a resilient connection in order to preserve the security requirements of the protocol. If user misbehaved, her identity could be revoked, enabling to take further actions. $\mathcal{U}$ sends the ticket T^*, and $\mathcal{P}$ checks it. If passed, $\mathcal{P}$ sends the commitment so that $r_{\mathcal{I}}$ will be disclosed if $\mathcal{U}$ behaves correctly. Once the user sends the secret value $r_{\mathcal{U}}$ encrypted through a shared key, then she receives the secret $r_{\mathcal{I}}$ together with the receipt R^* from $\mathcal{P}$. The service provider $\mathcal{P}$ and the user $\mathcal{U}$ take the following steps:

showTicket $\mathcal{U}$ computes:

1. sends ticket $\mathsf{m}_1 = (\mathsf{T}^* \| i)$ to $\mathcal{P}$. As a general case, we suppose that the service costs s of the n times that the e-ticket can be spent. So, the value i is computed as $i \leftarrow j + s$;

verifyTicket $\mathcal{P}$ executes:

1. verifies the ticket signature, T.Sv, T.Ti, and T.Tv;
2. if the verifications fail, $\mathcal{P}$ omits m_1, and aborts the ticket verification;
3. else $\mathcal{P}$ looks for the ticket T^* in the database using T.Sn and locking this item; later, it verifies that the ticket has not been spent by retreiving the information related to the ticket $(j, \mathsf{h}_{(r_{\mathcal{U}}, n-j)})$ in the provider's database (if no information is found, then j is set to $j = 0$):
   (a) if $(i > j)$ then:
      i. computes $A_{\mathcal{P},i} \leftarrow PRNG(\mathsf{h}_K) \oplus \mathsf{h}_{(r_{\mathcal{I}}, n-i)}$, where $PRNG(\mathsf{h}_K)$ is a secure pseudorandom number generator and, $\mathsf{h}_K \leftarrow hash(K)$ is

the seed. Note that K and $r_{\mathcal{I}}$ are obtained from $\delta_{\mathcal{T},\mathcal{P}}$, then the provider is able to compute $\mathsf{h}_{(r_{\mathcal{I}},n-i)} \leftarrow hash^{(n-i)}(r_{\mathcal{I}})$;

ii. encrypts $A_{\mathcal{P},i}$ with the public key of the TTP $\mathcal{T}$: $enc_{pk_{\mathcal{T}}}(A_{\mathcal{P},i})$;

iii. stores $A_{\mathcal{P},i}$ for future use;

iv. assigns $\mathsf{V}_{\mathsf{succ}} \leftarrow (\mathsf{T.Sn} \| \mathtt{flag}_1 \| \tau_1 \| enc_{pk_{\mathcal{T}}}(A_{\mathcal{P},i}) \| j)$, ($\tau_1$ is the verification timestamp). The $\mathtt{flag}_1$ indicates that the ticket is valid and has not been spent yet. The signature is noted: $\mathsf{V}_{\mathsf{succ}}{}^* \leftarrow (\mathsf{V}_{\mathsf{succ}} \| sig_{\mathcal{P}}(\mathsf{V}_{\mathsf{succ}}))$;

v. sends $\mathsf{m}_2 = \mathsf{V}_{\mathsf{succ}}{}^*$ to $\mathcal{U}$;

(b) if $(i \leq j)$ then:

i. computes $\mathsf{h}_{(r_{\mathcal{U}},n-i)} \leftarrow hash^{(j-i)}(\mathsf{h}_{(r_{\mathcal{U}},n-j)})$

ii. assigns $\mathsf{V}_{\mathsf{fail}} \leftarrow (\mathsf{T.Sn} \| \mathsf{h}_{(r_{\mathcal{U}},n-i)} \| \mathtt{flag}_0 \| i \| \tau_1)$. The $\mathtt{flag}_0$ indicates that the ticket has been spent, i.e. it is not valid. The signature is noted: $\mathsf{V}_{\mathsf{fail}}{}^* \leftarrow (\mathsf{V}_{\mathsf{fail}} \| sig_{\mathcal{P}}(\mathsf{V}_{\mathsf{fail}}))$;

iii. sends $\mathsf{m}_2 = \mathsf{V}_{\mathsf{fail}}{}^*$ to $\mathcal{U}$;

showProof $\mathcal{U}$ executes:

1. verifies $\mathcal{P}$'s signature;
2. if $\mathsf{V}_{\mathsf{succ}}{}^*$ or either $\mathsf{V}_{\mathsf{fail}}{}^*$ are not received, the *Claim* m_2 *not received* is called;
   (a) if $\mathsf{V}_{\mathsf{fail}}{}^*$ is received, $\mathcal{U}$ aborts the verification process. If the response is not correct, $\mathcal{U}$ can contact the TTP to reconsider the situation by calling *Claim* m_2 *not received*;
   (b) if $\mathsf{m}_2 \leftarrow \mathsf{V}_{\mathsf{succ}}{}^*$ is received, $\mathcal{U}$ has to verify the signature and data. If verifications are correct she continues the protocol. Otherwise, $\mathcal{U}$ can contact the TTP by calling *Claim* m_2 *not received*;
3. calculates $A_{\mathcal{U},i} \leftarrow PRNG(K) \oplus \mathsf{h}_{(r_{\mathcal{U}},n-i)}$, using the shared value K as seed;
4. sends $\mathsf{m}_3 = (\mathsf{T.Sn} \| A_{\mathcal{U},i})$ to $\mathcal{P}$;

verifyProof $\mathcal{P}$ follows the next steps:

1. if $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$ is not received, the *Claim* m_3 *not received* is called;
2. obtains T.Sn, and computes $\mathsf{h}_{(r_{\mathcal{U}},n-i)} \leftarrow A_{\mathcal{U},i} \oplus PRNG(K)$;
3. verifies $\mathsf{h}_{(r_{\mathcal{U}},n-j)} \stackrel{?}{=} hash^{s}(\mathsf{h}_{(r_{\mathcal{U}},n-i)})$;
4. if $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$ does not match, the *Claim* m_3 *not received* is called;

5. generates τ_2 and verifies it using the ticket expiry date $(\mathsf{T.Ti}, \mathsf{T.Tv})$ and the timestamp τ_1;
6. signs $A_{\mathcal{P},i}$ approving then the verification with timestamp τ_2: $\mathsf{R} \leftarrow (A_{\mathcal{P},i} \| \mathsf{T.Sn} \| \tau_2)$, and $\mathsf{R}^* \leftarrow (\mathsf{R} \| sig_{\mathcal{P}}(\mathsf{R}))$;
7. stores, updates its database, and unlocks the Sn from the database: $[\mathsf{R}^*, (\mathsf{h}_{(r_{\mathcal{U}},n-j)} \blacktriangleleft \mathsf{h}_{(r_{\mathcal{U}},n-i)}), (j \blacktriangleleft i)]$;
8. sends $\mathsf{m}_4 = \mathsf{R}^*$ to $\mathcal{U}$;

getValidationConfirmation $\mathcal{U}$ follows the next steps:

1. checks the signature of R^*;
2. computes $\mathsf{h}_{(r_{\mathcal{I}},n-i)} \leftarrow A_{\mathcal{P},i} \oplus PRNG(\mathsf{h}_K)$;
3. verifies $\mathsf{h}_{(r_{\mathcal{I}},n-j)} \stackrel{?}{=} hash^{i-j}(\mathsf{h}_{(r_{\mathcal{I}},n-i)})$;
4. if all verifications are correct, then stores and updates her database $[\mathsf{R}^*, (\mathsf{h}_{(r_{\mathcal{U}},n-j)} \blacktriangleleft \mathsf{h}_{(r_{\mathcal{U}},n-i)}), (j \blacktriangleleft i)]$; or else calls *Claim* m_4 *not received* to the TTP.

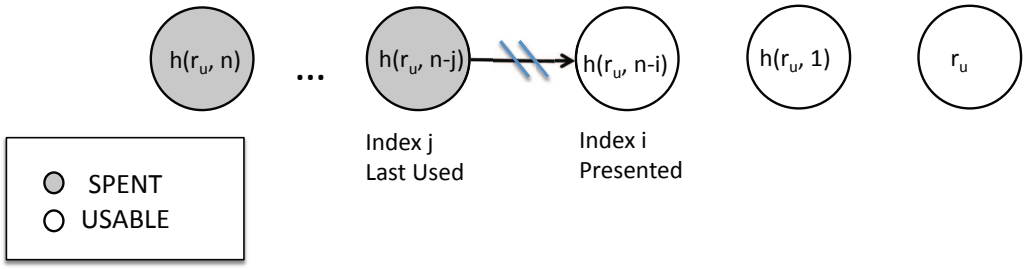

Figure 4.3: Correct use

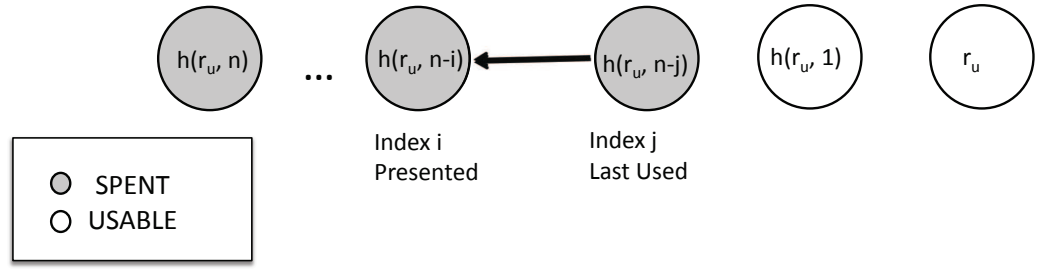

Figure 4.4: Reutilization

User ($\mathcal{U}$)	**Service Provider($\mathcal{P}$)**
Ticket Verification	
computes the remaining times $i \leftarrow j + s$	
$\mathsf{m}_1 = (\mathsf{T}^* \| i)$ ⟶	
	verifies ticket signature, T.Sv, T.Ti, and T.Tv
	if verification fails: abort
	looks up T^* in DB using T.Sn, then locks
	verifies ticket is not spent with $(j, \mathsf{h}_{(r_{\mathcal{U}}, n-j)})$ in the DB
	if no information found, then $j = 0$
	if $(i > j)\{$
	K and $r_{\mathcal{I}}$ are obtained from $\delta_{\mathcal{T},\mathcal{P}}$
	$\mathsf{h}_{(r_{\mathcal{I}}, n-i)} \leftarrow hash^{(n-i)}(r_{\mathcal{I}})$
	$A_{\mathcal{P},i} \leftarrow PRNG(\mathsf{h}_K) \oplus \mathsf{h}_{(r_{\mathcal{I}}, n-i)}$
	$enc_{pk_{\mathcal{T}}}(A_{\mathcal{P},i})$
	store $A_{\mathcal{P},i}$
	$\mathsf{V}_{\mathsf{succ}} \leftarrow (\mathsf{T.Sn} \| \mathtt{flag}_{\mathtt{valid}} \| \tau_1 \| enc_{pk_{\mathcal{T}}}(A_{\mathcal{P},i}) \| j)$
	$\mathsf{V}_{\mathsf{succ}}{}^* \leftarrow (\mathsf{V}_{\mathsf{succ}} \| sig_{\mathcal{P}}(\mathsf{V}_{\mathsf{succ}}))$
	$\mathsf{m}_2 = \mathsf{V}_{\mathsf{succ}}{}^*$
	$\}$**else**$\{$
	$\mathsf{h}_{(r_{\mathcal{U}}, n-i)} \leftarrow hash^{(j-i)}(\mathsf{h}_{(r_{\mathcal{U}}, n-j)})$
	$\mathsf{V}_{\mathsf{fail}} \leftarrow (\mathsf{T.Sn} \| \mathsf{h}_{(r_{\mathcal{U}}, n-i)} \| \mathtt{flag}_{\mathtt{spent}} \| i \| \tau_1)$
	$\mathsf{V}_{\mathsf{fail}}{}^* \leftarrow (\mathsf{V}_{\mathsf{fail}} \| sig_{\mathcal{P}}(\mathsf{V}_{\mathsf{fail}}))$
	$\mathsf{m}_2 = \mathsf{V}_{\mathsf{fail}}{}^*$
	$\}$
⟵ m_2	
verify signature of m_2	
if $\mathsf{V}_{\mathsf{succ}}{}^*$ or $\mathsf{V}_{\mathsf{fail}}{}^*$ not received, call *Claim* m_2 *Not Received*	
if $\mathsf{V}_{\mathsf{fail}}{}^*$ received: abort	
if $\mathsf{V}_{\mathsf{succ}}{}^*$ received, verify signature and data	
$A_{\mathcal{U},i} \leftarrow PRNG(K) \oplus \mathsf{h}_{(r_{\mathcal{U}}, n-i)}$	
$\mathsf{m}_3 = (\mathsf{T.Sn} \| A_{\mathcal{U},i})$ ⟶	
	if $\mathsf{h}_{(r_{\mathcal{U}}, n-i)}$ not received, call *Claim* m_3 *Not Received*
	$\mathsf{h}_{(r_{\mathcal{U}}, n-i)} \leftarrow A_{\mathcal{U},i} \oplus PRNG(K)$
	verify $\mathsf{h}_{(r_{\mathcal{U}}, n-j)} \stackrel{?}{=} hash^s(\mathsf{h}_{(r_{\mathcal{U}}, n-i)})$
	if $\mathsf{h}_{(r_{\mathcal{U}}, n-i)}$ does not match, call *Claim* m_3 *Not Received*
	generate τ_2
	verify time: $(\mathsf{T.Ti} \| \mathsf{T.Tv} \| \tau_2 \| \tau_1)$
	$\mathsf{R} \leftarrow (A_{\mathcal{P},i} \| \mathsf{T.Sn} \| \tau_2)$
	$\mathsf{R}^* \leftarrow (\mathsf{R} \| sig_{\mathcal{P}}(\mathsf{R}))$
	store, update DB, and unlock Sn: $[\mathsf{R}^*, (\mathsf{h}_{(r_{\mathcal{U}}, n-j)} \blacktriangleleft \mathsf{h}_{(r_{\mathcal{U}}, n-i)}), (j \blacktriangleleft i)]$ [2]
⟵ $\mathsf{m}_4 = \mathsf{R}^*$	
verify signature of R^*	
$\mathsf{h}_{(r_{\mathcal{I}}, n-i)} \leftarrow A_{\mathcal{P},i} \oplus PRNG(\mathsf{h}_K)$	
verify $\mathsf{h}_{(r_{\mathcal{I}}, n-j)} \stackrel{?}{=} hash^{i-j}(\mathsf{h}_{(r_{\mathcal{I}}, n-i)})$	
if verifications correct, store and update DB $[\mathsf{R}^*, (\mathsf{h}_{(r_{\mathcal{U}}, n-j)} \blacktriangleleft \mathsf{h}_{(r_{\mathcal{U}}, n-i)}), (j \blacktriangleleft i)]$	
otherwise call *Claim* m_4 *Not Received*	

Table 4.4: Ticket Verification subprotocol

The *Ticket verification* protocol is a fair exchange protocol with the existence of an offline TTP [Krem 02] between the user and the provider of the service (a valid e-ticket is given in exchange for the permission to use the service). This enables dispute resolution protocols in case of incorrect behaviour of the actors so as to preserve the security of the system. In case of dispute, they can contact the TTP following these protocols:

Claim m_2 not received

This protocol can be executed if $\mathcal{U}$ sends m_1 and says that she has not received $m_2 = V_{succ}{}^*$ from $\mathcal{P}$.

Claim User $\mathcal{U}$ executes:

1. sends the ticket $m_1 = (T^* \| i)$ to the TTP $\mathcal{T}$;

Response TTP $\mathcal{T}$ follows the next steps:

1. checks the information, signature and timestamp;
2. if the verification is correct, generates $(T.Sn \| \tau_3)$; then
3. signs the information $m_5 = ((T.Sn \| \tau_3) \| sig_{\mathcal{T}}(T.Sn \| \tau_3))$; and
4. sends m_5 to both $\mathcal{U}$ and $\mathcal{P}$. This entails acceptance of $\mathcal{U}$'s sent information and then $\mathcal{P}$ has the responsibility to unblock and send a correct m_2 to continue with the verification phase at sub-phase *verifyTicket*. After that, if the service cannot be finally guaranteed, $\mathcal{U}$ could demonstrate to a third party (by showing m_5) that $\mathcal{U}$ behaved correctly and $\mathcal{P}$ was the responsible of the denial of service;

verifyTicketWithTTP Service provider $\mathcal{P}$ executes:

1. executes *verifyTicket* normally;
2. sends m_2 to both $\mathcal{T}$ and $\mathcal{U}$, and continues the *Ticket verification* steps at point *showproofc*. The TTP has to store m_2 and m_5 because the user can go to an external dispute resolution system (if m_2 is still wrong) to solve the problem. In this case, the TTP will be able to provide these evidences.

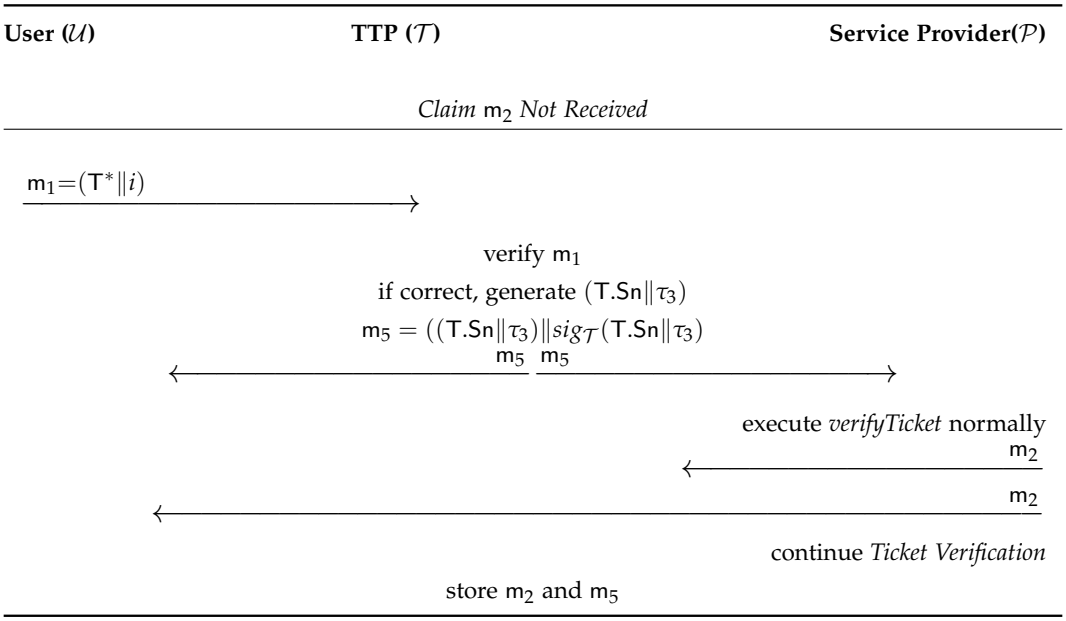

Table 4.5: Claim m_2 Not Received subprotocol

Claim m_3 not received

This protocol can be executed if $\mathcal{P}$ sends m_2 and says that has not received $\mathsf{m}_3 = A_{\mathcal{U},i}$ (with a correct $\mathsf{h}_{(r_\mathcal{U}, n-i)}$ inside) from $\mathcal{U}$.

Claim Provider $\mathcal{P}$ executes:

1. blocks the ticket $\mathsf{T.Sn}$ till the reception of m_3 from $\mathcal{U}$ or m_5 by $\mathcal{T}$;
2. another $\mathsf{T.Sn}'$ received from the same connection could not be accepted and $\mathsf{m}_2 = \mathsf{V_{succ}}^*$ would be repeatedly sent in order to unblock the ticket identified by $\mathsf{T.Sn}$.

Claim m_4 not received

This protocol can be executed if $\mathcal{U}$ sends m_3 and says that has not received $\mathsf{m}_4 = \mathsf{R}^*$ (with the contained $\mathsf{h}_{(r_\mathcal{I}, n-i)}$) from $\mathcal{P}$.

Claim User $\mathcal{U}$ follows the next steps:

1. sends to the TTP $\mathcal{T}$: $(\mathsf{m}_1 \| \mathsf{m}_2 \| \mathsf{m}_3) = (\mathsf{T}^* \| \mathsf{V_{succ}}^* \| (\mathsf{T.Sn} \| A_{\mathcal{U},i}))$;

Response TTP $\mathcal{T}$ executes:

1. verifies $(\mathsf{m}_1 \| \mathsf{m}_2 \| \mathsf{m}_3)$; if verification fails, it aborts the claim;
2. computes $A_{\mathcal{P},i} \leftarrow PRNG(\mathsf{h}_K) \oplus \mathsf{h}_{(r_\mathcal{I}, n-i)}$ using K and $r_\mathcal{I}$. Note that K and $r_\mathcal{I}$ can be obtained by decrypting $\delta_{\mathcal{T}\|\mathcal{P}}$ and then $\mathcal{P}$ can compute $\mathsf{h}_{(r_\mathcal{I}, n-i)} \leftarrow hash^{(n-i)}(r_\mathcal{I})$;
3. checks that $A_{\mathcal{P},i} \stackrel{?}{=} \mathsf{m}_2.\mathsf{V_{succ}}.A_{\mathcal{P},i}$;
4. verifies that $\mathsf{h}_{(r_\mathcal{I}, n-i)}$ matches with $\mathsf{T.h}_{(r_\mathcal{I}, n)}$;
5. checks that $\mathsf{m}_1.i > \mathsf{m}_2.\mathsf{V_{succ}}.j$
6. computes $\mathsf{h}_{(r_\mathcal{U}, n-i)} \leftarrow A_{\mathcal{U},i} \oplus PRNG(K)$ and verifies that $hash^i(\mathsf{h}_{(r_\mathcal{U}, n-i)}) \stackrel{?}{=} \mathsf{T.h}_{(r_\mathcal{U}, n)}$
7. if everything is successful, it then generates $(\mathsf{T.Sn} \| A_{\mathcal{P},i} \| A_{\mathcal{U},i} \| \tau_4)$; otherwise, it publishes the entity which misbehaved in accordance with the above verifications;
8. signs the information $\mathsf{m}_6 = ((\mathsf{T.Sn} \| A_{\mathcal{P},i} \| A_{\mathcal{U},i} \| \tau_4) \| sig_\mathcal{T}((\mathsf{T.Sn} \| A_{\mathcal{P},i} \| A_{\mathcal{U},i} \| \tau_4))$;
9. sends m_6 to $\mathcal{U}$.

Message m_6 can be used as an evidence in case of a user demand for the right to use the service in an external dispute resolution system.

User ($\mathcal{U}$)	**TTP ($\mathcal{T}$)**
Claim m_4 *Not Received*	
$(\mathsf{m}_1 \| \mathsf{m}_2 \| \mathsf{m}_3) = (\mathsf{T}^* \| \mathsf{V}_{\mathsf{succ}}{}^* \| (\mathsf{T.Sn} \| A_{\mathcal{U},i}))$ $\longrightarrow$	
	verify $(\mathsf{m}_1 \| \mathsf{m}_2 \| \mathsf{m}_3)$
	if correct, $\delta_{\mathcal{T},\mathcal{P}}$
	$\mathsf{h}_{(r_{\mathcal{I}},n-i)} \leftarrow hash^{(n-i)}(r_{\mathcal{T}})$
	$A_{\mathcal{P},i} \leftarrow PRNG(\mathsf{h}_K) \oplus \mathsf{h}_{(r_{\mathcal{I}},n-i)}$
	verify $A_{\mathcal{P},i} \stackrel{?}{=} \mathsf{m}_2.\mathsf{V}_{\mathsf{succ}}.A_{\mathcal{P},i}$
	verify $\mathsf{m}_1.i > \mathsf{m}_2.\mathsf{V}_{\mathsf{succ}}.j$
	$\mathsf{h}_{(r_{\mathcal{U}},n-i)} \leftarrow A_{\mathcal{U},i} \oplus PRNG(K)$
	verify $hash^i(\mathsf{h}_{(r_{\mathcal{U}},n-i)}) \stackrel{?}{=} \mathsf{T.h}_{(r_{\mathcal{U}},n)}$
	if verify ok, $(\mathsf{T.Sn} \| A_{\mathcal{P},i} \| A_{\mathcal{U},i} \| \tau_4)$
	$\mathsf{m}_6 = ((\mathsf{T.Sn} \| A_{\mathcal{P},i} \| A_{\mathcal{U},i} \| \tau_4) \| sig_{\mathcal{T}}(\mathsf{T.Sn} \| A_{\mathcal{P},i} \| A_{\mathcal{U},i} \| \tau_4))$
$\longleftarrow$ m_6	

Table 4.6: Claim m_4 Not Received subprotocol

4.1.3 The case of multiple providers

The described proposal states that only one provider is able to give a certain service, thus enabling offline verification. Nevertheless, this scenario could be extended to the existence of multiple providers that give a certain service. Then the same ticket could be accepted in different places, however guaranteeing the control of ticket overspending through online verification between all the providers. The encryption $enc_{sk_{\mathcal{P}}}$ would require a system to share data among the group of providers, enabling the access to K and $r_{\mathcal{I}}$. Special care about the distribution and control of used tickets should be taken (controlled by the existence of $r_{\mathcal{U}}$ in the database for that ticket). There should be a central database where all the providers could store all the used tickets, and then the verification would be online by imperative. In this scenario, the central server would only control the database, as the providers could be able to verify signatures and make all the cryptographic operations in order to perform all the critical real-time operations. This central database can be placed in the cloud; nonetheless, this can cause a delay that should be studied in detail in future work. Another option is to have all the databases actively connected one to each other, and achieve Atomic Broadcast as in [Pedo 00], in order to perform atomic operations to the databases (i.e. avoiding concurrent verifications using the same ticket in different providers). Expired tickets could be removed from the database for storage efficiency, and, moreover, only ticket serial numbers would have to be stored in the database instead of storing all the ticket information.

4.2 Security and privacy considerations

Proposition 4.1. *The proposed e-ticketing system preserves authenticity, non-repudiation, integrity and the expiry date of the e-ticket.*

Claim 4.1.1. *It is computationally unfeasible to make a new fraudulent e-ticket.*

Security Argument. A valid e-ticket has the form $\mathsf{T}^* = (\mathsf{T}, sig_{\mathcal{I}}(\mathsf{T}))$. Then, the first step that the provider $\mathcal{P}$ takes when an e-ticket is received is the verification of the signature. The *Ticket verification* protocol will continue only if this verification ends correctly; otherwise, $\mathcal{P}$ refuses $\mathcal{U}$'s request. Thus, making a new fraudulent valid

e-ticket would be equivalent to breaking the signature scheme and that would be computationally unfeasible as we have supposed that the issuer $\mathcal{I}$ uses a secure signature scheme.

Claim 4.1.2. *The issuer cannot deny the emission of a valid e-ticket.*

Security Argument. A valid e-ticket has $\mathcal{I}$'s signature and the signature scheme used is secure. Consequently, the identity of the issuer is associated to the ticket; the signature is a non-repudiation evidence of origin.

Claim 4.1.3. *The content of the e-ticket cannot be modified.*

Security Argument. Suppose that someone modifies the content of the ticket, then a new $\mathcal{I}$'s signature has to be generated over the modified content; otherwise, the e-ticket will not be valid. Again, if it is computationally unfeasible to forge the $\mathcal{I}$'s signature, it is unfeasible to modify the content of the e-ticket.

Claim 4.1.4. *The e-ticket will be no longer valid after the ticket validity time* $\mathsf{T.Tv}$.

Security Argument. The provider $\mathcal{P}$ receives the e-ticket from the user at the *Ticket verification* protocol before allowing access to the service. $\mathcal{P}$ first checks the correctness of the e-ticket (obviously that includes the verification of $\mathsf{T.Tv}$). If the verification is not correct, $\mathcal{P}$ stops the protocol and the user has no access to the service. Also, according to the *Claim 3*, the user cannot tamper $\mathsf{T.Tv}$.

Result 4.1. *According to the definitions given in §4.1 and the Claims 4.1.1, 4.1.2, 4.1.3 and 4.1.4, we can assure that the protocol achieves the properties specified in Proposition 4.1.*

Proposition 4.2. *The e-ticketing system described in §4.1 is anonymous. The service offered is revocable anonymous.*

Claim 4.2.1. *An e-ticket is anonymous.*

Security Argument. A valid e-ticket has the following information $\mathsf{T} = (\mathsf{Sn}\|\mathsf{Sv}\|Pseu_{\mathcal{U}}\|\mathsf{Tv}\|\mathsf{Ti}\|\mathsf{h}_{(r_{\mathcal{I}},n)}\|\mathsf{h}_{(r_{\mathcal{U}},n)}\|\delta_{\mathcal{T},\mathcal{P}}\|\ldots)$. The information related to the user's identity is solely $Pseu_{\mathcal{U}} = (y_{\mathcal{U}}\|sig_{\mathcal{P}}(hash(y_{\mathcal{U}})))$, where $y_{\mathcal{U}} = \alpha^{x_{\mathcal{U}}} \pmod p$. The user's identity is $x_{\mathcal{U}}$, thus an enemy has to solve the problem of computing the discrete logarithm to know the identity of the user. Currently no efficient algorithms are known to compute this mathematical problem.

Claim 4.2.2. *The purchase of an e-ticket is anonymous.*

Security Argument. As the protocol in §4.1.2 specifies, the channel between $\mathcal{U}$ and $\mathcal{I}$ of the ticket is anonymous. The protocol uses a Schnorr's ZKP to provide the user identity to the $\mathcal{I}$, so that the issuer can be sure that the connected user who wants to buy the ticket is the right holder of the pseudonym $Pseu_{\mathcal{U}}$ without disclosing her real identity. Thus, the user does not need to reveal her identity to buy an e-ticket.

Claim 4.2.3. *A fake user cannot buy an e-ticket impersonating other user.*

Security Argument. In order to buy a ticket, the user has to perform a Schnorr's ZKP to prove knowledge of the identity to the issuer without revealing it. The user has to compute w_1 such as $\alpha^{w_1} \stackrel{?}{=} A_1 \cdot {y_{\mathcal{U}}}^c \pmod{p}$. As far as any user preserves the privacy of her identity $x_{\mathcal{U}}$ (which links to $Cert_{\mathcal{U}}$ through the cooperation of $\mathcal{T}$), anyone else will not be able to compute such w_1. In this case, user can only be accused through $x_{\mathcal{U}}$ of ticket overspending (supposing that the user keeps $x_{\mathcal{U}}$ secretly), because she solely has the information to perform the *Ticket verification* protocol. Thus, the e-ticketing system also preserves the exculpability property.

Result 4.2. *According to the definitions given at §4.1 and the Claims 4.2.1, 4.2.2 and 4.2.3 we can assert the Proposition 4.2. The e-ticketing system is anonymous and this anonymity could be revocable in case of a user's fraudulent action. The pseudonym manager $\mathcal{T}$ knows the correspondence between $x_{\mathcal{U}}$ and $y_{\mathcal{U}}$ (see Algorithm: 'Pseudonym renewal'). Therefore, $\mathcal{T}$ could reveal the association between $x_{\mathcal{U}}$ and $y_{\mathcal{U}}$ due to law enforcement (e.g. a judge could request the user's identity to $\mathcal{T}$).*

Proposition 4.3. *The protocol satisfies the property of exculpability and a malicious service provider cannot reduce the times that a reusable ticket can be used.*

Claim 4.3.1. *The user $\mathcal{U}$ is able to prove that she has already validated the ticket.*

Security Argument. If a user $\mathcal{U}$ executes successfully the *Ticket verification* protocol, $\mathcal{U}$ will obtain the exculpability proof $r_{\mathcal{I}}$. She can use this proof to demonstrate that the ticket has been validated. If the *Ticket verification* protocol is stopped and $\mathcal{U}$ does not obtain the exculpability proof after the revelation of $r_{\mathcal{U}}$, she can execute *Claim* m_4 *not received*. This way $\mathcal{U}$ would obtain an alternative exculpability proof from the TTP.

Claim 4.3.2. *The service provider cannot falsely accuse the user of ticket overspending.*

Security Argument. When the service provider $\mathcal{P}$ receives the message m_1 in step 1 of the *Ticket verification* protocol (*showTicket*), $\mathcal{P}$ looks for the ticket that matches with the received serial number in its database. If the ticket has been already spent, the service provider will find the overspending proof $r_{\mathcal{U}}$ together with the ticket. The service provider has to show this element to accuse the user of overspending. If the user has not validated the ticket before, then the service provider does not have the element ($\mathcal{U}$ will send it in step 3: *showproofc*), as the inversion of the *hash* function is believed to be computationally infeasible, and collisions in this *hash* function can neither exist, so $\mathcal{P}$ cannot falsely accuse the user of overspending. If the service provider, even not being able to prove the overspending, decides to deny the service to the user, the user can contact the TTP in order to solve the situation through *Claim* m_2 *not received*.

Claim 4.3.3. *The provider $\mathcal{P}$ cannot falsely accuse the user of spending a coupon $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$, which has not already been used.*

Security Argument. The provider $\mathcal{P}$ cannot deduce any $\mathsf{h}_{(r_{\mathcal{U}},n-k)} \forall k < j$. When the user $\mathcal{U}$ spends the ith coupon of her ticket, she sends $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$ to the provider $\mathcal{P}$ (see step *shoowproofc* of the *Ticket verification* protocol). Then the provider stores this value in order to avoid overspending. According to the hash functions properties and, as it shows in Figures 4.3 and 4.4, from $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$ it is only possible to deduct an $\mathsf{h}_{(r_{\mathcal{U}},n-k)} \forall k > j$, since it is not possible to go in the direction of $r_{\mathcal{U}}$. So, the provider is not able to deduce any non-spent value of the hash function chain.

Result 4.3. *According to the definitions given in §4.1 and the Claims 4.3.1, 4.3.2 and 4.3.3, we can assure that the protocol achieves the property specified in the Proposition 4.3. The ticket verification process is a fair exchange: any part can obtain the exculpability proof of the other part without revealing its own proof.*

Proposition 4.4. *The tickets issued by the protocol described in §4.1.2 can be preset to be reusable tickets, both for a limited number of verifications or a limited period of time.*

Claim 4.4.1. *The protocol allows the creation of N-usable tickets maintaining the security properties of the non reusable tickets, including exculpability.*

Security Argument. During the execution of the *Ticket verification* protocol, $\mathcal{U}$ uses the last element of the chain of proofs $\mathsf{h}_{(r_{\mathcal{U}},n)}$ and receives an element containing the last element of the chain of issuer proofs $\mathsf{h}_{(r_{\mathcal{I}},n)}$ in exchange. Due to the properties

of hash functions, $\mathcal{U}$ cannot generate $\mathsf{h}_{(r_{\mathcal{I}},n-i)}$ and $\mathcal{P}$ cannot generate $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$. The successive verifications will use the remaining elements of the chain in the reverse order.

Claim 4.4.2. *The protocol allows the creation of period-usable tickets maintaining the security properties of the non-reusable tickets, including exculpability.*

Security Argument. In this case, the concept of overspending is not applicable. The user will obtain a verification proof each time she executes the *Ticket verification* protocol, obtaining an exculpability proof providing that the time of the verification attempt is less than the limit of the validity period.

Result 4.4. *According to Claims 4.4.1 and 4.4.2, we can assert Proposition 4.4. The protocol is flexible enough to be used with all kinds of services, with independence from its reusability requirements.*

Proposition 4.5. *The protocol avoids overspending with minimum requirements of persistent connections with a centralized system.*

Claim 4.5.1. *The protocol avoids overspending.*

Security Argument. If a user tries to overspend a ticket, she will send to $\mathcal{P}$ a spent hash value (a $\mathsf{h}_{(r_{\mathcal{U}},n-i)}$ with $i \leq j$). The provider will verify that $\mathsf{h}_{(r_{\mathcal{U}},n-j)} \stackrel{?}{=} hash^{s}(\mathsf{h}_{(r_{\mathcal{U}},n-i)})$. This verification will always fail, because the hash chain goes in the opposite direction (see Figures 4.3 and 4.4). Then the provider will block the ticket identified by $\mathsf{T.Sn}$ until the reception of a non-spent hash value.

Claim 4.5.2. *If the ticket can only be validated by one provider the verification is offline.*

Security Argument. Provider $\mathcal{P}$ maintains a database with the serial numbers of the e-tickets that have been already validated (together with their exculpability proofs) until their expiry date. With the contents of this database the provider has enough information to decide if $\mathcal{P}$ accepts and validates a new ticket, because $\mathcal{P}$ can check both the issuer's signature and the fact that the e-ticket has not been spent before. So the provider does not need to contact to any party during the verification of an e-ticket.

Claim 4.5.3. *If the ticket can be validated with several providers, the providers must then be connected and share a database of spent tickets.*

Security Argument. The set of providers maintain a shared database with the serial numbers of the e-tickets that have been already validated (together with their exculpability proofs) until their expiry date. The contents of this database are used by the providers to decide if they accept and validate a new ticket. So the provider does not need connection to the issuer during the verification of an e-ticket, but the set of providers must have a shared database instead.

Result 4.5. *According to the definitions given in §4.1 and the Claims 4.5.1, 4.5.2 and 4.5.3, we can assure that the protocol achieves Proposition 4.5. The issuer is offline during the verification phase and the providers contact each other only in some kind of services. In all cases, the protocol prevents ticket overspending.*

4.3 Implementation details and results

There are several important factors to consider when we design an e-ticketing system that should be usable in practice. The response time is one of them. Thus, we have implemented our protocol and we present some results regarding its time performance.

In §4.3.1, we describe the developed components, the development environment and the hardware that we have used. Next, the testing methodology is described in §4.3.2, i.e. the system can be configured using different key lengths. We can assume that we would obtain more security with larger keys but the computational cost would be higher. We want to study how the key length influences the computational cost. Next, we present the obtained results differentiating the costs in the user side (§4.3.3) and the server side (§4.3.4). Finally, in §4.3.5, the required database size is stated for each key length case.

4.3.1 E-ticketing system configuration and experimental details

As introduced in §4.1, our system comprises three main phases: pseudonym renewal, ticket purchase and ticket verification, and four participants: the user, the service provider, the ticket issuer and the pseudonym manager. Therefore, the system implementation requires four components: one for each entity in the system. Nonetheless, we have grouped the service provider, the ticket issuer and the pseudonym manager in one server for practical reasons, see Figure 4.5. The server takes the role of different servers in a PC. The server component has been de-

veloped with the Java programming language (Java 2 Standard Edition), allowing portability in a great number of platforms.

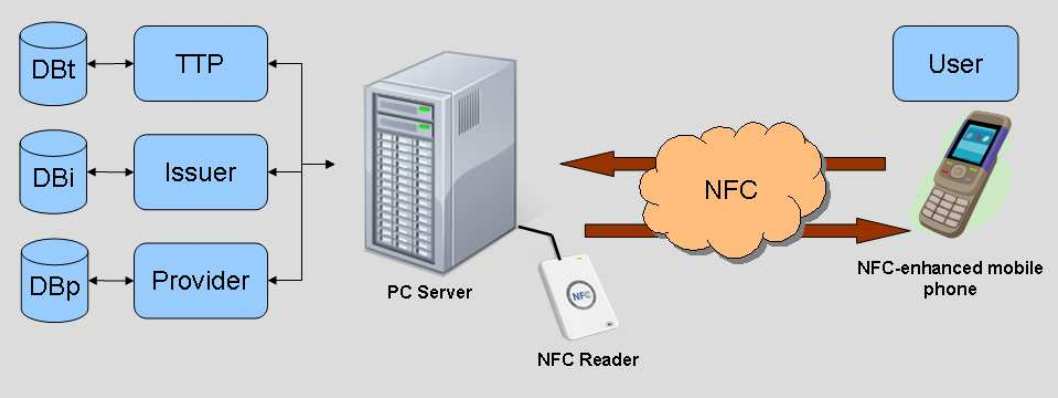

Figure 4.5: Architecture of the testing environment

The user interacts with the other participants (the service provider, the ticket issuer and the pseudonym manager) by means of a mobile phone, so that the user component (client) should be executed in a mobile phone. Given that a great number of mobile phones can execute Java applications, we have developed the client in Java 2 Micro Edition (J2ME). The mobile phone that has been used is a Nokia 6212 Classic with an embedded API for NFC communication.

The communication between server and client is performed via Near Field Communication (NFCIP-1, ISO18092). The server uses an Arygon NFC Reader (ADRA-USB) in order to connect with the mobile phone. The equipment of the entire scheme is detailed in Table 4.7.

It should be taken into consideration that the mobile phone acts as the initiator of the transactions, and the server is the target, i.e. the server is waiting for $\mathcal{U}$'s requests.

Finally, we have used the BouncyCastle crypto library [3] for all the cryptographic operations in both J2SE and J2ME.

4.3.2 Testing methodology

The e-ticketing system can be configured with the key length parameter l. This parameter l refers to the key size (in bits) of the RSA cryptosystem used in the pro-

[3] http://www.bouncycastle.org/

Table 4.7: Equipment specification details

Computer(server)	CPU	AMD Athlon 64 X2 Dual 5000+ (2.59GHz)
	RAM	2 GB
	OS	Windows XP
	Java version	Java 6
	NFC reader	Arygon ADRA-USB
Mobile phone(client)	Model	Nokia 6212 NFC classic
	Java version	J2ME (Series 40 SDK 1.0 with JSR 257 extension)

tocol, as well as the number of bits of the generated prime numbers for the generation of the $\mathbb{Z}_q$ and $\mathbb{Z}_p$. The larger the parameter is, the harder the cryptosystem is, so we have a more secure system. On the other hand, the time consumption is also increased, and has to be evaluated.

We have run the protocols with different key sizes of 512, 1024 and 2048 bits, respectively. The results are studied in §4.3.3 and §4.3.4, evaluating the costs in the user side and the server side, respectively. Regarding the length of the keys l, at the present time a size of $l = 1024$ bits is considered computationally safe [Stan 07]. According to that, we have tested our scheme with a smaller length ($l = 512$ bits) and a larger one ($l = 2048$ bits). In this way, we can examine how the key length influences the system performance.

We have executed several test for every key length and protocol, so that the times shown in the following sections are the average of these times.

4.3.3 Experimental results in the client side

We have studied the global times of the protocol as well as the partial times of each protocol (pseudonym renewal, ticket purchase and ticket verification), in order to identify the most costly parts of each protocol in the client (user) side.

Global time performance results

Figure 4.6 shows the average time (in ms) required to complete each transaction (*Pseudonym renewal*, *Ticket purchase* and *Ticket verification* phases) taking into ac-

count the interaction with the other entities. These results are given depending on the used key length l (in bits) with its values 512, 1024 and 2048, respectively. We especially focus on the *Ticket verification* phase, where the delay time has to be strongly reduced if we assume a mass-transit scenario. This delay varies from 1.1 to 2.5 seconds depending on the key length l parameter, what makes the proposal definitely practical. In general, all the transactions are considered practical in 1024 bits (cost lower than 2s), and they become increased in 2048 bits. We detail the times of each phase by considering the costs of all their subphases in order to show the most costly operations in terms of delay times.

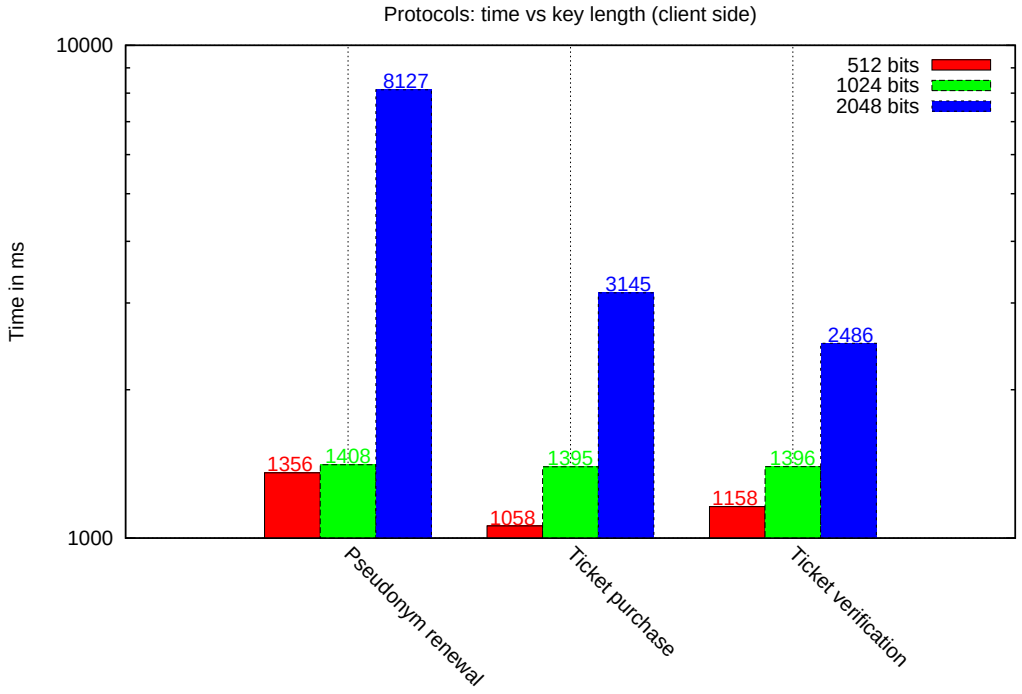

Figure 4.6: Computational performance of every protocol using several key lengths in the client side

Detailed time performance of the pseudonym renewal

Figure 4.7 shows the partial time intervals of the *Pseudonym renewal* phase. As expected, the decryption of the signed pseudonym (t_3) is the most costly operation in this phase, and increases obviously depending on the key length l parameter. This operation is performed by the user in the mobile phone. There are not great remarks in the other operations, as precomputation of the non-interactive values helps to reduce the global transaction times.

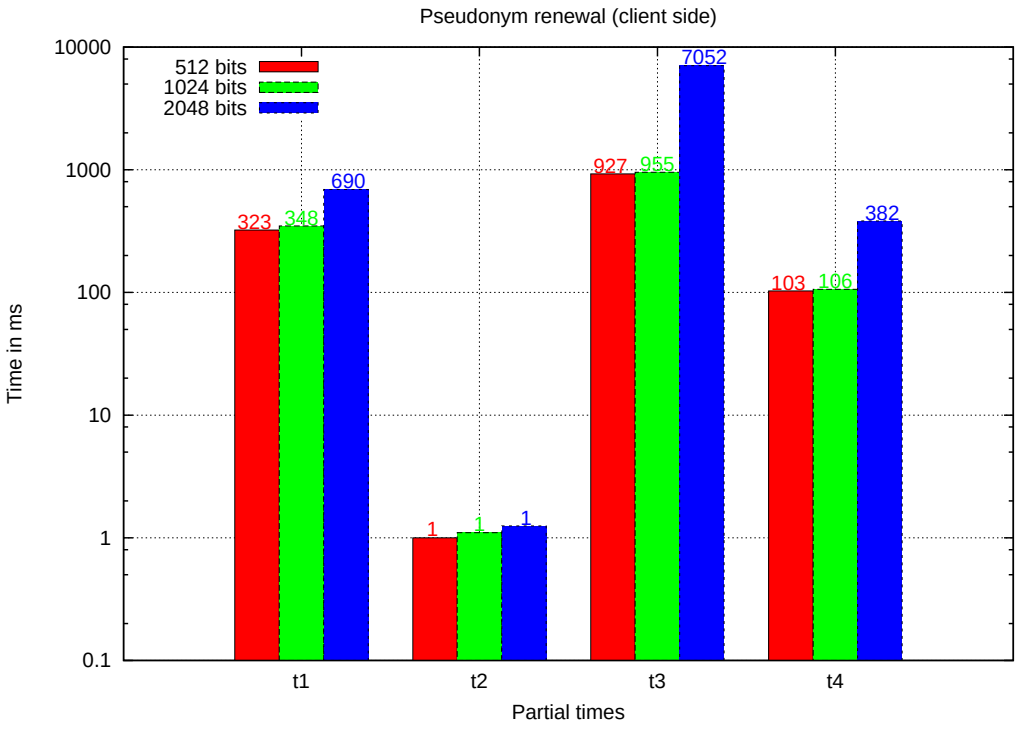

Figure 4.7: Partial times of the pseudonym renewal (see Table 4.8 for t_i details)

Table 4.8: Details of the pseudonym renewal partial times

Partial time	Description
t_1	Sending of the pseudonym request
t_2	Reception of the pseudonym response
t_3	Decryption of the signed pseudonym
t_4	Pseudonym's signature verification

We have obtained similar results for 512 and 1024, where the variation between them is few milliseconds; when we use the 2048-bit key, the computational times are higher than we expected. This is due to the actual computational power, i.e. the times required to compute modular exponentiations with 512 and 1024 bits are quite low and they are practically the same. In this case (512 and 1024), the communication costs can have more influence in the final time than the computational cost.

Detailed time performance of the ticket purchase

Figure 4.8 shows the partial time intervals of the *Ticket purchase* phase. The main costs remain on the computation and transmission of the Schnorr's ZKP (t_3 and t_4), as well as the communication cost of the first commitment (t_1), especially those with 2048 bits. The verification of the ticket signature (t_7) varies from 100 to 400 ms. Once again, some values have been precomputed to reduce the time of the protocol execution.

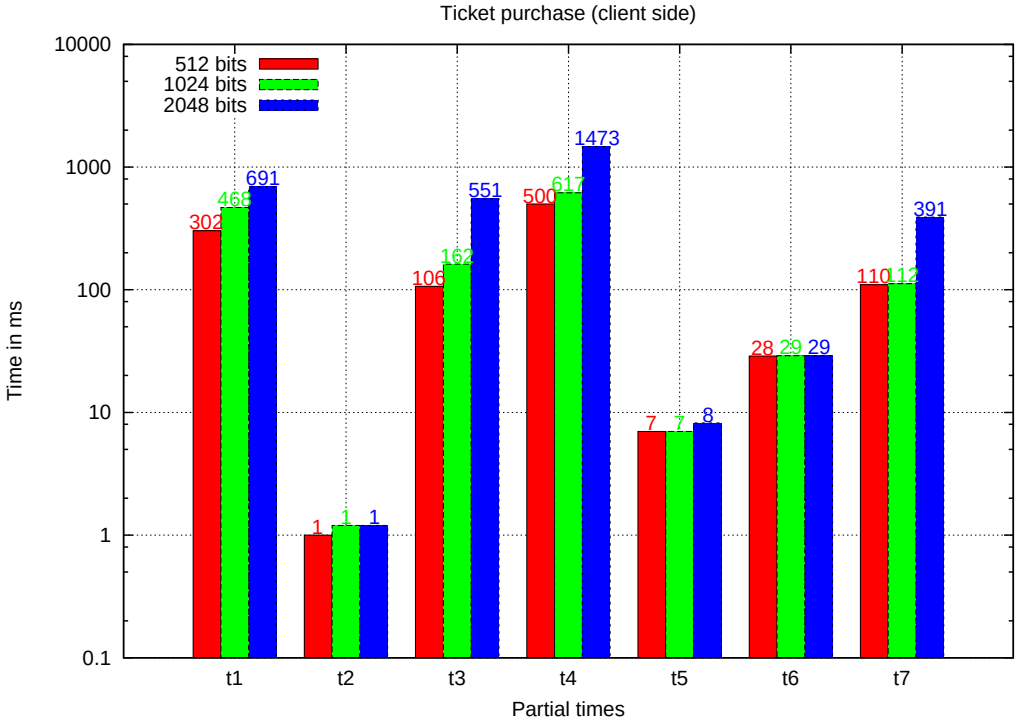

Figure 4.8: Partial times of the ticket purchase (see Table 4.9 for t_i details)

Table 4.9: Details of the ticket purchase partial times

Partial time	Description
t_1	Sending of the commitment
t_2	Reception of the challenge
t_3	Computation of the Schnorr's ZKP response
t_4	Sending of the Schnorr's ZKP response
t_5	Computation of the shared symmetric key
t_6	Reception of the ticket
t_7	Verification of the ticket data

Detailed time performance of the ticket verification

Figure 4.9 shows the partial time intervals of the *Ticket verification* phase. The most remarkable costs remain on the connection and sending of the ticket (t_1),

depending on the amount of data with its key length (parameters and signature), followed by the signature verification of the response (t_3), the sending of the symmetric encryption of the parameter $r_{\mathcal{U}}$ (t_5), and finally the verification of the receipt (t_7). Other operations such as the reception of the response (t_2), the computation of the symmetric encryption of $r_{\mathcal{U}}$ (t_4), the reception of the receipt (t_6) and the computation of the symmetric decryption of $r_{\mathcal{I}}$ (t_8) have not become costly.

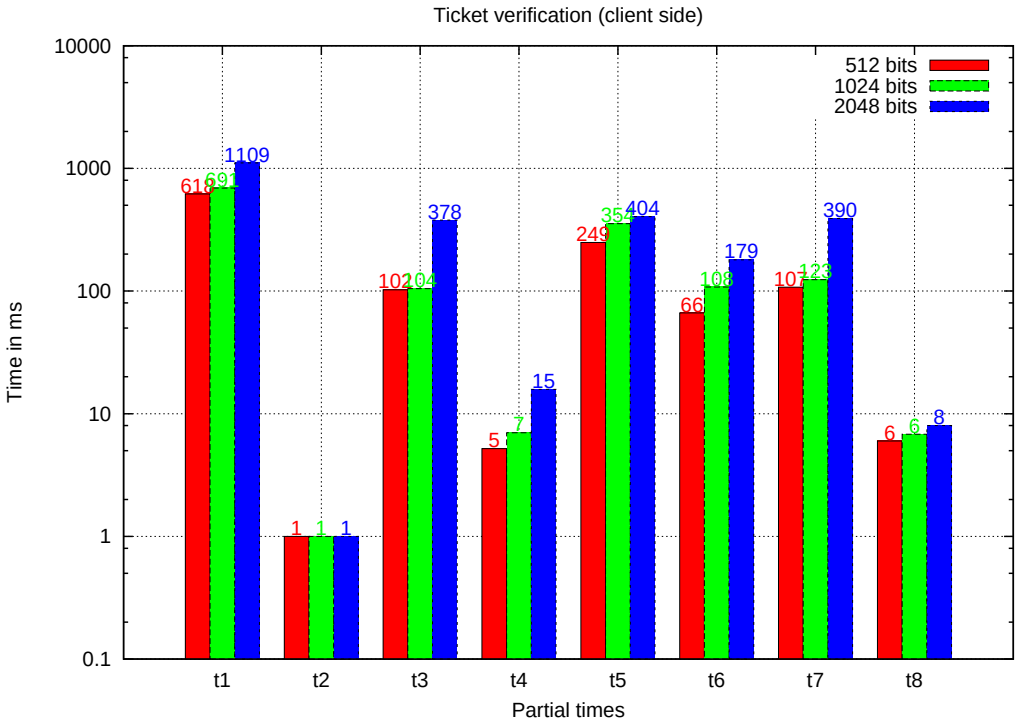

Figure 4.9: Partial times of the ticket verification (see Table 4.10 for t_i details)

Table 4.10: Details of the ticket verification partial times

Partial time	Description
t_1	Sending of the ticket
t_2	Reception of the response
t_3	Verification of the response
t_4	Computation of the symmetric encryption of $r_{\mathcal{U}}$
t_5	Sending of the symmetric encryption of $r_{\mathcal{U}}$
t_6	Reception of the receipt
t_7	Verification of the receipt data
t_8	Computation and verification of the symmetric decryption of $r_{\mathcal{I}}$

Independently from the key length l parameter, there is a variation in the communication times depending on the steps of the protocol, as the server and the client have to synchronize their protocol steps in order to exchange their information.

4.3.4 Performance results in the server side

We have studied the global times of the protocol as well as the partial times of each protocol (*Pseudonym renewal*, *Ticket purchase* and *Ticket verification* phases), in order to identify the most costly parts of each protocol in the server side.

Global time performance results

Figure 4.10 shows the average time (in ms) required to complete each transaction (*Pseudonym renewal*, *Ticket purchase* and *Ticket verification* phases) taking into account the interaction with the user by each entity (Trusted Third Party, Issuer and Service Provider). These results are given depending on the used key length l (in bits) with its values 512, 1024 and 2048, respectively. We focus especially on the *Ticket verification* phase, where the delay time has to be strongly reduced if we assume a mass-transit scenario. This delay varies from 0.7 to 2 seconds depending on the key length l parameter, what makes the proposal definitely practical, especially until 1024 bits. We detail the times of each phase by considering the costs of

all their subphases in order to show the most costly operations in terms of delay times.

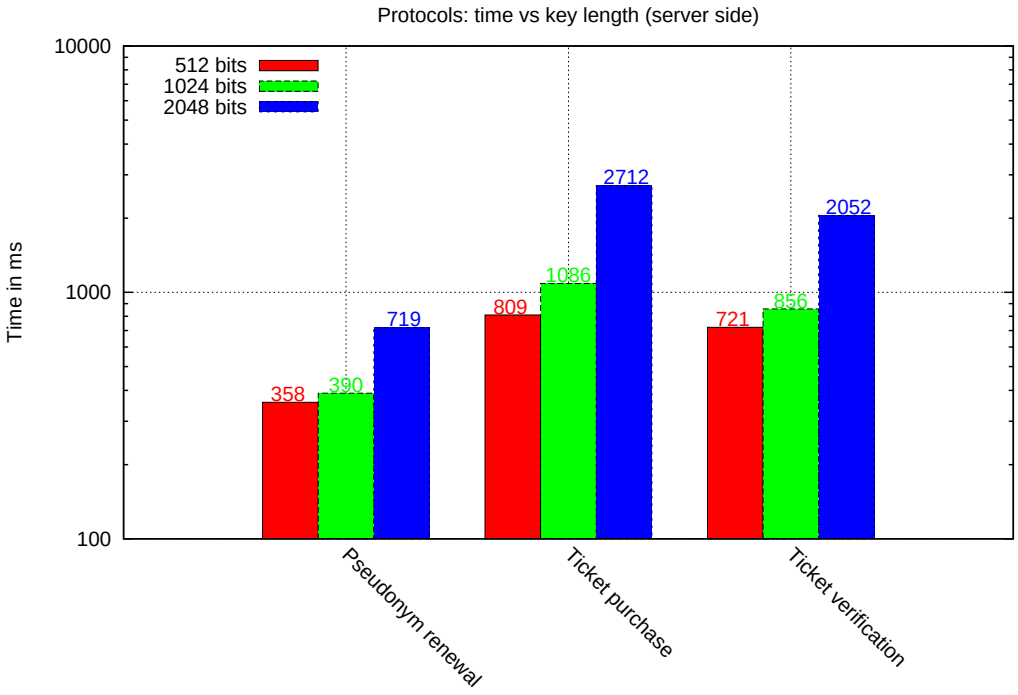

Figure 4.10: Computational cost of every protocol using several key lengths in the server side

Detailed time performance of the pseudonym renewal

Figure 4.11 shows the partial time intervals of the *Pseudonym renewal* phase. In this part, the communication with the client (reception of the request and sending of the signed pseudonym) is the major part of the protocol.

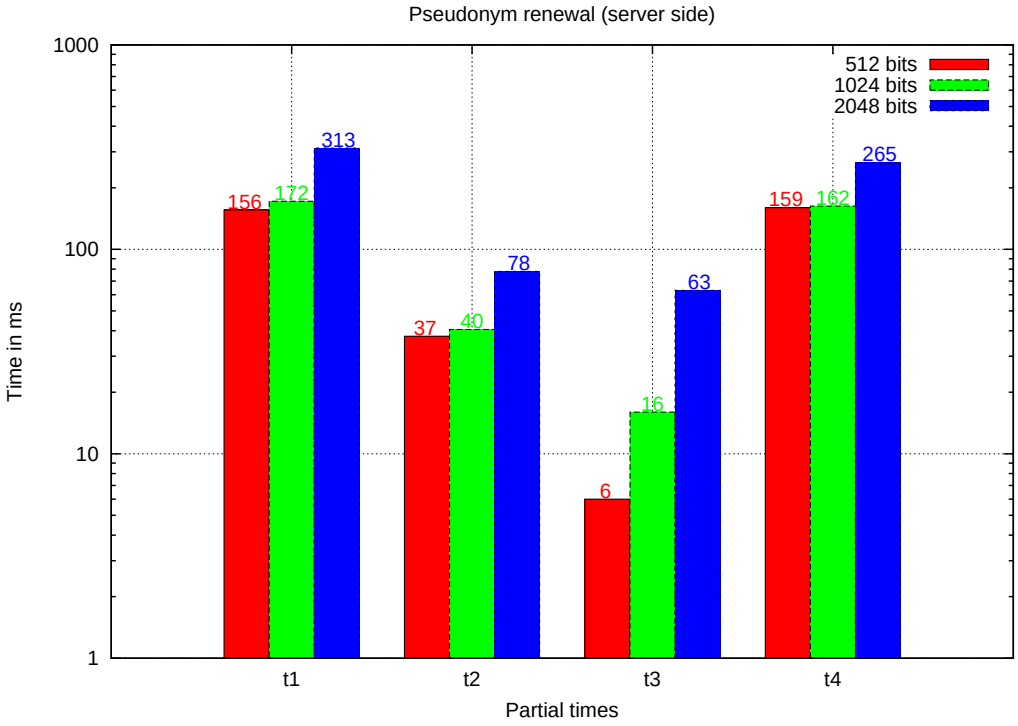

Figure 4.11: Partial times of the pseudonym renewal (see Table 4.11 for t_i details)

Table 4.11: Details of the pseudonym renewal partial times

Partial time	Description
t_1	Reception of the pseudonym request
t_2	Pseudonym extraction
t_3	Verification & signature of the pseudonym
t_4	Sending the signed pseudonym

Detailed time performance of the ticket purchase

Figure 4.12 shows the partial time intervals of the *Ticket purchase* phase. The main costs are, again, for communication (and synchronization) with the client (t_1, t_4, t_7), and only the verification of the Zero-Knowledge Proof is the most costly computation part (mainly for 2048 bits).

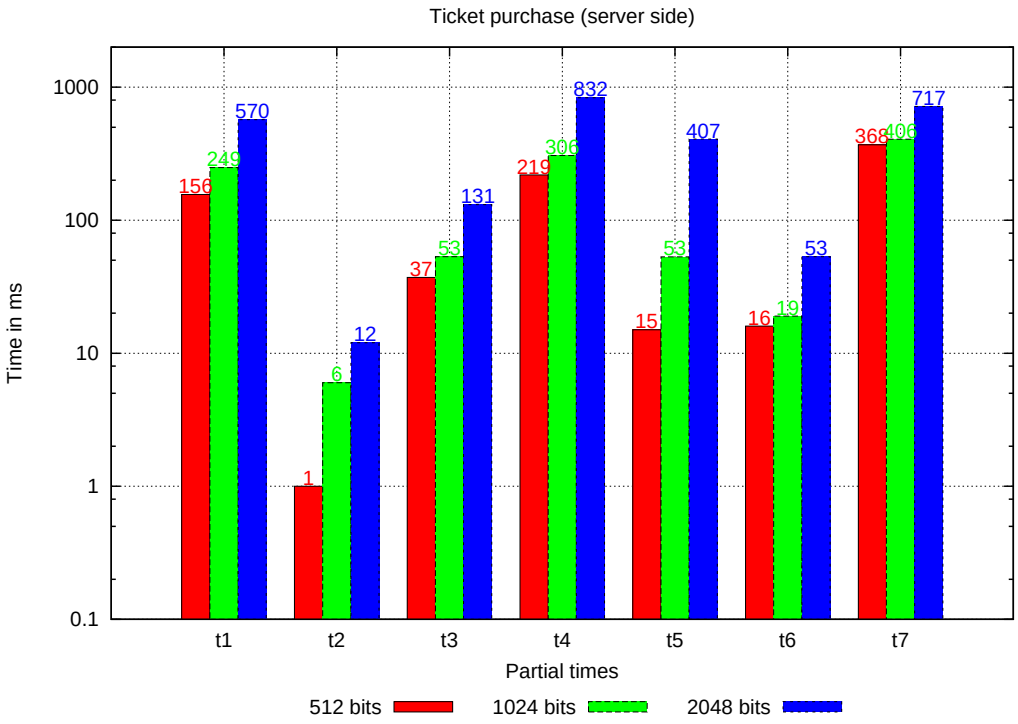

Figure 4.12: Partial times of the ticket purchase (see Table 4.12 for t_i details)

Table 4.12: Details of the ticket purchase partial times

Partial time	Description
t_1	Reception of the commitment
t_2	Signature verification
t_3	Computation & sending of the challenge
t_4	Reception of the ZKP response
t_5	Verification of the ZKP response
t_6	Generation of the ticket
t_7	Sending of the ticket

Detailed time performance of the ticket verification

Figure 4.13 shows the partial time intervals of the *Ticket verification* phase. The main costs are also related to communication (t_1 for sending of the ticket, t_5 for sending the symmetric encryption of $r_{\mathcal{U}}$), but there are also some computation costs to be taken into account (t_3 verification of the response, and t_7 verification of the receipt data), specially at 2048 bits.

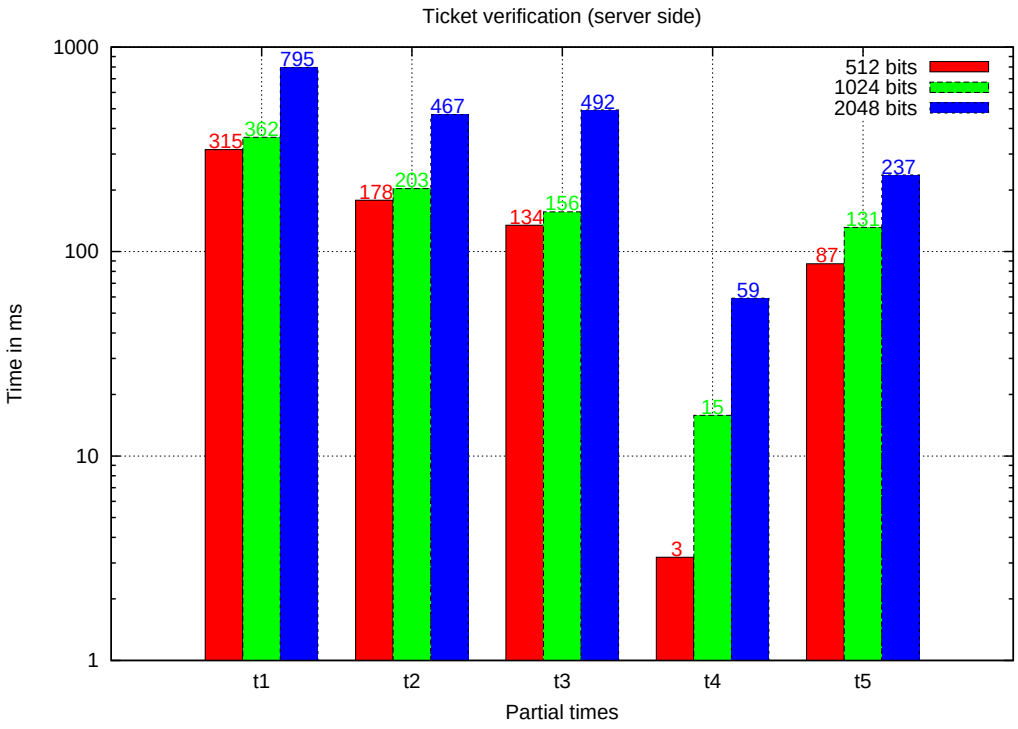

Figure 4.13: Partial times of the ticket verification (see Table 4.13 for t_i details)

Table 4.13: Details of the ticket verification partial times

Partial time	Description
t_1	Reception of the ticket
t_2	Generation of the response
t_3	Reception of the symmetric encryption of $r_{\mathcal{U}}$
t_4	Generation of the receipt
t_5	Sending of the receipt

4.3.5 Database size and other system requirements

Table 4.14 shows the size of every register in the database. This size depends on the key length parameter l: 512, 1024 or 2048 bits. In the table, we also detail the attributes which are stored into the database, and also their partial sizes.

Table 4.14: Details of the database register sizes (in bytes) depending on the key length parameter l

Parameter	Key size		
	512 bits	1024 bits	2048 bits
T^*	870 B	995 B	1765 B
R^*	218 B	281 B	538 B
$r_{\mathcal{I}}$	64 B	128 B	256 B
$r_{\mathcal{U}}$ or $h_{(r_{\mathcal{U}},use)}$	64 B	128 B	256 B
use	1 B	1 B	1 B
TOTAL	1217 B	1533 B	2816 B

We analyse the capacity requirements of this protocol for a real mass-transport system. The Tokyo Subway is the metro system which has most annual passenger rides. In 2009 registered 3160 M rides, that is an average of 8.7M daily rides [4]. If we take our 1024-bit results (1533 B per register), a new daily capacity of 12.43 GiB would be required. According to what we suggested for the maintenance of the database, the tickets have a validity time, and after that period they could be

[4] http://geography.about.com/od/urbaneconomicgeography/a/Busiest-Subways.htm

removed. If we set, for example, a 60-day validity period, a capacity of 750 GiB would be required, and it could therefore be usable in this kind of mass-transport system.

4.4 Conclusions and related publications

We have presented an e-ticketing scheme that includes exculpability and reusability as security requirements. Moreover, we have shown the experimental results in order to prove its usability in mobile devices for users.

A first version of the protocol that included exculpability was presented in an international conference in Athens (Greece), 2010, with its proceedings published in 2011. The protocol was then improved and extended with a new security requirement to be included, reusability, also performing the experimental results on a real scenario with mobile devices, and published in an ISI-JCR Journal in 2012. We show the publications as follows:

- A. Vives-Guasch, M.M. Payeras-Capellà, M. Mut-Puigserver, and J. Castellà-Roca. "E-ticketing Scheme for mobile devices with exculpability". In *Data Privacy Management and Autonomous Spontaneous Security (DPM), 5th International Workshop, LNCS 6514, pp. 79–92, doi: 10.1007/978-3-642-19348-4_7*, 2011.

- A. Vives-Guasch, M.M. Payeras-Capellà, M. Mut-Puigserver, J. Castellà-Roca, and J.L. Ferrer-Gomila. "A secure e-ticketing scheme for mobile devices with Near Field Communication (NFC) that includes exculpability and reusability". *IEICE Transactions on Information and Systems, Vol.E95-D No.1, pp. 78–93, doi: 10.1587/transinf.E95.D.78*, 2012.

Chapter 5

Secure Automatic Fare Collection system with Short-Term Linkability

This chapter introduces the contribution to Automatic Fare Collection (AFC) systems, focused especially on transportation systems.

Contents

Section §5.1 first details the security requirements to be fulfilled. Section §5.2 describes the solution for time-based fare calculation, and section §5.3 presents the adaptation for distance-based fare calculation. Section §5.4 details the security and privacy considerations, and the implementation details and results in §5.5. Section §5.6 states the conclusions and related publications.

5.1 Requirements of the fare collection systems

We first define the security requirements that are common in our scenario. We then detail them depending on time- or distance-based protocol version.

5.1.1 Common security requirements

Transport services give a receipt or a ticket to users in order to be further verified; then, this receipt is a proof that the protocol was followed correctly. In these electronic systems, the following security requirements have to be guaranteed:

- Authenticity (Def. 2.1). A ticket is authentic when any party can verify that the e-ticket information has been generated by its legitimate issuer.
- Integrity (Def. 2.2). An electronic ticket cannot be modified without being detected by any party.
- Non-repudiation (Def. 2.3). The requirement of non repudiation comprehends the fulfillment of the non repudiation of origin and receipt. This means that neither the issuer nor the receiver of an e-ticket can deny its emission and its reception.
- Non-overspending (Def. 2.5). E-tickets can only be used as agreed in the contract between the issuer and the user.
- Revocable anonymity (Def. 2.7). Anonymity of users has to be guaranteed, but it could be revoked in case of misbehaviour.
- Short-term linkability (Def. 2.9, 2.10). Several tickets from the same user cannot be attributed to this determined user. In this scenario, the provider

can only trace an entrance of a user with its corresponding exit, but it can never trace different journeys of a same user, which could enable generation of profiles.

- Expiry date (Def. 2.16). A ticket could be valid only during a determined time interval. It is also called as validity time.

5.1.2 Requirements for time-based systems

Time-based fares are most appropriate in environments where the most relevant parameter of the service given is the time. Some good examples of that applied to the transport systems are: taxi services and parking places services. Thus, time-based pricing approaches will require time accounts rather than paying for boarding structures. So, in this case, the system has to:

- Create a proper timestamp when a new ticket is issued.
- This timestamp creates a time-window where the user has the right to use the service.
- The time-window has an initial-date and a expiry-date which determine the maximum period of the service.
- The fare to be paid is proportional to the period of time that the costumer has used the service. The longer the period is, the higher the fare will be.
- The timestamp must be checked at the system exit in order to compute the service fare.

5.1.3 Requirements for distance-based systems

Distance-based systems calculate the fare to be paid as a function of the entrance and the exit point. The system has to:

- Include the identifier of the entrance station in the entrance ticket.
- Define a validity period for each ticket.
- Make a correlation between the distance covered by the user and the fare to be paid. The larger the distance, the higher the fare.

In distance-based systems, beside the entrance point, the entrance ticket must include a new item, which is the direction of the journey of the user. If the service checks the direction included in the entrance ticket, confaculated aatacks against a user who exits the destination station can be prevented if the system fulfills the following requirements:

- At the entrance station, users must obtain the entrance ticket at different points according to their direction, that is, entrances are separated by direction.
- Users are not able to change the direction of the movement without exiting the service. When a user arrives at the destination station, she must check out according to her direction. That is, system exits are separated by direction, as depicted in Figure 5.1.

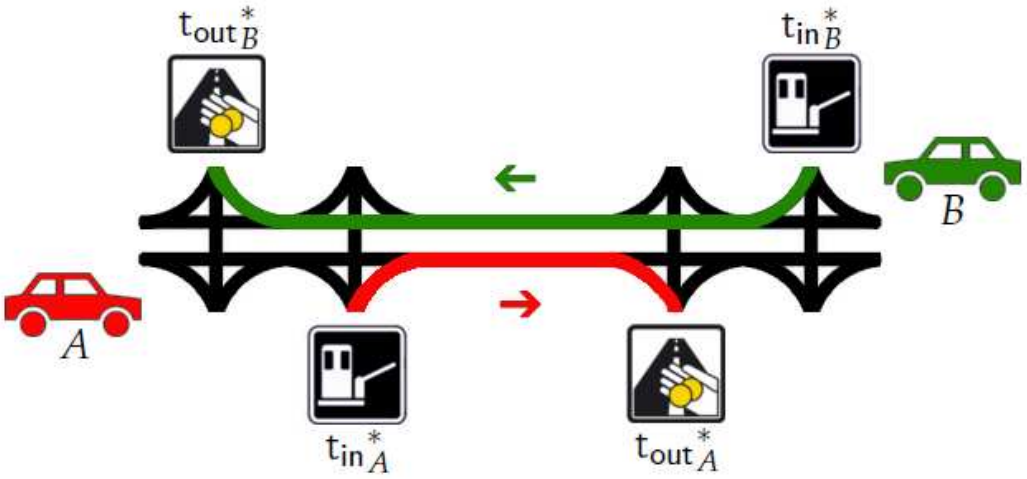

Figure 5.1: Separated directions in a compliant distance-based AFC.

If the service is non-compliant with the previous requirements, then a more complex protocol will be required to solve the emerged problems due to the possibility of confabulated attacks. The solution for compliant services is described in §5.3.1, while the solution for non-compliant services is included in §5.3.4.

5.2 Time-based fare collection protocol

In this section, we describe our Time-Based Fare Collection system which provides anonymity to the users by the use of group signatures [Bone 04b] for mass-transport services. First, we present our modifications to the BBS scheme in order to guarantee short-term linkability. Then, we describe the parties involved in the system, the security requirements to be guaranteed, the information which is contained in the entrance and exit tickets, and finally the phases in which the system consists of.

5.2.1 Short-term linkability

We present an extension of the original background defined in §3, a self-adaptation for achieving short-term linkability. A user can thus decide whether to perform a signature linkable to the previous generated one or not. We add some procedures to be used in the proposal, and also a Zero-Knowledge Proof verification over the group signature scheme in order to achieve active verification for their signers.

Linkable group signatures

We define two new procedures based on the original BBS scheme [Bone 04b] in order to construct linkable group signatures. $SignLinkable_G$ and $VerifyLinkable_G$ are novel and we have defined them as an extension of the original procedures.

SignLinkable*$_G$*(gpk, gsk[i], M) Given a group public key gpk, a private user's key $gsk[i]$ and a message M, compute and output a signature σ. In order to use this procedure correctly, it is defined as follows:

- First time: use standard $Sign_G(gpk, gsk[i], M)$:
  1. generate a linear encryption of A: $(T_1, T_2, T_3) \leftarrow (u^\alpha, v^\beta, Ah^{\alpha+\beta})$ for $\alpha, \beta \overset{R}{\leftarrow} \mathbb{Z}_p$;
  2. compute the helper values $\delta \leftarrow x\alpha$ and $\mu \leftarrow x\beta$;
  3. select $r_\alpha, r_\beta, r_x, r_\delta, r_\mu \overset{R}{\leftarrow} \mathbb{Z}_p$;
  4. compute the values R_1, R_2, R_3, R_4, R_5. For computational simplicity, note that 2 out of the 3 pairings which are needed to calculate R_3, can be already precomputed in the setup, namely $p_2 = e(h, w)$ and

$p_3 = e(h, g_2)$, as their value is computed from the group public parameters. Then, only the first pairing $p_1 = e(T_3, g_2)$ needs to be computed when signing. This is a note from a computational point of view, so we maintain the original notation.

$$\begin{aligned} R_1 &\leftarrow u^{r_\alpha}, & R_2 &\leftarrow v^{r_\beta}, \\ R_3 &\leftarrow e(T_3, g_2)^{r_x} \cdot e(h, w)^{-r_\alpha - r_\beta} \cdot e(h, g_2)^{-r_\delta - r_\mu}, \\ R_4 &\leftarrow T_1^{r_x} \cdot u^{-r_\delta}, & R_5 &\leftarrow T_2^{r_x} \cdot v^{-r_\mu}. \end{aligned} \tag{5.1}$$

5. self-compute the challenge: $c \leftarrow H(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5)$
6. compute the values:

$$\begin{aligned} s_\alpha &\leftarrow r_\alpha + c\alpha, & s_\beta &\leftarrow r_\beta + c\beta, & s_x &\leftarrow r_x + cx, \\ & & s_\delta &\leftarrow r_\delta + c\delta, & s_\mu &\leftarrow r_\mu + c\mu. \end{aligned} \tag{5.2}$$

7. output $\sigma \leftarrow (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$.

- Linkable further times: use $SignLinkable_G(gpk, gsk[i], M)$:
  1. use the same pair (α, β) producing the same linear encryption of A as in the first time: $(T_1, T_2, T_3) = (u^\alpha, v^\beta, Ah^{\alpha+\beta})$;
  2. reuse the helper values $\delta = x\alpha$ and $\mu = x\beta$;
  3. select $r_\alpha', r_\beta', r_x', r_\delta', r_\mu' \xleftarrow{R} \mathbb{Z}_p$;
  4. compute the values $R_1', R_2', R_3', R_4', R_5'$. For computational simplicity, note that all the 3 pairings needed to calculate R_3 can be already precomputed, 2 of them in the setup, namely $p_2 = e(h, w)$ and $p_3 = e(h, g_2)$. This can be achieved because their value is computed from the group public parameters, and the first pairing $p_1 = e(T_3, g_2)$ can be reused from the first signature computed from $Sign_G(gpk, gsk[i], M)$, where T_3 is firstly generated.

$$\begin{aligned} R_1' &\leftarrow u^{r_\alpha'}, & R_2' &\leftarrow v^{r_\beta'}, \\ R_3' &\leftarrow e(T_3, g_2)^{r_x'} \cdot e(h, w)^{-r_\alpha' - r_\beta'} \cdot e(h, g_2)^{-r_\delta' - r_\mu'}, \\ R_4' &\leftarrow T_1^{r_x'} \cdot u^{-r_\delta'}, & R_5' &\leftarrow T_2^{r_x'} \cdot v^{-r_\mu'}. \end{aligned} \tag{5.3}$$

  5. self-compute the challenge: $c' \leftarrow H(M', T_1, T_2, T_3, R_1', R_2', R_3', R_4', R_5')$
  6. compute the values:

$$\begin{aligned} s_\alpha' &\leftarrow r_\alpha' + c'\alpha, & s_\beta' &\leftarrow r_\beta' + c'\beta, & s_x' &\leftarrow r_x' + c'x, \\ & & s_\delta' &\leftarrow r_\delta' + c'\delta, & s_\mu' &\leftarrow r_\mu' + c'\mu. \end{aligned} \tag{5.4}$$

7. output $\sigma' \leftarrow (T_1, T_2, T_3, c', s_\alpha{}', s_\beta{}', s_x{}', s'_\delta, s'_\mu)$.

It becomes easy to verify that several signatures are produced by the same user, as the information (T_1, T_2, T_3) is public in the same signature. In addition, the random values $(r_\alpha, r_\beta, r_x, r_\delta, r_\mu)$ must be different from previous values, that is: $(r_\alpha{}' \neq r_\alpha, r_\beta{}' \neq r_\beta, r_x{}' \neq r_x, r'_\delta \neq r_\delta, r'_\mu \neq r_\mu)$ in order not to reveal information.

$\boldsymbol{VerifyLinkable_G(\sigma, \sigma')}$ This algorithm takes two signatures $\sigma = (T_1, T_2, T_3, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$ and $\sigma' = (T'_1, T'_2, T'_3, c', s_\alpha{}', s_\beta{}', s_x{}', s'_\delta, s'_\mu)$ as input and outputs *true* or $false$ depending on whether the signatures have been produced by the same signer's pseudonym.

$$T_1 \stackrel{?}{=} T_1{}'$$

$$T_2 \stackrel{?}{=} T_2{}'$$

$$T_3 \stackrel{?}{=} T_3{}'$$

5.2.2 System participants

The following participants are involved in the proposed system:

- User $\mathcal{U}$: accesses to the transport system and pays for the received service at the exit. $\mathcal{U}$ performs these actions with her mobile device.
- Service provider ($\mathcal{P}_\mathcal{S}$ source station, $\mathcal{P}_\mathcal{D}$ destination station): checkpoint that controls the tickets used by $\mathcal{U}$. The fare to be paid by $\mathcal{U}$ is computed by $\mathcal{P}_\mathcal{D}$ according to the parameters established (time-based or distance-based fares)
- Payment TTP $\mathcal{M}_\mathcal{C}$: manages all the payments of the users when they exit from the system.
- Group TTP $\mathcal{M}_\mathcal{G}$: manages the group keys and the revocation list. It can revoke the user's anonymity in case of misbehaviour.

5.2.3 Ticket information

In this section, we describe the information that is included in the entrance ticket in Table 5.1, and the information in the exit ticket in Table 5.2. To give a description of the protocols, we use the notation described in Table 5.3.

NAME	NOTATION
Serial number	Sn
Entrance station ($\mathcal{P}_S$ id)	Ps
Entrance timestamp	τ_1
$\mathcal{U}$'s commitment	σ^*
Digital signature	$sig_{\mathcal{P}_S}(\mathsf{t_{in}})$

Table 5.1: Information in entrance ticket $\mathsf{t_{in}}^*$

NAME	NOTATION
$\mathsf{t_{in}}$ serial number	$\mathsf{t_{in}.Sn}$
Destination station	Pd
Paid fare	a
Payment & exit timestamp	τ_2
Digital signature (by $\mathcal{P}_\mathcal{D}$)	$sig_{\mathcal{P}_\mathcal{D}}(\mathsf{t_{out}})$

Table 5.2: Information in exit ticket $\mathsf{t_{out}}^*$

Name	Notation
Group public key	gpk
List of group private keys	$gsk[\,]$
List of group revocations	$grt[\,]$
Exponentiation base	α
Prime number	p
Prime number	q
$\mathcal{U}$'s pseudonym (for payment)	$y_{\mathcal{U}}$
Inverse exponentiation of $y_{\mathcal{U}}$ (secret)	$x_{\mathcal{U}}$
j-th random number	r_j
Exponentiation of r_j	s_j
j-th challenge for $\mathcal{U}$ to show	c_j
Challenge c_j's response by $\mathcal{U}$ authorship of $y_{\mathcal{U}}$	ω_j
Probabilistic encryption of $y_{\mathcal{U}}$	$\delta_{\mathcal{U}}$
j-th timestamp	τ_j
Verification parameter	k
Hash image of parameter k	h_k
Digital signature of the content c	$sig_E(c)$
$\mathcal{U}$'s commitment generated by the entity E	σ^*
Entrance ticket, signed by $\mathcal{P}_{\mathcal{S}}$	$\mathsf{t_{in}}^*$
$\mathsf{t_{in}}$ serial number	Sn
Source service provider identifier	Ps
Exit ticket, signed by $\mathcal{P}_{\mathcal{D}}$	$\mathsf{t_{out}}^*$
Challenge & fare, signed	β^*
Fare calculation function by $\mathcal{P}_{\mathcal{D}}$ for $\mathcal{U}$	$f()$
Fare to be paid	a
Destination service provider identifier	Pd
Probabilistic encryption of $\mathcal{U}$'s verification data	$\gamma_{\mathcal{U}}$
Probabilistic encryption of $\mathcal{P}_{\mathcal{D}}$'s verification data	$\gamma_{\mathcal{P}_{\mathcal{D}}}$
Payment acceptance signed by $\mathcal{M}_{\mathcal{C}}$	ok^*
Payment rejection signed by $\mathcal{M}_{\mathcal{C}}$	ko^*

Table 5.3: Notation information, in appearance order

5.2.4 Protocol specification

Phases

In the protocol, the following phases take place:

- Setup: $\mathcal{M}_\mathcal{G}$ generates all the group keys, revocation lists, etc.
- User Registration: $\mathcal{U}$ registers at $\mathcal{M}_\mathcal{G}$ and receives a group key pair. $\mathcal{U}$ also registers at $\mathcal{M}_\mathcal{C}$ through a pseudonym that will be used only for payments. In the AFC system, $\mathcal{M}_\mathcal{C}$ is an entity that establishes the accounts of user and service provider. This entity processes the related payment messages and guarantees the payment for authorized transactions according to the protocol specifications.
- System entrance: the user joins in the source station and generates a group signature that certifies that she is a valid system group member, while her identity is not disclosed. When this signature is sent to the service provider $\mathcal{P}_\mathcal{S}$, she receives an entrance ticket from $\mathcal{P}_\mathcal{S}$, which will have to be showed at the destination station.
- System exit: the user performs a weak authentication to the destination checkpoint $\mathcal{P}_\mathcal{D}$ and shows the entrance ticket. $\mathcal{P}_\mathcal{D}$ then calculates the fare to be paid. The user has to accept the fare and sends this information securely to $\mathcal{M}_\mathcal{C}$ with her payment pseudonym authentication (only $\mathcal{M}_\mathcal{C}$ has knowledge of this pseudonym, $\mathcal{P}_\mathcal{D}$ can not disclose that information). Then, $\mathcal{M}_\mathcal{C}$ charges the fare to $\mathcal{U}$'s account. If all the process is performed correctly, the user receives an exit ticket, which proves that the user has followed the protocol correctly.

Setup

This phase is executed once at first. $\mathcal{M}_\mathcal{G}$ executes $KeyGen_G(n)$ which generates a group of preset size n, and outputs $(gpk, gsk[\,], grt[\,], \alpha, p, q)$, where gpk is the common group public key, $gsk[i]$ is the private key for each user $\mathcal{U}_i$, $grt[\,]$ is the revocation list, and (α, p, q) are public parameters. The parameter α is the public exponentiation base, and (p, q) prime numbers where $p = 2q + 1$, and they are cardinals of their corresponding groups $\mathbb{Z}_p$ and $\mathbb{Z}_q$. Moreover, each service provider

generates its key pair and shows its public key. The private group keys $gsk[i]$ are issued when users are registered in the group.

User registration

$\mathcal{U}$ registers at the group TTP $\mathcal{M}_\mathcal{G}$ and receives the group key pair $(gpk, gsk[i])$. At this point, the users agree that their identity will be disclosed if they are not honest, or if a judge requires to revoke their anonymity.

Next, $\mathcal{U}$ also registers anonymously to the payment TTP $\mathcal{M}_\mathcal{C}$ with the authorization of $\mathcal{M}_\mathcal{G}$; the user owns a pseudonym $y_\mathcal{U}$ which is an exponentiation of a random value $x_\mathcal{U} \xleftarrow{R} \mathbb{Z}_q$, where $y_\mathcal{U} \leftarrow \alpha^{x_\mathcal{U}} \pmod p$; only this information $y_\mathcal{U}$ will be showed to $\mathcal{M}_\mathcal{C}$ and authenticated through Schnorr's Zero-Knowledge Proof [Schn 91], proving knowledge of $x_\mathcal{U}$ without disclosing that secret. Thus, privacy is preserved for users, but this anonymity could be revoked by $\mathcal{M}_\mathcal{G}$ if necessary. The user registration protocol is defined as follows:

generatePseudonym: The user $\mathcal{U}$ computes:

1. generates her payment pseudonym as a random value $x_\mathcal{U} \xleftarrow{R} \mathbb{Z}_q$;
2. computes $y_\mathcal{U} \leftarrow \alpha^{x_\mathcal{U}} \pmod p$;
3. sends her identity $\mathcal{U}_i$, her certificate $Cert_{\mathcal{U}_i}$ and a signed message containing the pseudonym $sig_\mathcal{U}(y_\mathcal{U} \| \text{`hello'})$ to the Group TTP $\mathcal{M}_\mathcal{G}$;

keyIssue: $\mathcal{M}_\mathcal{G}$ sends the group key pair $(gpk, gsk[i])$ together with the public parameters (α, p, q) and the signature $sig_{\mathcal{M}_\mathcal{G}}(y_\mathcal{U})$ to $\mathcal{U}$;

startingZKP: $\mathcal{U}$ performs:

1. generates a random value $r_0 \xleftarrow{R} \mathbb{Z}_q$;
2. computes $s_0 \leftarrow \alpha^{r_0} \pmod p$;
3. sends $(y_\mathcal{U} \| s_0 \| sig_{\mathcal{M}_\mathcal{G}}(y_\mathcal{U}))$ to the Payment TTP $\mathcal{M}_\mathcal{C}$;

challengeGeneration: $\mathcal{M}_\mathcal{C}$ generates a challenge value $c_0 \xleftarrow{R} \mathbb{Z}_q$ and sends it to $\mathcal{U}$;

proofGeneration: $\mathcal{U}$ computes the Schnorr's ZKP proof $\omega_0 \leftarrow r_0 + c_0 \cdot x_\mathcal{U} \pmod q$ and sends it to $\mathcal{M}_\mathcal{C}$;

verifyPseudonym: $\mathcal{M}_\mathcal{C}$ verifies that $\alpha^{\omega_0} \stackrel{?}{=} s_0 \cdot (y_\mathcal{U})^{c_0}$.

User ($\mathcal{U}$)	**Group TTP ($\mathcal{M}_\mathcal{G}$)**	**Payment TTP ($\mathcal{M}_\mathcal{C}$)**
	User Registration	
$x_\mathcal{U} \xleftarrow{R} \mathbb{Z}_q$		
$y_\mathcal{U} \leftarrow \alpha^{x_\mathcal{U}} \pmod p$		
$\mathcal{U}_i \Vert Cert_{\mathcal{U}_i} \Vert sig_\mathcal{U}(y_\mathcal{U} \Vert \texttt{'hello'})$ $\longrightarrow$		
	$(gpk \Vert gsk[i] \Vert \alpha \Vert p \Vert q \Vert sig_{\mathcal{M}_\mathcal{G}}(y_\mathcal{U}))$ $\longleftarrow$	
$r_0 \xleftarrow{R} \mathbb{Z}_q$		
$s_0 \leftarrow \alpha^{r_0} \pmod p$		
$(y_\mathcal{U} \Vert s_0 \Vert sig_{\mathcal{M}_\mathcal{G}}(y_\mathcal{U}))$ $\longrightarrow$		
		$c_0 \xleftarrow{R} \mathbb{Z}_q$
		c_0 $\longrightarrow$
$\omega_0 \leftarrow r_0 + c_0 \cdot x_\mathcal{U} \pmod q$		
ω_0 $\longrightarrow$		
		verify $\alpha^{\omega_0} \stackrel{?}{=} s_0 \cdot (y_\mathcal{U})^{c_0}$

Table 5.4: User Registration subprotocol

System entrance

When $\mathcal{U}$ has correctly entered the system, an entrance ticket $\mathsf{t_{in}}$ is then received. $\mathsf{t_{in}}$ will be later used in order to authorize the user to pay the calculated fee. The system entrance protocol is defined as follows:

getService: The user $\mathcal{U}$ performs:

1. generates a random value $r_1 \xleftarrow{R} \mathbb{Z}_q$;
2. computes $s_1 \leftarrow \alpha^{r_1} \pmod p$;
3. computes $\delta_\mathcal{U} \leftarrow enc_{pk_{\mathcal{M}_\mathcal{C}}}(y_\mathcal{U})$, where the encryption is probabilistic;
4. generates a random value $k \xleftarrow{R} \mathbb{Z}_q$;
5. computes the *hash()* function of k: $h_k \leftarrow hash(k)$;
6. composes $\sigma \leftarrow (s_1 \Vert \delta_\mathcal{U} \Vert h_k)$, and signs it with $gsk[i]$, her private group key: $\sigma^* \leftarrow (\sigma \Vert \bar{\sigma})$ where $\bar{\sigma} \leftarrow Sign_G(gpk, gsk[i], \sigma)$;

7. sends σ^* to $\mathcal{P}_\mathcal{S}$;

generateTicket: The source service provider $\mathcal{P}_\mathcal{S}$ computes:

1. verifies the signature of σ^*; this entails to check if the signer is a valid group member: $Verify_G(gpk, \sigma, \bar{\sigma})$;
2. generates a timestamp τ_1;
3. composes the entrance ticket $\mathsf{t_{in}} \leftarrow (\mathsf{Sn} \| \mathsf{Ps} \| \tau_1 \| \sigma^*)$ and signs it ${\mathsf{t_{in}}}^* \leftarrow (\mathsf{t_{in}} \| sig_{\mathcal{P}_\mathcal{S}}(\mathsf{t_{in}}))$;
4. sends ${\mathsf{t_{in}}}^*$ to $\mathcal{U}$;

verifyEntrance: $\mathcal{U}$ verifies the signature of ${\mathsf{t_{in}}}^*$ and its content;

User ($\mathcal{U}$)		**Source Service Provider ($\mathcal{P}_S$)**
	System entrance	
$r_1 \xleftarrow{R} \mathbb{Z}_q$		
$s_1 \leftarrow \alpha^{r_1} \pmod p$		
$\delta_{\mathcal{U}} \leftarrow enc_{pk_{\mathcal{M}_C}}(y_{\mathcal{U}})$		
$k \xleftarrow{R} \mathbb{Z}_q$		
$h_k \leftarrow hash(k)$		
$\sigma \leftarrow (s_1 \Vert \delta_{\mathcal{U}} \Vert h_k)$		
$\bar{\sigma} \leftarrow Sign_G(gpk, gsk[i], \sigma))$		
$\sigma^* \leftarrow (\sigma \Vert \bar{\sigma})$		
	$\xrightarrow{\sigma^*}$	
		$Verify_G(gpk, \sigma, \bar{\sigma})$
		generate τ_1
		[TIME] $\mathsf{t_{in}} \leftarrow (\mathsf{Sn} \Vert \mathsf{Ps} \Vert \tau_1 \Vert \sigma^*)$
		[DISTANCE] $\mathsf{t_{in}} \leftarrow (\mathsf{Sn} \Vert \mathsf{Ps} \Vert \tau_1 \Vert \tau_v \Vert \sigma^* \Vert \xi)$
		$\mathsf{t_{in}}^* \leftarrow (\mathsf{t_{in}} \Vert sig_{\mathcal{P}_S}(\mathsf{t_{in}}))$
	$\xleftarrow{\mathsf{t_{in}}^*}$	
verify $\mathsf{t_{in}}^*$		

Table 5.5: System entrance subprotocol

System exit

When the user exits the system, she sends the ticket entrance $\mathsf{t_{in}}$ to the destination service provider $\mathcal{P}_\mathcal{D}$, and the fare to be paid is calculated. If $\mathcal{U}$ behaves correctly, an exit ticket $\mathsf{t_{out}}$ is received, and can be later showed as a receipt, entailing that the protocol has been followed correctly. The system exit protocol is defined as follows:

showTicket: $\mathcal{U}$ encrypts k and sends $(\mathsf{t_{in}}^* \| enc_{pk_{\mathcal{P}_\mathcal{D}}}(k))$ to $\mathcal{P}_\mathcal{D}$;

verifyTicket: The destination service provider $\mathcal{P}_\mathcal{D}$:

1. verifies the signature of $\mathsf{t_{in}}^*$, which is computed by $\mathcal{P}_\mathcal{S}$;
2. verifies that $\sigma.h_k \stackrel{?}{=} hash(k)$, which proves that $\mathcal{U}$ is the right holder of the ticket $\mathsf{t_{in}}$;
3. verifies that $\mathsf{t_{in}}.\mathsf{Sn}$ has not been previously used;
4. generates a timestamp τ_2 (obviously $\tau_1 \leq \tau_2$);
5. calculates the fare to be paid depending on the elapsed time between corresponding timestamps (τ_1, τ_2): $a \leftarrow f_t(\mathsf{t_{in}}.\mathsf{Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$;
   Therefore, in this case, $f_t()$ is a function especially designed for computing the fare between two stations on a time-based fare system.
6. generates a challenge $c_1 \stackrel{R}{\leftarrow} \mathbb{Z}_q$;
7. composes $\beta \leftarrow (\mathsf{t_{in}}^* \| k \| a \| c_1 \| \tau_2 \| \mathsf{Pd})$, and signs it $\beta^* \leftarrow (\beta \| sig_{\mathcal{P}_\mathcal{D}}(\beta))$;
8. sends β^* to $\mathcal{U}$ (in case of dispute, β can be used by $\mathcal{U}$ as an evidence to prove that she has exit at τ_2, see claim 2);
9. composes $\gamma_{\mathcal{P}_\mathcal{D}} \leftarrow (\beta.a \| \mathsf{t_{in}}.\mathsf{Sn} \| \mathsf{t_{in}}.\sigma \| c_1)$;

setPayment: $\mathcal{U}$ takes the following steps:

1. verifies the signature of β^* which is computed by $\mathcal{P}_\mathcal{D}$;
2. computes $\omega_1 \leftarrow r_1 + c_1 \cdot x_\mathcal{U} \pmod q$;
3. composes and encrypts $\gamma_\mathcal{U} \leftarrow enc_{pk_{\mathcal{M}_\mathcal{C}}}(\omega_1 \| \mathsf{t_{in}}.\mathsf{Sn} \| \beta.a)$;
4. sends $\gamma_\mathcal{U}$ to $\mathcal{P}_\mathcal{D}$;

sendingPaymentInfo: $\mathcal{P}_\mathcal{D}$ resends $\gamma_\mathcal{U}$ and $\gamma_{\mathcal{P}_\mathcal{D}}$ to the payment TTP $\mathcal{M}_\mathcal{C}$;

verifyPayment: $\mathcal{M}_\mathcal{C}$:

1. decrypts $\gamma_{\mathcal{U}}$ in order to obtain the Schnorr's proof ω_1;
2. decrypts $\mathsf{t_{in}}.\sigma.\delta_{\mathcal{U}}$ in order to obtain the pseudonym $y_{\mathcal{U}}$ and charge the fee to the corresponding user's account;
3. verifies the identity of $\mathcal{U}$ through Schnorr's ZKP: $\alpha^{\omega_1} \stackrel{?}{=} s_1 \cdot (y_{\mathcal{U}})^{c_1}$;
4. if it is correct, the fare a is charged to the user's account that possesses $y_{\mathcal{U}}$ and the protocol continues. Otherwise, it composes a payment rejection $ko \leftarrow (\text{`}\texttt{authentication error}\text{'} \| \gamma_{\mathcal{U}})$, signs it $ko^* \leftarrow (ko \| sig_{\mathcal{M}_C}(ko))$, sends it to $\mathcal{P}_{\mathcal{D}}$ and stops the protocol;
5. composes $ok \leftarrow (\mathsf{t_{in}}.\mathsf{Sn} \| \beta.a \| \text{`}\texttt{ok}\text{'})$ and signs it $ok^* \leftarrow (ok \| sig_{\mathcal{M}_C}(ok))$;
6. sends ok^* to $\mathcal{P}_{\mathcal{D}}$;

setExit: $\mathcal{P}_{\mathcal{D}}$ takes the following steps:

1. composes $\mathsf{t_{out}} \leftarrow (\mathsf{t_{in}}.\mathsf{Sn} \| \mathsf{Pd} \| \beta.a \| \beta.\tau_2 \| \text{`}\texttt{leaving}\text{'})$ and signs it $\mathsf{t_{out}}^* \leftarrow (\mathsf{t_{out}} \| sig_{\mathcal{P}_{\mathcal{D}}}(\mathsf{t_{out}}))$;
2. sends $\mathsf{t_{out}}^*$ to $\mathcal{U}$ and allows her to exit the system successfully;

checkTicket: $\mathcal{U}$ verifies the signature of $\mathsf{t_{out}}^*$ and its content.

User ($\mathcal{U}$)	**Destination Service Provider ($\mathcal{P}_\mathcal{D}$)**	**Payment TTP ($\mathcal{M}_\mathcal{C}$)**
	System exit	
$enc_{pk_{\mathcal{P}_\mathcal{D}}}(k))$ $(\mathsf{t_{in}}^* \| enc_{pk_{\mathcal{P}_\mathcal{D}}}(k))$ $\longrightarrow$		
	verify $\mathsf{t_{in}}^*$ verify $\sigma.h_k \stackrel{?}{=} hash(k)$ verify $\mathsf{t_{in}.Sn}$ is not used [DISTANCE] verify $\mathsf{t_{in}}.\tau_v$ has not expired [DISTANCE] verify direction $\mathsf{t_{in}}.\xi$ is correct generate timestamp τ_2 [TIME] calculate fare $a \leftarrow f_t(\mathsf{t_{in}.Ps, Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$ [DISTANCE] calculate fare $a \leftarrow f_d(\mathsf{t_{in}.Ps, Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$ $c_1 \stackrel{R}{\leftarrow} \mathbb{Z}_q$ $\beta \leftarrow (\mathsf{t_{in}}^* \| k \| a \| c_1 \| \tau_2 \| \mathsf{Pd})$ $\beta^* \leftarrow (\beta \| sig_{\mathcal{P}_\mathcal{D}}(\beta))$	
β^* $\longleftarrow$		
verify β^* $\omega_1 \leftarrow r_1 + c_1 \cdot x_\mathcal{U} \pmod q$ $\gamma_\mathcal{U} \leftarrow enc_{pk_{\mathcal{M}_\mathcal{C}}}(\omega_1 \| \mathsf{t_{in}.Sn} \| \beta.a)$ $\gamma_\mathcal{U}$ $\longrightarrow$	$\gamma_{\mathcal{P}_\mathcal{D}} \leftarrow (\beta.a \| \mathsf{t_{in}.Sn} \| \mathsf{t_{in}}.\sigma \| c_1)$	
		$\gamma_\mathcal{U} \| \gamma_{\mathcal{P}_\mathcal{D}}$ $\longrightarrow$

		decrypt $\gamma_\mathcal{U}$ and obtain ω_1 proof
		decrypt $\mathsf{t_{in}}.\sigma.\delta_\mathcal{U}$ to obtain pseudonym $y_\mathcal{U}$
		verify identity $\alpha^{\omega_1} \stackrel{?}{=} s_1 \cdot (y_\mathcal{U})^{c_1}$
		charge fare a
		if charge fails{
		$ko \leftarrow (\text{'}\texttt{auth error}\text{'} \| \gamma_\mathcal{U})$
		$ko^* \leftarrow (ko \| sig_{\mathcal{M}_\mathcal{C}}(ko))$
		}else{
		$ok \leftarrow (\mathsf{t_{in}.Sn} \| \beta.a \| \text{'}\texttt{ok}\text{'})$
		$ok^* \leftarrow (ok \| sig_{\mathcal{M}_\mathcal{C}}(ok))$

		ok^* / ko^* $\longleftarrow$
	$\mathsf{t_{out}} \leftarrow (\mathsf{t_{in}.Sn} \| \mathsf{Pd} \| \beta.a \| \beta.\tau_2 \| \text{'}\texttt{leaving}\text{'})$ $\mathsf{t_{out}}^* \leftarrow (\mathsf{t_{out}} \| sig_{\mathcal{P}_\mathcal{D}}(\mathsf{t_{out}}))$	
$\mathsf{t_{out}}^*$ $\longleftarrow$		
verify $\mathsf{t_{out}}^*$		

Table 5.6: System exit subprotocol

5.2.5 User's claims

During the *System exit* protocol, $\mathcal{P}_\mathcal{D}$ could not follow the protocol due to different reasons (i.e. $\mathcal{P}_\mathcal{D}$ may fail, make mistakes, crash or commit dishonest actions). Consequently, the honest user would receive an improper service. To solve this problem, our protocol can face two of the user's claims.

Claim 1: an incorrect β^* is received

During the *System exit* protocol, $\mathcal{U}$ sends the validation information $(\mathsf{t_{in}}^* \| k)$, but $\mathcal{P}_\mathcal{D}$ misbehaves and sends a wrong β^* (e.g. the message has an inaccurate τ_2) to $\mathcal{U}$ or, simply, $\mathcal{P}_\mathcal{D}$ doesn't send it. Then, this user can claim to receive a valid β^* to Payment TTP $\mathcal{M}_\mathcal{C}$ by following these steps:

claim1Request: The user $\mathcal{U}$ resends $(\mathsf{t_{in}}^* \| k)$ and the incorrect β^* (if this is the case) to $\mathcal{M}_\mathcal{C}$;

claim1Response: The Payment TTP $\mathcal{M}_\mathcal{C}$:

1. verifies the signature of $\mathsf{t_{in}}^*$ which is computed by $\mathcal{P}_\mathcal{S}$;
2. verifies that $\sigma.h_k \stackrel{?}{=} hash(k)$, which proves that $\mathcal{U}$ is the right holder of the ticket $\mathsf{t_{in}}$;
3. in case of an incorrect β^*, $\mathcal{M}_\mathcal{C}$ verifies that the parameters $\beta.\tau_2$ or $\beta.a$ are not right (e.g. $\beta.\tau_2$ is greater than the current time)
4. generates a new timestamp τ_2'. This τ_2' have to represent a slightly reduced time than the current time;
5. calculates the fare to be paid depending on the elapsed time between corresponding timestamps (τ_1, τ_2'): $a \leftarrow f_t(\mathsf{Pd}, \mathsf{t_{in}}.\mathsf{Ps}, \mathsf{t_{in}}.\tau_1, \tau_2')$;
6. generates a challenge $c_1 \stackrel{R}{\leftarrow} \mathbb{Z}_q$;
7. composes $\beta \leftarrow (\mathsf{t_{in}}^* \| a \| c_1 \| \tau_2' \| \mathsf{Pd})$, and signs it $\beta^* \leftarrow (\beta \| sig_{\mathcal{M}_\mathcal{C}}(\beta))$;
8. sends β^* to $\mathcal{U}$;

resume: The *System exit* protocol continues normally.

User ($\mathcal{U}$)	**Payment TTP ($\mathcal{M}_\mathcal{C}$)**
Claim 1: an incorrect β^* is received	
$\xrightarrow{(\mathsf{t_{in}}^*, k[,\beta^*])}$	
	verify $\mathsf{t_{in}}^*$
	verify $\sigma.h_k \stackrel{?}{=} hash(k)$
	verify β^*
	verify $\beta.\tau_2, \beta.a$
	[DISTANCE] verify $\mathsf{t_{in}}.\tau_v$ has not expired
	[DISTANCE] verify direction $\mathsf{t_{in}}.\xi$ is correct
	if verifications fail: abort
	generate timestamp τ_2'
	[TIME] calculate fare $a = f_t(\mathsf{t_{in}}.\mathsf{Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2')$
	[DISTANCE] calculate fare $a = f_d(\mathsf{t_{in}}.\mathsf{Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2')$
	$c_1 \xleftarrow{R} \mathbb{Z}_q$
	$\beta = (\mathsf{t_{in}}^*, a, c_1, \tau_2', \mathsf{Pd})$
	$\beta^* = (\beta, Sign_{\mathcal{M}_\mathcal{C}}(\beta))$
$\xleftarrow{\beta^*}$	
continue *System exit* protocol	

Table 5.7: Claim 1 subprotocol

Claim 2: an incorrect $\mathsf{t_{out}}^*$ is received

During the *System exit* protocol, $\mathcal{U}$ sends the validation information $(\mathsf{t_{in}}^* \| k \| \gamma_\mathcal{U})$, but $\mathcal{P}_\mathcal{D}$ misbehaves and sends an incorrect $\mathsf{t_{out}}^*$ to $\mathcal{U}$ or simply refuses to send it. Then, the user can contact the Payment TTP $\mathcal{M}_\mathcal{C}$ and she can claim to receive a valid $\mathsf{t_{out}}^*$ by following these steps:

claim2Request: The user $\mathcal{U}$ resends $(\mathsf{t_{in}}^* \| k \| \beta^* \| \gamma_\mathcal{U})$ to $\mathcal{M}_\mathcal{C}$;

claim2Response: The Payment TTP $\mathcal{M}_\mathcal{C}$:

1. verifies the signature of $\mathsf{t_{in}}^*$ which was computed by $\mathcal{P}_\mathcal{S}$;
2. verifies the identity of $\mathcal{U}$ through Schnorr's ZKP: $\alpha^{\omega_1} \stackrel{?}{=} s_1 \cdot (y_\mathcal{U})^{c_1}$;

3. verifies that $\sigma.h_k \stackrel{?}{=} hash(k)$, which proves that $\mathcal{U}$ is the right holder of the ticket $\mathsf{t_{in}}$;
4. calculates the fare to be paid depending on the elapsed time between corresponding timestamps (τ_1, τ_2) : $a \leftarrow f_t(\mathsf{t_{in}.Ps}, \beta.\mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \beta.\tau_2)$. Then, $\mathcal{M}_\mathcal{C}$ verifies that the calculated value a is equal to $\beta.a$. In case of a negative verification, the user is addressed to execute the protocol specified in claim 1;
5. composes $\mathsf{t_{out}} \leftarrow (\mathsf{t_{in}.Sn} \| \beta.a \| \beta.\tau_2 \| \texttt{'leaving'})$, and signs it ${\mathsf{t_{out}}}^* \leftarrow (\mathsf{t_{out}} \| sig_{\mathcal{M}_\mathcal{C}}(\mathsf{t_{out}}))$;
6. sends ${\mathsf{t_{out}}}^*$ to $\mathcal{U}$;

resume: The *System exit* protocol continues normally.

In both claims, the Payment TTP $\mathcal{M}_\mathcal{C}$ has to warn $\mathcal{P}_\mathcal{D}$ of its misbehavior or communication problems with users. $\mathcal{M}_\mathcal{C}$ also has to alert $\mathcal{P}_\mathcal{D}$ of possible further actions if this problem persists.

User ($\mathcal{U}$)	**Payment TTP ($\mathcal{M}_\mathcal{C}$)**
Claim 2: an incorrect $\mathsf{t_{out}}^*$ is received	
	$\xrightarrow{(\mathsf{t_{in}}^*, k, \beta^*, \gamma_\mathcal{U})}$
	verify $\mathsf{t_{in}}^*$
	verify identity $\alpha^{\omega_1} \stackrel{?}{=} s_1 \cdot (y_\mathcal{U})^{c_1}$
	verify $\sigma.h_k \stackrel{?}{=} hash(k)$
	[TIME] calculate fare $a = f_t(\mathsf{t_{in}}.\mathsf{Ps}, \beta.\mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \beta.\tau_2)$
	[DISTANCE] verify $\mathsf{t_{in}}.\tau_v$ has not expired
	[DISTANCE] verify direction $\mathsf{t_{in}}.\xi$ is correct
	[DISTANCE] calculate fare $a = f_d(\mathsf{t_{in}}.\mathsf{Ps}, \beta.\mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \beta.\tau_2)$
	verify $a \stackrel{?}{=} \beta.a$
	$\mathsf{t_{out}} = (\mathsf{t_{in}}.\mathsf{Sn}, \beta.a, \beta.\tau_2)$
	$\mathsf{t_{out}}^* = (\mathsf{t_{out}}, Sign_{\mathcal{M}_\mathcal{C}}(\mathsf{t_{out}}))$
	$\xleftarrow{\mathsf{t_{out}}^*}$
continue *System exit* protocol	

Table 5.8: Claim 2 subprotocol

5.2.6 Provider's claims

During the *System exit* protocol, $\mathcal{U}$ could not follow the protocol due to different reasons (i.e., $\mathcal{U}$ might fail, make mistakes, crash or commit **dishonest actions**). To solve this problem, our protocol can face two provider's claims.

Claim 3: An incorrect $(\mathsf{t_{in}}^* \| k)$ is received

During the *System exit* protocol, $\mathcal{P}_\mathcal{D}$ receives the first step of the verification information $(\mathsf{t_{in}}^* \| k)$, but this information could be not correct, or could not link. Then, this service provider can claim to disclose the user's identity by following these steps:

claim3Request: The destination service provider $\mathcal{P}_\mathcal{D}$ sends $(\mathsf{t_{in}}^* \| k)$ to $\mathcal{M}_\mathcal{G}$;

appealingUser: The user $\mathcal{U}$ is required to also send $(\mathsf{t_{in}}^* \| k)$ to $\mathcal{M}_\mathcal{G}$, in order to avoid false accusations;

claim3Response: If $\mathcal{U}$ does not send the required items, the Group TTP $\mathcal{M}_\mathcal{G}$ computes:

1. verifies the signature of $(\mathsf{t_{in}}^* \| k)$ which is generated by $\mathcal{P}_\mathcal{S}$;
2. verifies the link with the hash value $\mathsf{t_{in}}.\sigma.h_k \stackrel{?}{=} hash(k)$; If the link is not verified, $\mathcal{M}_\mathcal{G}$ aborts the claim;
3. verifies the group signature of $\mathsf{t_{in}}.\sigma^*$ which is generated by $\mathcal{U}$, disclosing then who is the signer inside the group;
4. sends the user identification $\mathcal{U}_i$ to $\mathcal{P}_\mathcal{D}$ and $y_\mathcal{U}$ to $\mathcal{M}_\mathcal{C}$;
5. $\mathcal{U}_i$ is added to the revocated list;

Provider ($\mathcal{P}_\mathcal{D}$)	**Group TTP ($\mathcal{M}_\mathcal{G}$)**	**User ($\mathcal{U}$)**
Claim 3: An incorrect $(\mathsf{t_{in}}^* \| k)$ is received		
$\xrightarrow{(\mathsf{t_{in}}^* \| k)}$		
		$\xleftarrow{(\mathsf{t_{in}}^* \| k)'}$
	if $\mathcal{U}$ does not send, then:	
	verify $(\mathsf{t_{in}}^*, k)$	
	verify $\mathsf{t_{in}}.\sigma.h_k \stackrel{?}{=} hash(k)$	
	verify $\mathsf{t_{in}}.\sigma^*$	
$\xleftarrow{\mathcal{U}_i}$		
add $\mathcal{U}_i$ to revocation list		

Table 5.9: Claim 3 subprotocol

Claim 4: An incorrect $\gamma_{\mathcal{U}}$ is received

During the *System exit* protocol, $\mathcal{P}_{\mathcal{D}}$ and $\mathcal{M}_{\mathcal{C}}$ receive the last step of the verification information $\gamma_{\mathcal{U}}$, but this information could be not correct. Then, this service provider can claim to disclose the user's identity by following these steps:

claim4Request: The Payment TTP $\mathcal{M}_{\mathcal{C}}$ composes a payment rejection $ko \leftarrow (\text{`}\texttt{verification information error}'\|\gamma_{\mathcal{U}})$, signs $ko^* \leftarrow (ko\|sig_{\mathcal{M}_{\mathcal{C}}}(ko))$ and sends it to $\mathcal{P}_{\mathcal{D}}$. The Payment TTP $\mathcal{M}_{\mathcal{C}}$ also sends $(sig_{\mathcal{P}_{\mathcal{D}}}(\gamma_{\mathcal{U}})\|\gamma_{\mathcal{P}_{\mathcal{D}}})$ to $\mathcal{M}_{\mathcal{G}}$ and stops the protocol.

providerInfo: The destination service provider $\mathcal{P}_{\mathcal{D}}$ sends $({t_{in}}^*\|k)$ to $\mathcal{M}_{\mathcal{G}}$;

appealingUser: $\mathcal{U}$ is required to also send $({t_{in}}^*\|k\|\gamma_{\mathcal{U}})$ to $\mathcal{M}_{\mathcal{G}}$, in order to avoid false accusations;

claim4Response: If $\mathcal{U}$ does not send the required items, the Group TTP $\mathcal{M}_{\mathcal{G}}$ computes:

1. verifies if the decrypted information of $\gamma_{\mathcal{U}}\|\gamma_{\mathcal{P}_{\mathcal{D}}}$ and $({t_{in}}^*\|k)$ link;
2. verifies the group signature of $t_{in}.\sigma^*$ which is generated by $\mathcal{U}$, disclosing then who is the signer inside the group;
3. sends the user identification $\mathcal{U}_i$ to $\mathcal{P}_{\mathcal{D}}$ and $y_{\mathcal{U}}$ to $\mathcal{M}_{\mathcal{C}}$;
4. $\mathcal{U}_i$ is added to the revocated list;

Provider/Payment TTP ($\mathcal{P}_\mathcal{D}$ / $\mathcal{M}_\mathcal{C}$)	**Group TTP ($\mathcal{M}_\mathcal{G}$)**	**User ($\mathcal{U}$)**

Claim 4: An incorrect $\gamma_\mathcal{U}$ is received

$ko = (error \| \gamma_\mathcal{U})$

$ko^* = (ko \| sig_{\mathcal{M}_\mathcal{C}}(ko))$

$\xrightarrow{ko^* \| sig_{\mathcal{P}_\mathcal{D}}(\gamma_\mathcal{U}) \| \gamma_{\mathcal{P}_\mathcal{D}}}$

$\xleftarrow{(\mathsf{t_{in}}^* \| k \| \gamma_\mathcal{U})}$

if $\mathcal{U}$ does not send, then:

verify $\gamma_\mathcal{U}, \gamma_{\mathcal{P}_\mathcal{D}}, (\mathsf{t_{in}}^* \| k)$

verify $\mathsf{t_{in}}.\sigma^*$

$\xleftarrow{\mathcal{U}_i - or - y_\mathcal{U}}$

add $\mathcal{U}_i$ to revocation list

Table 5.10: Claim 4 subprotocol

5.3 Distance-based fare collection protocol

In this section, we describe our Distance-Based Fare Collection system, which provides anonymity to the users by the use of group signatures [Bone 04b] for mass-transport services. We have adapted the Time-Based protocol in §5.2 into a Distance-Based Fare Collection Protocol with small changes. Then, the complexity of the adaptation depends mainly on the degree of the requirements' compliance described in §5.1.3.

Firstly, we show the changes to be made to the requirement compliant distance-based systems in §5.3.1. Later, in §5.3.2, we describe the services which are non-compliant, introducing then a fraud attack: the colluding attack, in §5.3.3. Finally, in §5.3.4, we describe the changes to be performed in the non-compliant distance-based systems.

5.3.1 Requirement compliant distance-based systems

Only some changes have to be performed to the protocol presented in §5.2 in order to adapt it to a distance-based automated fare collection system that fulfils the requirements presented in §5.1.3. We only have to add one item to the information gathered in the entrance ticket. We add the direction of the journey (ξ) and the validity time (τ_v), so now an entrance ticket is: $\mathsf{t_{in}} = (\mathsf{Sn}\|\mathsf{Ps}\|\tau_1\|\tau_v\|\sigma^*\|\xi)$.

generateTicket: The source service provider $\mathcal{P}_\mathcal{S}$ computes:

1. verifies the signature of σ^*; this entails to check if the signer is a valid group member: $Verify_G(gpk, \sigma, \bar{\sigma})$;
2. generates a timestamp τ_1;
3. composes the entrance ticket: $\mathsf{t_{in}} \leftarrow (\mathsf{Sn}\|\mathsf{Ps}\|\tau_1\|\tau_v\|\sigma^*\|\xi)$ and signs it: $\mathsf{t_{in}}^* \leftarrow (\mathsf{t_{in}}\|sig_{\mathcal{P}_\mathcal{S}}(\mathsf{t_{in}}))$;
4. sends $\mathsf{t_{in}}^*$ to $\mathcal{U}$;

Concerning the protocol specifications, we only have to change the action **verifyTicket** performed by $\mathcal{P}_\mathcal{D}$ in the system exit subprotocol and the **claim1Response** and **claim2Response**, because we have to modify the way of calculating the fare according to a distance-based criteria. Then, **verifyTicket** for a distance-based scheme is performed as follows:

verifyTicket: The destination service provider $\mathcal{P}_{\mathcal{D}}$ performs:

1. verifies the signature of $\mathsf{t_{in}}^*$ which is computed by $\mathcal{P}_{\mathcal{S}}$;
2. verifies that $\sigma.h_k \stackrel{?}{=} hash(k)$, which proves that $\mathcal{U}$ is the right holder of the ticket $\mathsf{t_{in}}$;
3. verifies that $\mathsf{t_{in}.Sn}$ has not been previously used;
4. verifies that the validity time τ_v has not expired, and that the direction ξ is correct;
5. generates a timestamp τ_2 (obviously $\tau_1 \leq \tau_2$);
6. calculates the fare depending on the entrance station ($\mathsf{t_{in}.Ps}$), the exit station (Pd) and their corresponding timestamps (τ_1, τ_2): $a \leftarrow f_d(\mathsf{t_{in}.Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$;
   Therefore, in this case, $f_d()$ is a function especially designed to compute the fare between two stations on a distance-based fare system.
7. generates a challenge $c_1 \stackrel{R}{\leftarrow} \mathbb{Z}_q$;
8. composes $\beta \leftarrow (\mathsf{t_{in}}^* \| k \| a \| c_1 \| \tau_2 \| \mathsf{Pd})$, and signs it $\beta^* \leftarrow (\beta \| sig_{\mathcal{P}_{\mathcal{D}}}(\beta))$;
9. sends β^* to $\mathcal{U}$ (in case of a dispute, β can be used by $\mathcal{U}$ as an evidence to prove that she has exit at τ_2- see claim 2);
10. composes $\gamma_{\mathcal{P}_{\mathcal{D}}} \leftarrow (\beta.a \| \mathsf{t_{in}.Sn} \| \mathsf{t_{in}}.\sigma \| c_1)$;

The **claim1Response** function in a distance-based system has to work as follows:

claim1Response: The Payment TTP $\mathcal{M}_{\mathcal{C}}$ computes:

1. verifies the signature of $\mathsf{t_{in}}^*$ which is computed by $\mathcal{P}_{\mathcal{S}}$;
2. verifies that $\sigma.h_k \stackrel{?}{=} hash(k)$, which proves that $\mathcal{U}$ is the right holder of the ticket $\mathsf{t_{in}}$;
3. in case of an incorrect β^*, $\mathcal{M}_{\mathcal{C}}$ verifies that the parameters $\beta.\tau_2$ or $\beta.a$ are not right (e.g. $\beta.\tau_2$ is greater than the current time)
4. verifies that the validity time τ_v has not expired, and that the direction ξ is correct;
5. generates a new timestamp τ_2.

6. calculates the fare to be paid depending on the entrance station ($\mathsf{t_{in}}.\mathsf{Ps}$) and the exit station ($\mathsf{Pd}$): $a \leftarrow f_d(\mathsf{Pd}, \mathsf{t_{in}}.\mathsf{Ps}, \mathsf{t_{in}}.\tau_1, \tau_2)$;
7. generates a challenge $c_1 \overset{R}{\leftarrow} \mathbb{Z}_q$;
8. composes $\beta \leftarrow (\mathsf{t_{in}}^* \| a \| c_1 \| \tau_2 \| \mathsf{Pd})$, and signs it $\beta^* \leftarrow (\beta \| sig_{\mathcal{M}_\mathcal{C}}(\beta))$;
9. sends β^* to $\mathcal{U}$;

Finally, the **claim2Response** has to be as follows:

claim2Response: The Payment TTP $\mathcal{M}_\mathcal{C}$ computes:

1. verifies the signature of $\mathsf{t_{in}}^*$ which was computed by $\mathcal{P}_\mathcal{S}$;
2. verifies the identity of $\mathcal{U}$ through Schnorr's ZKP: $\alpha^{\omega_1} \overset{?}{=} s_1 \cdot (y_\mathcal{U})^{c_1}$;
3. verifies that $\sigma.h_k \overset{?}{=} hash(k)$, which proves that $\mathcal{U}$ is the right holder of the ticket $\mathsf{t_{in}}$;
4. verifies that the validity time τ_v has not expired, and that the direction ξ is correct;
5. calculates the fare to be paid depending on the entrance station ($\mathsf{t_{in}}.\mathsf{Ps}$) and the exit station ($\beta.\mathsf{Pd}$): $a \leftarrow f_d(\mathsf{t_{in}}.\mathsf{Ps}, \beta.\mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \beta.\tau_2)$. Then, $\mathcal{M}_\mathcal{C}$ verifies that the calculated value a is equal to $\beta.a$. In case of a negative verification, the user will be addressed to execute the claim 1 subprotocol;
6. composes $\mathsf{t_{out}} \leftarrow (\mathsf{t_{in}}.\mathsf{Sn} \| \beta.a \| \beta.\tau_2 \| \texttt{'leaving'})$, and signs it $\mathsf{t_{out}}^* \leftarrow (\mathsf{t_{out}} \| sig_{\mathcal{M}_\mathcal{C}}(\mathsf{t_{out}}))$;
7. sends $\mathsf{t_{out}}^*$ to $\mathcal{U}$;

5.3.2 Non-compliant distance-based services

In this section, we treat the services which depend on distance and do not fulfil the requirements. Users could access and exit the system in different points, but not separating them by their direction in this case. That is, from this point of view, entrances and exits may be indistinguishable in some parts of the system, as they have common areas. In this situation, a major complexity in the transport system could open security holes and require then deeper security measures. We describe a possible colluding attack in the following section.

5.3.3 Colluding attacks

In this section, the problem of station entrances and exits in a distance-based fare collection system, which are not surely distinguishable, is addressed. If the stations do not differentiate their exits depending on the direction, then the colluding attacks could be easily performed without detection. Imagine that a determined user $\mathcal{U}_1$ joins the system in a determined station $\mathcal{P}_{\mathcal{S}1}$ and exits the system in $\mathcal{P}_{\mathcal{D}1}$, and another $\mathcal{U}_2$ joins in $\mathcal{P}_{\mathcal{S}2}$ and exits in $\mathcal{P}_{\mathcal{D}2}$. Consequently, they would have to pay their fares depending on the differences of distance between stations, that is: $f_{D_1}(\mathcal{P}_{\mathcal{S}1}, \mathcal{P}_{\mathcal{D}1}, \tau_{11}, \tau_{12})$ and $f_{D_2}(\mathcal{P}_{\mathcal{S}2}, \mathcal{P}_{\mathcal{D}2}, \tau_{21}, \tau_{22})$. In this scenario, users could collaborate in some advantaging cases in order that they both pay less if they exchanged their received entrance tickets; that is: ${f_{D_1}}'(\mathcal{P}_{\mathcal{S}2}, \mathcal{P}_{\mathcal{D}1}, \tau_{21}, \tau_{12})$ and ${f_{D_2}}'(\mathcal{P}_{\mathcal{S}1}, \mathcal{P}_{\mathcal{D}2}, \tau_{11}, \tau_{22})$. For a graphical explanation, see Fig. 5.2, where we can see that the cars traveling in opposite directions can enter the system through the same station. After obtaining a ticket, they are able to choose the direction. Similarly, they exit the system using the same provider. The provider is not able to determine what was the direction of the car. This fact can be used by users to try confabulated attacks.

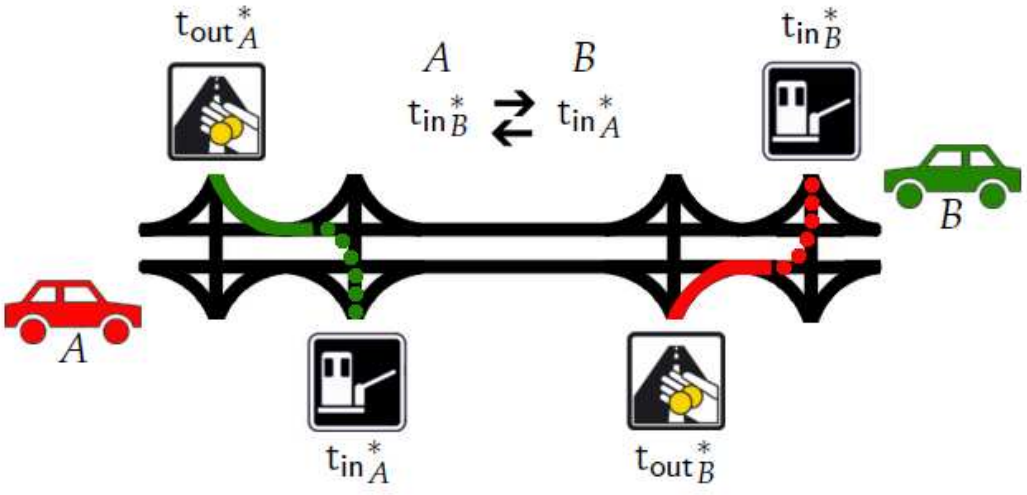

Figure 5.2: Non-compliant distance-based AFC.

With the current system, it would be quite easy to perform it without detection, as the entrance and exit tickets are not *hard-linked* between them. In Fig. 5.3, we have analyzed the fraud ratio, regarding the cases where users could take advantage of this exchange and would not be detectable with the previous system. The scenario is a lineal sort of stations with only common areas of entrances/exits, and we evaluate this ratio depending on the number of stations.

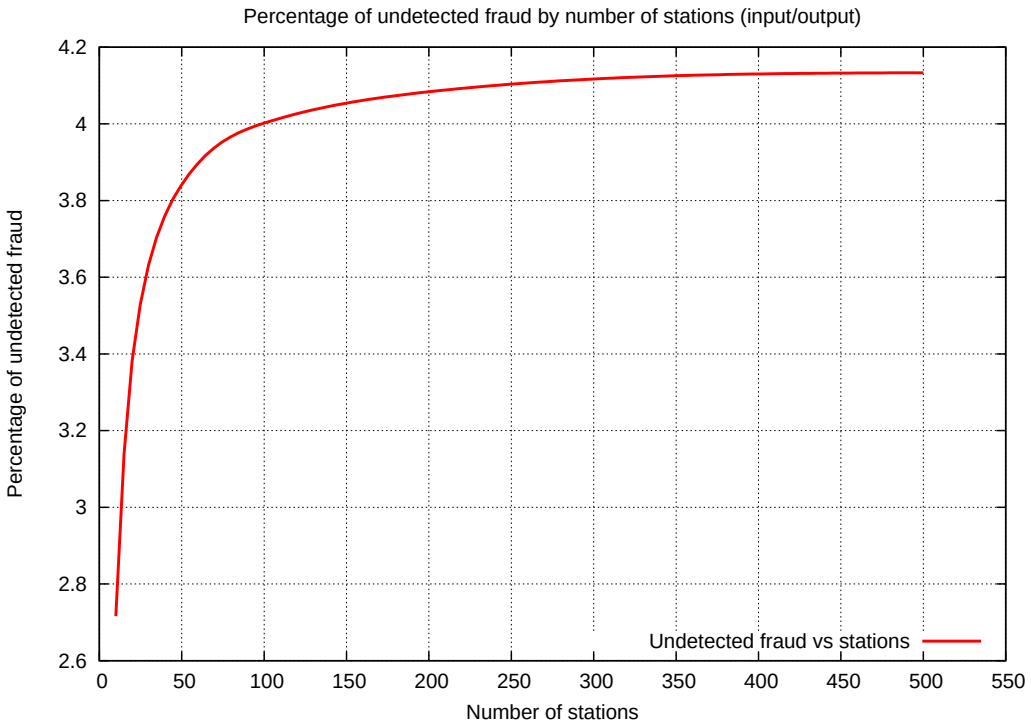

Figure 5.3: Percentage of undetected fraud by number of stations in a linear scenario (in %)

In the next section, we propose a system that can be used in order to detect this colluding attack.

5.3.4 Distance-based fare collection for non-compliant services

Our intention is to produce a hard link between the entrance ticket and the proofs at the exit. The major change is that the k parameter and its hash function have been replaced by a second signature in the exit, which is linkable to the signature in the entrance (see §5.2.1 for details). Taking this modification into account, security is incremented, but their computational costs will be higher. An analysis

and comparison of the computational times is presented in §5.5. In this section, we only describe here only the changes to be applied in the current protocol.

The *System entrance* changes in the protocol are:

getService: The user $\mathcal{U}$ performs:

1. generates a random value $r_1 \xleftarrow{R} \mathbb{Z}_q$;
2. computes $s_1 \leftarrow \alpha^{r_1} \pmod{p}$;
3. computes $\delta_{\mathcal{U}} \leftarrow enc_{pk_{\mathcal{MC}}}(y_{\mathcal{U}})$, where the encryption is probabilistic;
4. composes $\sigma \leftarrow (s_1 \| \delta_{\mathcal{U}})$, and signs it with $gsk[i]$, her private group key: $\sigma^* \leftarrow (\sigma \| \bar{\sigma} \leftarrow Sign_G(gpk, gsk[i], \sigma))$;
5. sends σ^* to $\mathcal{P}_{\mathcal{S}}$;

generateTicket: The source service provider $\mathcal{P}_{\mathcal{S}}$:

1. verifies the signature of σ^*; this entails to check if the signer is a valid group member: $Verify_G(gpk, \sigma, \bar{\sigma})$;
2. generates a timestamp τ_1;
3. composes the entrance ticket $\mathsf{t_{in}} \leftarrow (\mathsf{Sn} \| \mathsf{Ps} \| \tau_1 \| \tau_v \| \sigma^* \| \xi)$ and signs it: $\mathsf{t_{in}}^* \leftarrow (\mathsf{t_{in}} \| sig_{\mathcal{P}_{\mathcal{S}}}(\mathsf{t_{in}}))$;
4. sends $\mathsf{t_{in}}^*$ to $\mathcal{U}$;

The modification of the *System exit* protocol works as follows:

previousStep: $\mathcal{P}_{\mathcal{D}}$ generates a $\Phi \xleftarrow{R} \mathbb{Z}_p$ value and sends it to $\mathcal{U}$;

showTicket: $\mathcal{U}$:

1. signs the received Φ as the same member as in the entrance: $\Phi^* \leftarrow (\Phi \| \bar{\Phi})$ where $\bar{\Phi} \leftarrow SignLinkable_G(gpk, gsk[i], \Phi))$;
2. sends $(\mathsf{t_{in}}^* \| \Phi^*))$ to $\mathcal{P}_{\mathcal{D}}$;

verifyTicket: The destination service provider $\mathcal{P}_{\mathcal{D}}$ performs:

1. verifies the signature of $\mathsf{t_{in}}^*$, which is computed by $\mathcal{P}_{\mathcal{S}}$;
2. verifies the group signature of Φ^*: $(Verify_G(gpk, \Phi, \bar{\Phi}))$, and that it is the same member as in the entrance: $VerifyLinkable_G(\mathsf{t_{in}}.\sigma^*, \Phi^*)$;
3. verifies that $\mathsf{t_{in}}.\mathsf{Sn}$ has not been previously used;

4. verifies that the validity time τ_v has not expired, and that the direction ξ is correct;
5. generates a timestamp τ_2 (obviously $\tau_1 \leq \tau_2$);
6. calculates the fare to be paid depending on the entrance station ($\mathsf{t_{in}}.\mathsf{Ps}$), the exit station ($\mathsf{Pd}$) and their corresponding timestamps (τ_1, τ_2): $a \leftarrow f_d(\mathsf{t_{in}}.\mathsf{Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$;
7. generates a challenge $c_1 \xleftarrow{R} \mathbb{Z}_q$;
8. composes $\beta \leftarrow (\mathsf{t_{in}}^* \| a \| c_1 \| \tau_2 \| \mathsf{Pd})$, and signs it $\beta^* \leftarrow (\beta \| sig_{\mathcal{P}_\mathcal{D}}(\beta))$;
9. sends β^* to $\mathcal{U}$ (in case of dispute β can be used by $\mathcal{U}$ as an evidence to prove that she has exit at τ_2- see claim 2);
10. composes $\gamma_{\mathcal{P}_\mathcal{D}} \leftarrow (\beta.a \| \mathsf{t_{in}}.\mathsf{Sn} \| \mathsf{t_{in}}.\sigma \| c_1)$;

The changes in *Claim 1* are:

claim1Request: The user $\mathcal{U}$ resends $(\mathsf{t_{in}}^* \| \Phi^*)$ and the incorrect β^* (if this is the case) to $\mathcal{M}_\mathcal{C}$;

claim1Response: The Payment TTP $\mathcal{M}_\mathcal{C}$:

1. verifies the signature of $\mathsf{t_{in}}^*$ which is computed by $\mathcal{P}_\mathcal{S}$;
2. verifies the group signature of Φ^*: $(Verify_G(gpk, \Phi, \bar{\Phi}))$, and that it is the same member than in the entrance: $VerifyLinkable_G(\mathsf{t_{in}}.\sigma^*, \Phi^*)$;
3. verifies that the validity time τ_v has not expired, and that the direction ξ is correct;
4. in case of an incorrect β^*, $\mathcal{M}_\mathcal{C}$ verifies that the parameters $\beta.\tau_2$ or $\beta.a$ are not right (e.g. $\beta.\tau_2$ is greater than the current time)
5. generates a new timestamp τ_2. This τ_2 has to represent a slightly more reduced time than the current time. $\mathcal{M}_\mathcal{C}$ can do that in order to compensate the user due to the time overhead produced by the present transaction, in relation to the time when the system exit subprotocol was executed;
6. calculates the fare to be paid depending on the entrance station ($\mathsf{t_{in}}.\mathsf{Ps}$), the exit station ($\mathsf{Pd}$) and their corresponding timestamps (τ_1, τ_2): $a \leftarrow f_d(\mathsf{t_{in}}.\mathsf{Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$;

7. generates a challenge $c_1 \xleftarrow{R} \mathbb{Z}_q$;
8. composes $\beta \leftarrow (\mathsf{t_{in}}^* \| a \| c_1 \| \tau_2 \| \mathsf{Pd})$, and signs it $\beta^* \leftarrow (\beta \| sig_{\mathcal{M}_\mathcal{C}}(\beta))$;
9. sends β^* to $\mathcal{U}$;

The changes in *Claim 2* are defined as follows:

claim2Request: The user $\mathcal{U}$ resends $(\mathsf{t_{in}}^* \| \Phi^* \| \beta^* \| \gamma_\mathcal{U})$ to $\mathcal{M}_\mathcal{C}$;

claim2Response: The Payment TTP $\mathcal{M}_\mathcal{C}$:

1. verifies the signature of $\mathsf{t_{in}}^*$, which was computed by $\mathcal{P}_\mathcal{S}$;
2. verifies that the validity time τ_v has not expired, and that the direction ξ is correct;
3. verifies the identity of $\mathcal{U}$ through Schnorr's ZKP: $\alpha^{\omega_1} \stackrel{?}{=} s_1 \cdot (y_\mathcal{U})^{c_1}$;
4. verifies the group signature of Φ^*: $(Verify_G(gpk, \Phi, \bar{\Phi}))$ and, that it is the same member as in the entrance: $VerifyLinkable_G(\mathsf{t_{in}}.\sigma^*, \Phi^*)$;
5. calculates the fare to be paid depending on the entrance station ($\mathsf{t_{in}}.\mathsf{Ps}$), the exit station ($\mathsf{Pd}$) and their corresponding timestamps (τ_1, τ_2): $a \leftarrow f_d(\mathsf{t_{in}}.\mathsf{Ps}, \mathsf{Pd}, \mathsf{t_{in}}.\tau_1, \tau_2)$. Then, $\mathcal{M}_\mathcal{C}$ verifies that the calculated value a is equal to $\beta.a$;
6. composes $\mathsf{t_{out}} \leftarrow (\mathsf{t_{in}}.\mathsf{Sn} \| \beta.a \| \beta.\tau_2 \| \texttt{'leaving'})$, and signs it $\mathsf{t_{out}}^* \leftarrow (\mathsf{t_{out}} \| sig_{\mathcal{M}_\mathcal{C}}(\mathsf{t_{out}}))$;
7. sends $\mathsf{t_{out}}^*$ to $\mathcal{U}$;

Equally, the changes in *Claim 3* are defined as follows:

claim3Request: The destination service provider $\mathcal{P}_\mathcal{D}$ sends $(\mathsf{t_{in}}^* \| \Phi^*)$ to $\mathcal{M}_\mathcal{G}$;

appealingUser: The user $\mathcal{U}$ is required to also send $(\mathsf{t_{in}}^* \| \Phi^*)$ to $\mathcal{M}_\mathcal{G}$, in order to avoid false accusations;

claim3Response: If $\mathcal{U}$ does not send the required items, the Group TTP $\mathcal{M}_\mathcal{G}$ performs the following steps:

1. verifies the signature of $(\mathsf{t_{in}}^* \| \Phi^*)$ which is generated by $\mathcal{P}_\mathcal{S}$;
2. verifies the group signature of Φ^*: $(Verify_G(gpk, \Phi, \bar{\Phi}))$, and that it is the same member as in the entrance: $VerifyLinkable_G(\mathsf{t_{in}}.\sigma^*, \Phi^*)$;

3. verifies the group signature of $\mathsf{t_{in}}.\sigma^*$ which is generated by $\mathcal{U}$, disclosing then who is the signer inside the group with $Open_G(gpk, gmsk, \mathsf{t_{in}}.\sigma^*)$;
4. sends the user identification $\mathcal{U}_i$ to $\mathcal{P}_\mathcal{D}$ and $y_\mathcal{U}$ to $\mathcal{M}_\mathcal{C}$;
5. $\mathcal{U}_i$ is added to the revocated list;

Finally, the changes in *Claim 4* are defined as follows:

providerInfo: The destination service provider $\mathcal{P}_\mathcal{D}$ sends $(\mathsf{t_{in}}^* \| \Phi^*)$ to $\mathcal{M}_\mathcal{G}$;

appealingUser: The user $\mathcal{U}$ is required to also send $(\mathsf{t_{in}}^* \| \Phi^* \| \gamma_\mathcal{U})$ to $\mathcal{M}_\mathcal{G}$, in order to avoid false accusations;

claim4Response: If $\mathcal{U}$ does not send the required items, the Group TTP $\mathcal{M}_\mathcal{G}$ performs the following steps:

1. verifies if the decrypted information of $\gamma_\mathcal{U}$, $\gamma_{\mathcal{P}_\mathcal{D}}$ and $(\mathsf{t_{in}}^*, \Phi^*)$ link;
2. verifies the group signature of $\mathsf{t_{in}}.\sigma^*$ which is generated by $\mathcal{U}$, disclosing then who is the signer inside the group with $Open_G(gpk, gmsk, \mathsf{t_{in}}.\sigma^*)$;
3. sends the user identification $\mathcal{U}_i$ to $\mathcal{P}_\mathcal{D}$ and $y_\mathcal{U}$ to $\mathcal{M}_\mathcal{C}$;
4. $\mathcal{U}_i$ is added to the revocated list;

5.4 Security and privacy considerations

Proposition 5.1. *The proposed system preserves authenticity, non-repudiation and integrity for the entrance and exit tickets.*

Claim 5.1.1. *The creation of fraudulent tickets is computationally unfeasible nowadays.*

Security Argument. On the one hand, the tickets are signed: $\mathsf{t_{in}}^* = (\mathsf{t_{in}} \| sig_{\mathcal{P}_\mathcal{S}}(\mathsf{t_{in}}))$ and $\mathsf{t_{out}}^* = (\mathsf{t_{out}} \| sig_{\mathcal{P}_\mathcal{D}}(\mathsf{t_{out}}))$ together with the sent information before the payment $\beta^* = (\beta \| sig_{\mathcal{P}_\mathcal{D}}(\beta))$. If an unauthorized entity can create a valid ticket (entrance or exit) without knowledge of the private keys of either $\mathcal{P}_\mathcal{S}$ or $\mathcal{P}_\mathcal{D}$, it could generate digital signatures while impersonating these providers. Supposing that we use a secure digital signature scheme, this operation is considered unfeasible. On the other hand, the user sends the verification information signed with her group private key $\sigma^* = (\sigma \| Sign_\mathcal{G}(\sigma))$. For the same reason, this signature guarantees that the message is authentic and that has been issued by a valid user (and not revoked) inside the group.

Claim 5.1.2. *The issuer of a ticket can not deny the emission of this ticket.*

Security Argument. The tickets are signed by its authorized issuer (service providers) and, by considering that the used signature scheme is secure, this operation could be only performed by these issuers. Thus, the issuer's identity is linked to the ticket and, for the properties of the electronic signature scheme, that issuer can not deny its authorship. The same occurs with the group signature scheme, in which the message authorship can be verified if identity is disclosed.

Claim 5.1.3. *The content of the tickets cannot be modified.*

Security Argument. If we suppose that the signature scheme is secure, that the *hash* summary function is collision-resistant, and that its inverse function is computationally unfeasible nowadays, if the ticket content was modified, the verification of the signature would then be incorrect. In order to pass the verification, the signature would need to be regenerated from the new ticket content. This operation is computationally unfeasible nowadays with the most current machines. The same occurs with the group signature scheme.

Result 5.1. *According to the definitions given in §5.1.1 and the Claims 5.1.1, 5.1.2 and 5.1.3, we can assure that the protocol achieves the security requirements of authenticity, non-repudiation and integrity.*

Proposition 5.2. *The system described in this proposal achieves revocable anonymity for users, and all the movements performed by a same user are untraceable between each other if the service providers attempt to trace them.*

Claim 5.2.1. *A ticket is anonymous.*

Security Argument. The information related to the user's identity is encrypted with the payment TTP's public key. The service providers ($\mathcal{P}_{\mathcal{S}}$ and $\mathcal{P}_{\mathcal{D}}$) can not access to this information because they need the private key of the TTP. In the system, users compute a group signature ($\mathsf{t}_{\mathsf{in}}.\sigma^* = (\sigma \| Sign_{\mathcal{G}}(\sigma))$) which certifies that the signer is a valid group member. If we analyse the properties of the group signature scheme, the providers cannot disclose the identity of the signature generator. In case of a controversial situation, the identity of the user who signed the content could be disclosed through the cooperation of both payment TTP $\mathcal{M}_{\mathcal{C}}$ and the group TTP $\mathcal{M}_{\mathcal{G}}$. If the user appears in the revocation list, her identity is revealed, thus enabling further actions.

Claim 5.2.2. *The user is anonymous, under the point of view of the service providers, during the payment phase.*

Security Argument. All the information related to the payment is encrypted and only the payment TTP can access to it. The service providers are excluded from the payment, and they only receive the payment confirmation from the payment TTP $\mathcal{M}_{\mathcal{C}}$. Then, $\mathcal{M}_{\mathcal{C}}$ has knowledge about $y_{\mathcal{U}}$ from the pair $(x_{\mathcal{U}}, y_{\mathcal{U}})$ where $y_{\mathcal{U}} = \alpha^{x_{\mathcal{U}}} \pmod{p}$, which identifies her as a valid user; then, the user authenticates by proving knowledge of $x_{\mathcal{U}}$ through Schnorr's ZKP [Schn 91].

Claim 5.2.3. *Multiple group signatures performed by the same user must be unlinkable between each other by the service providers or by other entities external to the system.*

Security Argument. The group signature proposal [Bone 04b] by Boneh, Boyen and Shacham uses a probabilistic signature scheme, that is, it is not possible to predict a ciphertext given a certain plaintext. This allows unlinkability between different group signatures performed by the same user.

Result 5.2. *According to the definitions given in §5.1.1 and the Claims 5.2.1, 5.2.2 and 5.2.3, we can assure that the protocol achieves the security requirements of revocable anonymity and unlinkability.*

Proposition 5.3. *The protocol avoids ticket overspending and also guarantees the control of the validity times.*

Claim 5.3.1. *The protocol avoids ticket overspending.*

Security Argument. If a user tries to overspend an entrance ticket, the serial number will be marked as already used. If this user misbehaviour can be proved, the group TTP $\mathcal{M}_{\mathcal{G}}$ could include this user to the revocation list.

Claim 5.3.2. *The ticket can not be further valid if its validity time τ_v has expired.*

Security Argument. The destination station $\mathcal{P}_{\mathcal{D}}$ receives the ticket from the user in order to be verified. In this verification, the current time is compared to the validity time τ_v of the entrance ticket $\mathsf{t_{in}}^*$ which is signed by $\mathcal{P}_{\mathcal{S}}$.

Result 5.3. *According to the definitions given in §5.1.1 and the Claims 5.3.1 and 5.3.2, we can assure that the protocol achieves the security requirements of non-overspending and the control of the validity time of the ticket.*

Proposition 5.4. *The proposed protocols avoid attacks made by confabulated users.*

Claim 5.4.1. *The time-based fare Collection system described in §5.2 cannot be attacked by confabulated users.*

Security Argument. The confabulated attack described in §5.3, which is based on the exchange of entrance tickets, is not applicable to time-based systems since users do not obtain any benefit from the exchange. The fare is calculated by using the entrance timestamp, so if the users exchange their tickets, the fares will be the same and one of the users will pay more than with his real ticket. For this reason, users are discouraged to exchange tickets.

Claim 5.4.2. *The distance-based fare collection system described in §5.3.1 cannot be attacked by confabulated users.*

Security Argument. In distance-based fare collection systems, an attack based on the exchange of the ticket would be profitable only in some cases. In all of these cases, the direction of the users must be different. If the distance-based fare collection system fulfils the requirements, and the directions are physically separated, then the users traveling in opposite directions would not be able to use an exchanged ticket to exit the system since the direction included in the ticket would be different from the real one from the user.

Claim 5.4.3. *The protocol presented in §5.3 can be used in all kinds of distance-based fare collection system and avoids confabulated attacks.*

Security Argument. Distance-based fare collection systems that do not fulfil the requirements need an improved protocol to avoid confabulated attacks. §5.3 includes the improved protocol with a new group signature. With the use of linkable signatures, even anonymously, the provider can assure that the user who exits the system is the one who obtained the entrance ticket. For this reason, users cannot attack the system by exchanging tickets.

Result 5.4. *According to the attack described in §5.3 and the protocols described in §5.2 and §5.3, together with the Claims 5.4.1, 5.4.2 and 5.4.3, we can assure that the protocol cannot be attacked by confabulated users, neither for time-based fare collection systems nor for distance-based fare collection systems.*

5.5 Experimental results

In order to evaluate the protocol performance, we have implemented the protocol described above. Since the protocol is intended to be used with mobile devices (handsets or in-car devices) we have implemented it in a mobile platform. There are several mobile platforms available but Android is the world leader of market share (data from the third quarter of 2012 [1]).

Its great success is attributed to the diversity of devices that include this mobile operating system, ranging from low class to high technology terminals. Furthermore, Android uses a variety of the well-known Java language for the application development. It is developed and actively supported by Google and there is also a big community of users who can provide valuable feedback, ideas and applications.

The first challenge has been the implementation of the Short Group Signature by Boneh, Boyen and Shacham.

As the scheme is based on bilinear maps and pairings, we chose to use the jPBC (Java Based Pairing Cryptography) [2] library. This library is a full Java port of PBC (Pairing Based Cryptography) [3] C library in which Shacham, one of the authors of the group signature, contributed to its development. The jPBC library provides us the ability to compute complex pairing operations over elliptic curves required by the group signature scheme.

As the group signature implementation uses pairing based cryptography, and in order to unify the development, we have decided that the randoms, exponentiations and arithmetic operations apply this kind of cryptography too. On the other hand, the common signatures and encryptions functions use the RSA algorithm, using the Bouncycastle library [4].

The implementation is split into two key parts, that is, the client side and the server side. Furthermore, it is important that the communications between each party are developed in XML format. Next, we briefly depict each side of the protocol.

- **Client side**. As we said, the user application ($\mathcal{U}$) is developed over the An-

[1] http://www.businesswire.com/news/home/20121101006891/en/Android-Marks-Fourth-Anniversary-Launch-75.0-Market

[2] http://gas.dia.unisa.it/projects/jpbc/

[3] http://crypto.stanford.edu/pbc/

[4] http://www.bouncycastle.org/

droid operating system, by using the API level 7 to accomplish compatibility between the two smartphones.

- **Server side**. This side compromises the *Group TTP* ($\mathcal{M}_G$), the *Payment TTP* ($\mathcal{M}_C$), the *Source Station* ($\mathcal{P}_S$) and the *Destination Station* ($\mathcal{P}_D$) servers. The entities are developed over Java JDK 6.0 and they have their own MySQL database to store the useful information. Then, each entity exposes its service through a server TCP port.

- **Communications**. The messages exchanged by the entities of the AFC infrastructure are serialized with XML and they are sent over a network communication. The XML format has a textual representation, it is portable and it is multiplatform, so it is suitable for our service.

5.5.1 Test scenario

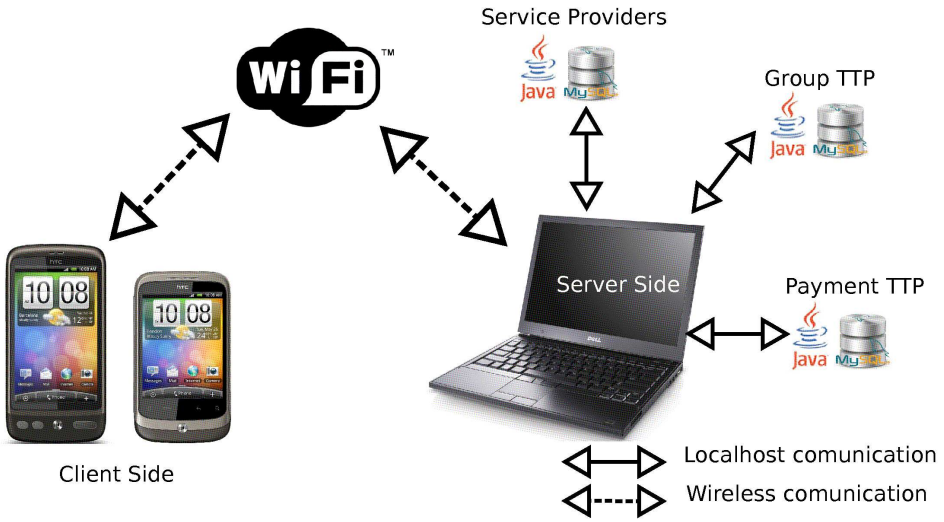

Figure 5.4: Scheme of the test scenario.

The test scenario we have used, as Fig. 5.4 shows, is composed by a notebook where the above servers are located, while the client side is tested over two Android smartphones. The first one is the HTC Desire and the other one is the HTC Wildfire. The former is a medium-high class device and the latter is a low-

medium class smartphone. The mobile phones and notebook features are listed in Table 5.11.

Device	CPU	RAM	ROM	OS
Notebook	Intel Core Duo 2 1.6GHz	4GB		Debian Linux 5.0
HTC Desire	Qualcomm Snapdragon 1GHz	576MB	512MB	Android 2.2
HTC Wildfire	Qualcomm MSM7225 528MHz	384MB	512MB	Android 2.1

Table 5.11: Technical features of test devices.

Testing the application over two types of smartphones can provide us with valuable information about the protocol performance over two phone types with different computing capabilities. It is noticeable that two different smartphones with close launch dates can have significant difference in their experimental results.

The connectivity between the Android client and the AFC is provided by a wireless 802.11g network using Java Sockets, but the connections between each infrastructure server are made over the laptop localhost network interface. Finally, the test with each device and each protocol version have been performed many times in order to get the average time for each protocol step.

5.5.2 Discussion

After the protocol implementation, we now analyze and discuss the time results for each test. We depict the benchmark results from less to more detailed. So, the first graph in Fig. 5.5 shows that the protocol execution over the HTC Desire is faster than over the HTC Wildfire, as expected. If we analyze the version of each protocol, for the time-based protocol the Wildfire spends around 113 seconds to execute the whole protocol, while Desire spends only 20 seconds, that is, 5.6 times less. Then, for the distance-based protocol the trend remains the same because over Wildfire, the modified protocol is completed after 134 seconds while the Desire only needs 26 seconds, that is, around 5.2 times less. At first glance, we can say that the difference between both protocols is not as notorious as we could expect, because whereas the Desire only expends 6 more seconds to complete compared to the time-based, the Wildfire needs 21 seconds more.

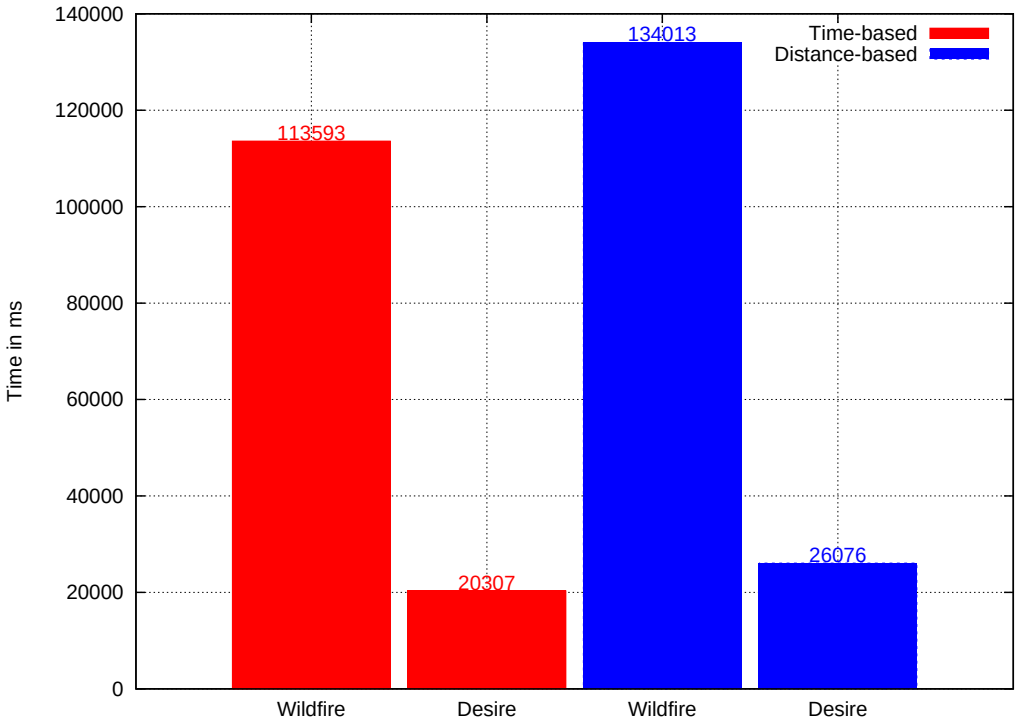

Figure 5.5: Total time expended for each protocol version over each smartphone.

Next, we are going to show a detailed study of the time spent and how we can improve the application performance for both smartphones. So, we will see the time spent in each protocol step to analyze where the application requires more computation power. In order to understand Fig. 5.6 and Fig. 5.7, we need to explain the meaning of each step in the x-axis:

- **Load Pairing**. The user application loads all the data necessary to do pairing operations, e.g., loading elliptic curve parameters from a file or preparing some Java objects.
- **Prepare RSA**. The RSA key pair is loaded from a PEM file stored in the phone data store, e.g., from the SD storage.
- **Request Pub.Param**. This is the time needed for the application to request the public parameter α needed in the next step to the Group TTP ($\mathcal{M}_\mathcal{G}$) server.
- **gTTP Reg**. This is the time used by the application to complete the registration to the Group TTP ($\mathcal{M}_\mathcal{G}$) server. It matches the protocol steps **generatePseudonym** and **keyIssue** from §5.2.4.
- **pTTP Reg**. The user registers to the Payment TTP ($\mathcal{M}_\mathcal{C}$) server. It matches the protocol steps **startingZKP**, **challengeGeneration**, **proofGeneration** and **verifyPseudonym** from §5.2.4.
- **Prepare Entrance**. This is the total precomputation time spent before the system entrance. It applies to both time-based and distance-based protocols. Here, we establish all the precomputable parameters needed to build the group signature.
- **Entrance**. This is the time spent to enter the system. It matches the protocol step from §5.2.4.
- **Prepare Exit**. This is the time spent by the exit group signature precomputation before the exit step. It applies only to the distance-based protocol version.
- **Exit**. Finally, this is the elapsed time by the system exit, which matches the protocol step from §5.2.4.

Fig. 5.6 shows the protocol time flow step by step for the time-based protocol. If we analyze the graph, we can see that the time is mostly consumed to compute the system entrance step. Therefore, the Desire device is only clearly faster on the system entrance. So, if we use precomputation before the system entrance, the performance of the protocol is similar in both devices and is only clearly greater where more computation power is needed.

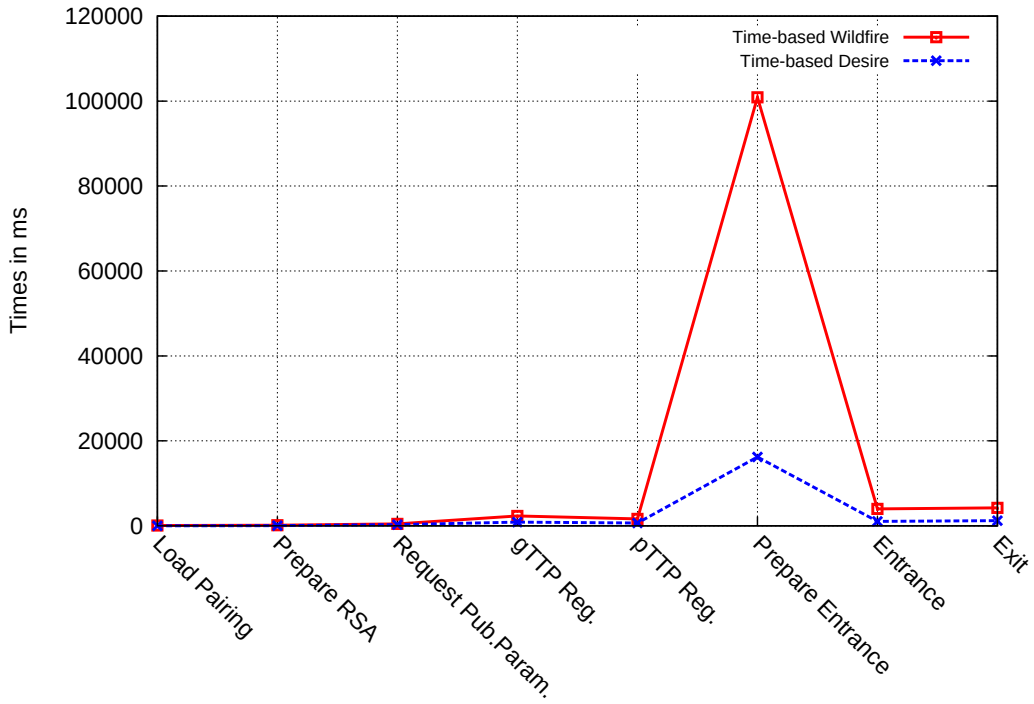

Figure 5.6: Time flow for the time-based protocol and smartphone.

Fig. 5.7 depicts similar data to Fig. 5.6 but now for the distance-based protocol. The trend is the same as Fig. 5.6 until the system entrance because previous steps are untouched. Afterwards, and before the system exit, a new prepare exit step, in which the client does more precomputations before she arrives to the exit appears. The important result is that the exit step is done by Desire in around 1.7 seconds whereas Wildfire does it in 4.7 seconds, so in this device it is 3 seconds more.

If we cross compare both protocols through each smartphone, we can see that the time needed to execute the modified version, without taking care of the precomputation time which can be computed offline by the client, is not much higher than the time spent by the original version. So we can state that the modified version increases the time cost but it remains usable for both devices.

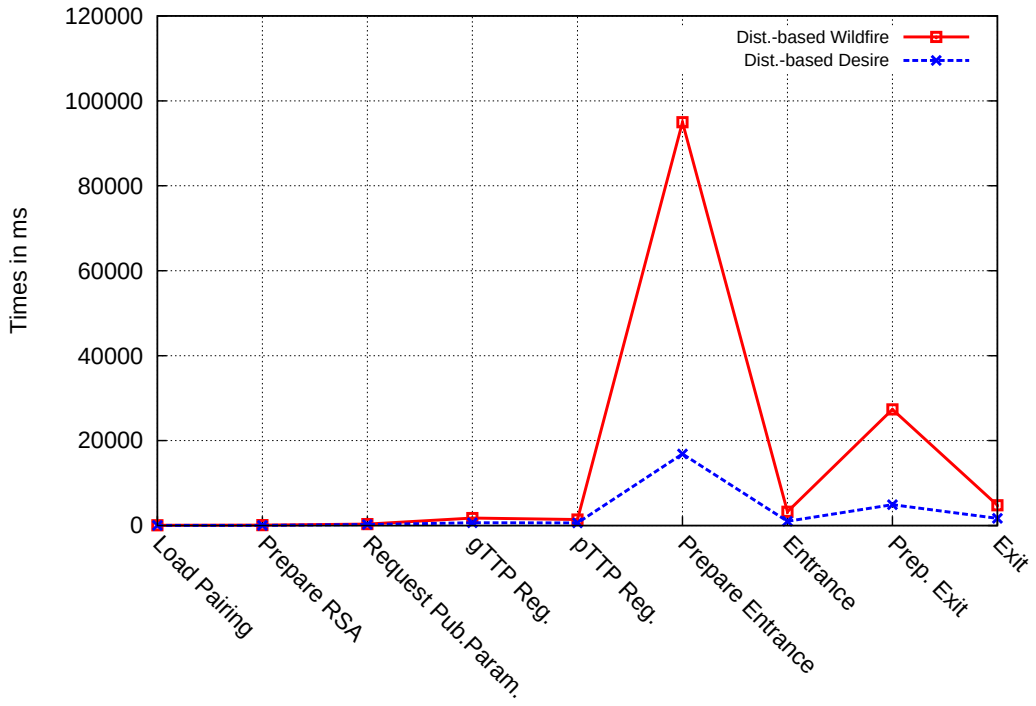

Figure 5.7: Time flow for the distance-based protocol and smartphone.

Fig. 5.8 and Fig. 5.9 will expose the different execution times of each protocol stage over both smartphones if we use precomputation or not. Now, in the x-axis we depict the protocol phases as following:

- **Init**. This phase belongs to the client application deployment time and also the time needed to request the public parameter α to the *Group TTP*.
- **Registration**. This phase adds the*Group TTP* registration and the *Payment TTP* registration times. So this is the time needed by a user to complete the system registration in order to use it.
- **Entrance**. It is the time required by the user until she receives the entrance ticket.
- **Exit**. It is the time needed by the user to leave the system until she obtains the exit ticket.
- **Pre. Comp**. This is the whole precomputation time. In the time-based protocol, the precomputation is only needed before the entrance step, whereas

in the distance-based protocol, the precomputation is done both before the entrance and the exit. Note that this stage only affects Fig. 5.9.

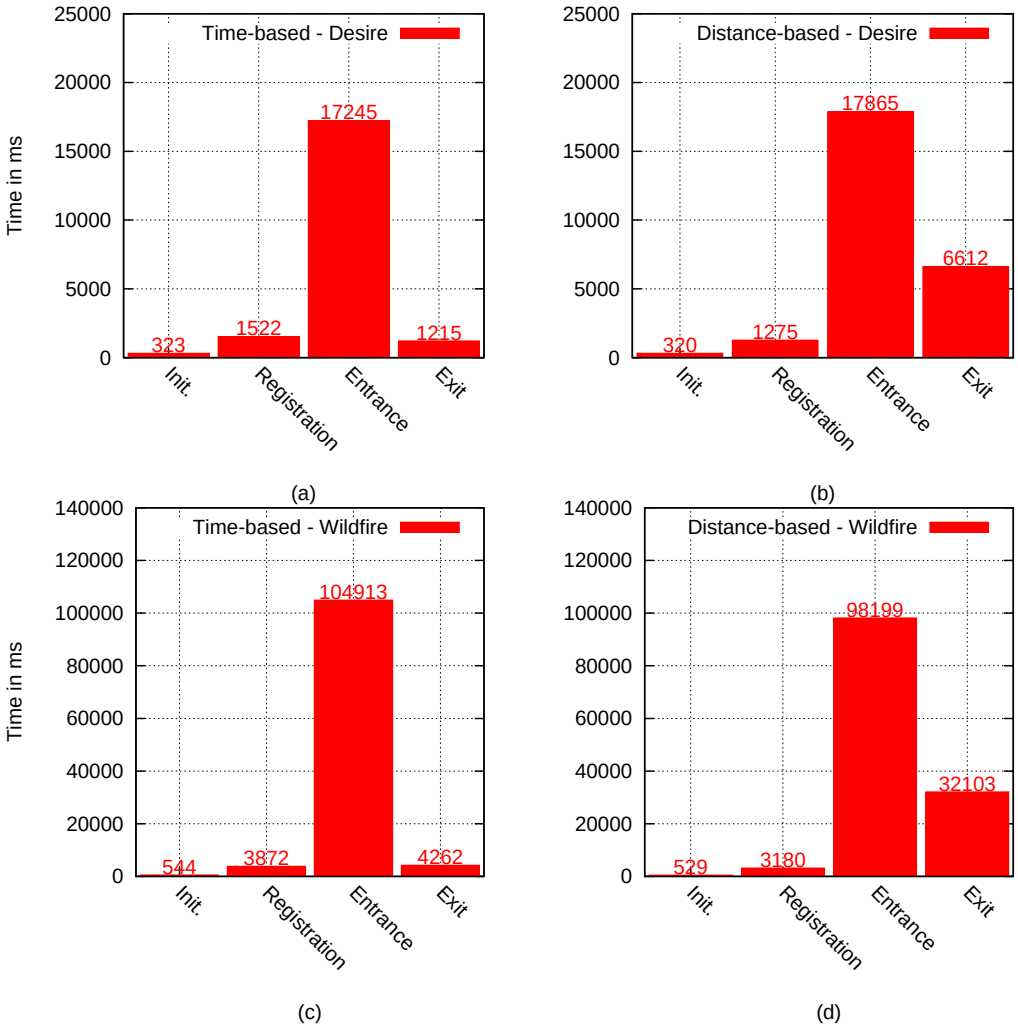

Figure 5.8: Protocol phases for each version and smartphone without precomputation.

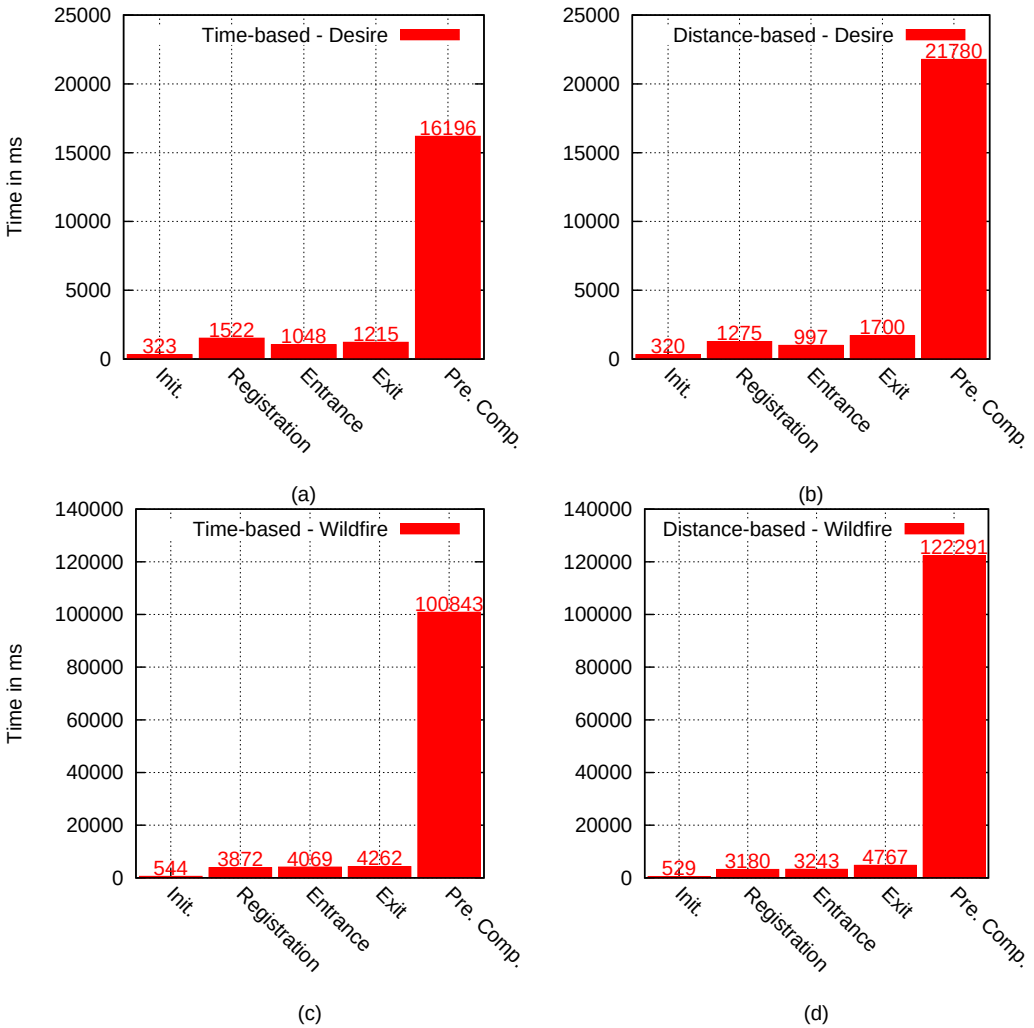

Figure 5.9: Protocol phases for each version and smartphone with precomputation.

On the one hand, Fig. 5.8 shows us that in the time-based protocol, the entrance is the bottleneck of the system. In the distance-based protocol, the entrance step consumes again a lot of time but less than the time-based protocol because an encryption is not necessary. Moreover, as a result of the security improvement in the system exit step, it is clear that this stage has taken longer to execute because more computation has been needed.

On the other hand, in Fig. 5.9, all the precomputation is taken out of the protocol and stored in the last bar. If we compare this graph set with the previous one, we can clearly see how the precomputations spent most of the time. For example, for the Wildfire smartphone, the entrance took around 100 seconds when precomputation was not applied. Instead, if precomputations are activated, the time spent is reduced to around 4.0 seconds. The same happens if we analyze the exit step using Wildfire, and the distance-based protocol, because this step took around 32 seconds, but if precomputations are applied, the time is reduced to only 4.7 seconds.

So, we can conclude the current analysis by remarking that the depicted scheme and the implementation developed is suitable to be used on AFC systems. Furthermore, the differences between time-based and distance-based protocol execution times are not large. In both cases, as we can see, the protocol can be also used with a medium class mobile device although it works faster with a high class smartphone. This suggests that in the near future, when more powerful mobile devices appear in the market, the protocol performance will be even better.

5.6 Conclusions and related publications

We have presented an Automatic Fare Collection system that is secure and preserves the privacy of users. We introduce the need of a short-term linkage of tickets from a same movement of a user, although leaving the rest of the other movements unlinkable between them.

A first version of the protocol was presented in an international conference in Paris (France), 2010. Then, the protocol was improved and extended to be usable for time- and distance-based systems, together with the experimental results in a real scenario with mobile devices (smartphones), and was published in an ISI-JCR Journal in 2012. The publications are detailed below:

- A. Vives-Guasch, J. Castellà-Roca, M.M Payeras-Capellà, M. Mut-Puigserver.

"An Electronic and Secure Automatic Fare Collection System with Revocable Anonymity for Users". In *Advances in Mobile Computing & Multimedia (MoMM), 8th International Conference, pp. 387–392, doi: 10.1145/1971519.1971585*, 2010.

- A.P. Isern-Deyà, A. Vives-Guasch, M. Mut-Puigserver, M.M Payeras-Capellà, J. Castellà-Roca. "A Secure Automatic Fare Collection System for Time-based or Distance-based Services with Revocable Anonymity for Users". *The Computer Journal, doi: 10.1093/comjnl/bxs033*, 2012.

Chapter 6

Short-Term Linkable Group Signatures with Categorized Batch Verification

This chapter presents the proposal of short-term linkable group signatures as an extension of the previous AFC proposal, thought for mass-verification scenarios where the verification times are critical, by using categorized batch verification.

Contents

This chapter is organized as follows. First, in §6.1, we make a brief introduction to the field. In §6.2, we present the related work which is focused on the security and privacy protection in Vehicular Ad-Hoc Networks (VANETs), and we also outline the contribution. In §6.3, we present a basic scheme description, requirements and main cryptographic techniques used in our proposal. Furthermore, §6.4 introduces our solution and the phases of our scheme are described, followed by a wide comparison with related solutions together with the security considerations in §6.5. Finally, conclusions and related publications are presented in §6.6.

6.1 Introduction

Data confidentiality is a requirement that has to be preserved, like data authenticity, integrity and user privacy during communication. For example, privacy is demanded by users in Wireless Body Sensor Networks (WBSN) where nodes are bound to human in order to measure medical data and user position [Sun 10]. In WBSN, users are concerned about their potential monitoring by a malicious observer. Moreover, the received messages which carry data from several tens of nodes must be verified in a short time. The same issues arise in VANETs. For better intuition, we apply our proposal to the use case of VANETs. Nevertheless, the solution can be applied to mass-verification systems where the privacy of users, data authenticity and integrity are required during dense communication. This proposal is an extension of the solution given in the Automatic Fare Collection system presented in the previous chapter. We present short-term linkability with categorised batch verification.

Wireless communications among vehicles bring many applications which can help drivers to, for example, prevent accidents or reduce traffic density. A vehicular ad-hoc network measures useful data like speed, location, road condition or alerts and distributes them using an On Board Unit (OBU) in a vehicle, in order to increase security on roads and reduce traffic jams. OBU can be an embedded device, a user smartphone or a navigation application with VANET. Self-organized VANET offers two types of communication: the wireless communication between a vehicle and a vehicle (V2V), and the communication between vehicles and the VANET infrastructure (V2I) represented by Road-Side Units (RSU), which are con-

nected to a fixed infrastructure (e.g. Internet). Security in VANETs plays a key role in the protection against bogus and malicious messages, misuse at roads, eavesdropping attacks, etc. The common solutions of digital signature guarantee the message integrity, authentication and non-repudiation. Furthermore, privacy is required due to the possibility of the tracking of drivers by malicious observers. Moreover, VANETs can serve in a dense urban traffic where hundreds of vehicles communicate in V2V or V2I, so that the security overhead and computation time must be minimal. In this case, the following scenario is considered: *Scenario 1*:

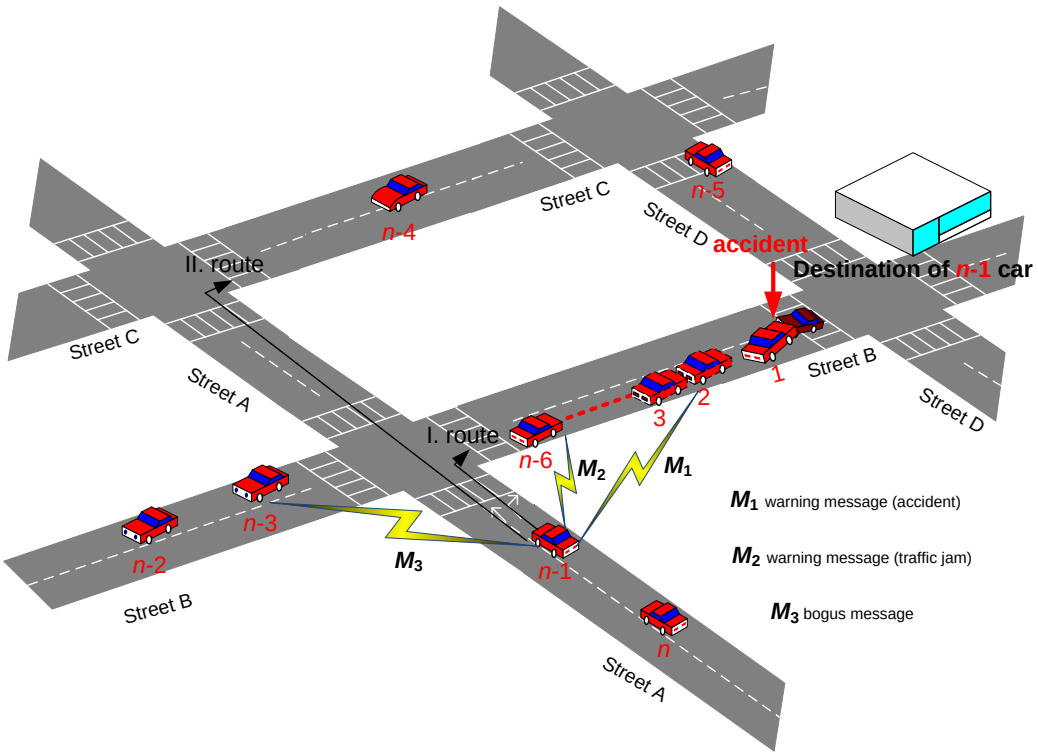

Figure 6.1: The VANETs in urban traffic - Scenario 1.

A driver, Alice (A), with the car no. 2, which is depicted in Fig.6.1, can register special events (accidents, traffic jams, roads under construction etc.). Depending on the type of event, A immediately broadcasts a warning message through the wireless V2V communication to all participating cars in VANET. In this scenario, an accident is depicted in Fig.6.1. Suppose that another driver, Bob (B), with the car no. $n-1$, who is in range and coming closer to A, receives this message. B also receives more messages from other cars in the area. Moreover, other messages can contain contradictory warnings or can be bogus. In little time, B must consider

the validity of these messages and has to quickly decide changing the route (from planned I. to II.). If *B* makes the right decision, she can avoid the situation referenced by the first warning message. It is obvious that the decision must come in real time and as soon as possible. In other cases, many cars exchange information between each other periodically (speed, direction, location, break alerts, etc.) and road conditions (change of road lanes, distance between cars, etc.). The type of information depends on the application used by the car, but message processing must be efficient because the sending period of beacon messages [1] is less than 300 ms [Huss 09].

The security proposals are challenged to connect privacy, security, efficiency and capable management in huge vehicular networks. The open problem of *Scenario 1* is how to verify a lot of anonymous messages in real time. The related work tries to solve this problem by using the batch verification of group signatures. But this approach takes more time than expected if the number of malicious messages appearing in batch is greater or equal than 15% from all the messages, as it is claimed in [Ferr 09]. In order to improve this issue, we propose a novel solution with categorized batch verification with short-term linkability, which can serve to recognize the malicious messages and exclude them from batch. Moreover, the short-term linkability significantly improves the signature phase, so that our scheme provides more efficient signature and verification than related works using group signatures.

6.2 Related work and contribution

In this section, we outline the related work and our contribution.

6.2.1 Related work

Generally, the protection of privacy in VANETs can be ensured by three approaches, i.e., pseudonyms, group signatures and hybrid schemes. Anonymization through pseudonyms has been proposed in [Gerl 07] and [Fons 07]. The work in [Raya 07] uses anonymous certificates which are stored in vehicles (usually in a tamper-proof device). This approach uses a set of short-lived pseudonyms, and privacy among vehicles is provided by changing these certified public keys. Nev-

[1] Beacon message – signal that indicates the proximity of other nodes/vehicles for communication

ertheless, in large urban VANETs, this approach is burdened by preloading and storing a large number of anonymous certificates with pseudonyms.

Group signatures (GS) in VANETs provide user anonymity by signing a message on behalf of a group. GS guarantee the unlinkability of honest users and the traceability of misbehaving users. The scheme [Lin 07] called GSIS uses the combination of a group signature based on [Bone 04b], with a hybrid membership revocation mechanism in the V2V communication, and Identity Based Group Signature (IBGS) in the V2I communication. The hybrid membership revocation with the list of revoked members or revocation list (RL) works with a threshold value T_τ. In case $|\text{RL}| < T_\tau$, the scheme uses revocation verification algorithm, otherwise, the scheme updates the public/private group keys of all non-revoked members. For efficient verification, the authors of [Zhan 08] propose a GS with batch verification in V2I, which performs three pairing operations. This scheme, which is called IBV, has several drawbacks such as using tamper-proof devices, which is vulnerable to tracking or impersonation attacks (see [Chim 11] for a complete description). The works by [Zhan 10] and [Wase 10] can efficiently verify a large number of messages in V2V. These schemes use short group signatures with fast batch verification (only 2 pairing operations are used instead of $5n$, where n is the number of messages). Nevertheless, the performance of batch verification degrades in dense V2V communication with bogus messages. The On Board Units (OBUs) must process the messages quickly, they have between 100 ms and 300 ms to process a message [Huss 09]. Thus, the computation of expensive pairing and exponentiation on limited On Board Units (OBUs) is a hard requirement to meet because of the short response time. This fact limits the VANETS in practice. The work [Qin 11] employs identity-based group signature with the batch verification, provides a scalable management of large VANETs and an efficient revocation of members, but suffers from more expensive signing and verification phases than GS.

In [Cala 07], vehicles locally generate short-lived certificates (pseudonyms) on the fly, with the help of GS. A Certification Authority (CA) maintains the mapping between identities and pseudonyms. One of the drawbacks is the security overhead of messages, which consists of the message signature by private short-lived key, public short-lived key and the group signature of public short-lived key. In [Stud 09], the solution called TACK uses short-lived keys (ECDSA) to secure V2V messages. Long-term pre-distributed keys (group signatures) are used for anony-

mous authentication in regions as well as to gain the new certified temporary key from the Regional Authorities (RAs). TACK supports desirable short-term linkability, but dense V2I communication leads to a delay in the join phase and OBU must broadcast ECDSA public key with the certificate in V2V. In [Chim 11], the two proposals called SPECS include the pseudonyms maintained by Trusted Authorities (TAs), the group signature with 2-pairing batch verification and the positive and negative bloom filter for the effectiveness of the verification phase. Nevertheless, SPECS strongly rely on TAs and Road-Side Units RSUs. Also, the communication delay plays a critical role between TAs and vehicles. In [Chen 11], the authors present a Threshold Anonymous Announcement (TAA) service based on the adaptation and mixture of direct anonymous attestation and one-time anonymous authentication. The computational cost of the signing algorithm takes only 6 scalar multiplications and 1 pairing operation, and the computational cost of the verification algorithm takes 5 scalar multiplications and 5 pairing operations. Nevertheless, the TAA scheme does not support batch verification.

6.2.2 Contribution

Similarly to [Zhan 08], [Zhan 10], [Wase 10] and [Wei 11], our proposed solution is based on group signature. We focus on the efficiency of signature/verification, security and privacy protection with respect to computationally limited RSUs. As related works, we assume OBUs to have enough computational power for basic modular arithmetic, pairing and cryptographic operations.

- In V2V communication, the solution provides the efficient signing with short-term linkability. The proposal uses the modified scheme of Wei et al. (WLZ scheme) [Wei 11]. Nevertheless, our solution adds the short-term linkability, thus obtaining a more efficient signing phase than in the WLZ scheme. Moreover, the WLZ scheme is focused on the V2V communication and does not describe the registration and join phases in detail. Finally, the short-term linkability is demanded for several applications [Stud 09] and can prevent Sybil and Denial of Service attacks.

- In V2V communication, the solution provides the efficient categorized batch verification with short-term linkability. In group signatures, the batch verification of n messages is generally more efficient than individual verification but the complexity of batch computation with bogus messages increases

from $\mathcal{O}(1)$ to $\mathcal{O}(ln\ n)$. In [Ferr 09], the authors claim that if a 15% or above of the signatures are invalid, then batch verification is not more efficient than individual verification. The proposal modifies the WLZ scheme [Wei 11], where the batch verification costs only 2 pairings and $11n$ exponentiations. But the WLZ scheme and related solutions use uncategorized batch verification which can cause less efficient verification if bogus messages appear during attacks like the Sybil attack, the Denial of Service (DoS) attack, etc. However, the solution applies categorized batch verification, which sorts potential honest messages to the first batch, and potential untrusted messages to the second or third batch with lower priorities, so the verification phase can be more efficient and prevent Sybil and DoS attacks.

- In V2I communication, the scheme uses probabilistic cryptography for keeping long-term unlinkability and the privacy protection of drivers. The join or registration phase takes only two messages (request/response) and the scheme does not need tamper-proof devices.
- We avoid the inefficient linear growth of revocation list with the secret keys of members. The proposal uses the revocation process with the expiration of timestamp in certified pseudonym, which revokes members. The proposal only uses a Group Temporary Revocation List (GTRL) broadcasted between group managers to deny malicious members accessing the group of VANET members.

6.3 Preliminaries

In this section, we outline the scheme, the requirements and the main cryptographic techniques used in the proposal.

6.3.1 Description of the scheme

The scheme, depicted in Fig.6.2, consists of a Trusted Authority ($\mathcal{T}_\mathcal{A}$), a Group Manager ($\mathcal{M}_\mathcal{G}$) and a Member ($\mathcal{V}$).

- $\mathcal{T}_\mathcal{A}$ issues certified member pseudonyms and generates all public cryptographic parameters in the solution. $\mathcal{T}_\mathcal{A}$ is a fully trusted entity and can reveal the real ID of a member in the revocation phase. $\mathcal{T}_\mathcal{A}$ is connected with all the group managers and manages the registration of all members.

- $\mathcal{M}_{\mathcal{G}}$ is an entity which manages several Road Side Units (RSUs) and generates group secret keys to members in the join phase. We assume that $\mathcal{M}_{\mathcal{G}}$ is honest and is securely connected with the own RSUs (e.g. via Transport Layer Security). $\mathcal{M}_{\mathcal{G}}$ can also trace and open the malicious messages in its own area but $\mathcal{M}_{\mathcal{G}}$ cannot reveal the member ID.

- $\mathcal{V}$ is a driver with the certified pseudonym, which is embedded in vehicle's OBU. After the registration of the driver in $\mathcal{T}_{\mathcal{A}}$ and joining in $\mathcal{M}_{\mathcal{G}}$'s area through the V2I communication, $\mathcal{V}$ can send or broadcast messages through the V2V communication. Furthermore, $\mathcal{V}$ can report a bogus message through the V2I communication to $\mathcal{M}_{\mathcal{G}}$.

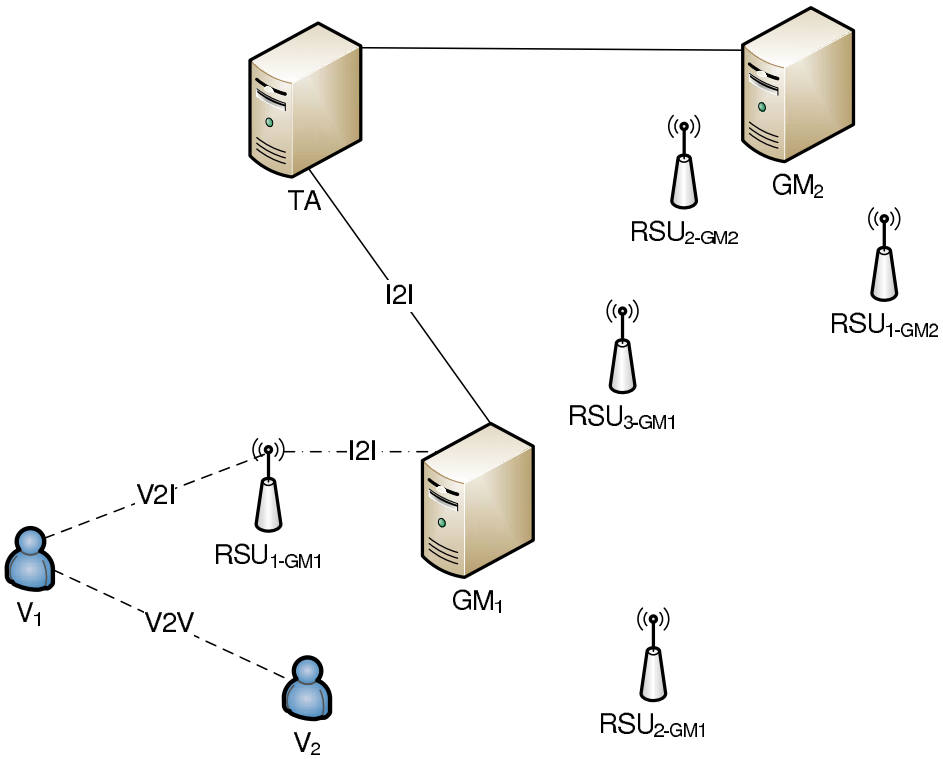

Figure 6.2: The parties in our model of secure and anonymous VANET.

6.3.2 Requirements

The scheme is designed to satisfy these security requirements:

- Message authenticity, integrity and non-repudiation (Def. 2.1, 2.2, 2.3). In

V2V communication, the group signature ensures that the message is signed by a vehicle which holds the right and fresh group key pair (authenticity). The system must verify the received messages, i.e., the messages that have not been modified once they have been sent (integrity). Members stay private but cannot deny that they created the signed messages (non-repudiation).

- Revocable Anonymity (Def. 2.7). The scheme protects privacy of the drivers in the long-term. A honest driver with OBU can use the pseudonym signed by $\mathcal{T}_{\mathcal{A}}$ to obtain group parameters and keys from $\mathcal{M}_{\mathcal{G}}$. Then, its OBU can sign every message on behalf of the group members and keep anonymity of drivers. Any possible malicious driver can be revealed by the collaboration of $\mathcal{M}_{\mathcal{G}}$ and $\mathcal{T}_{\mathcal{A}}$. If some member breaks the rules, her messages can be opened by $\mathcal{M}_{\mathcal{G}}$ and his pseudonym is sent to $\mathcal{T}_{\mathcal{A}}$, which can extract the member's ID. Next time, when an adversary requests a new pseudonym with a fresh timestamp (e.g. via IETF RFC 3161), $\mathcal{T}_{\mathcal{A}}$ checks if her ID appears in the list of globally revoked members. However, when a member misuses the VANETs for her own benefit, breaks the rules or causes an accident, $\mathcal{M}_{\mathcal{G}}$ obtains her pseudonym from her signed messages and, sends it to $\mathcal{T}_{\mathcal{A}}$, who revokes the anonymity, and obtains her ID.

- Short-term Linkability (Def. 2.9, 2.10). In several VANETs applications like the safe change of road lanes and the short-term mapping of vehicle movements, the short-term linkability is a desirable property [Stud 09]. In a short period, i.e., every $100 \div 300$ ms, the broadcasted V2V beacon messages are used to trace the position and direction of the vehicle. The current proposals which use group signatures cannot link related messages from one vehicle sent in a short interval. The scheme balances the privacy of drivers and the linkability of messages, which is available only for a short interval. On the other hand, long-term unlinkability is ensured using the probabilistic encryption and changing the pseudonyms in the group signature.

6.3.3 Cryptography background

The solution employs the ECDSA signature scheme with the public/private keys of $\mathcal{T}_{\mathcal{A}}$, $\mathcal{M}_{\mathcal{G}}$ and $\mathcal{V}$. Additionally, we use a probabilistic ElGamal encryption/decryption during the join of members. The modified short group signature WLZ scheme [Wei 11] based on the BBS04 scheme [Bone 04b] is used in the V2V

communication. This scheme uses bilinear maps and is based on q-SDH problem and Decision Linear problem, which have been studied in [Bone 04b].

We follow the notation of [Bone 04b] for the concept of bilinear maps already explained in §3.4. From that section, we remember the basic notation as follows: $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are multiplicative cyclic groups of a prime order p. Then, g_1 is a generator of $\mathbb{G}_1$, g_2 is a generator of $\mathbb{G}_2$ and ψ is an isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$ that $\psi(g_2) = g_1$. Finally, e is a computable bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, with the properties of bilinearity and non-degeneracy.

We maintain the supposition that the SDH assumption holds on $(\mathbb{G}_1, \mathbb{G}_2)$, and also that the Decision Linear assumption holds on $\mathbb{G}_1$.

6.4 Solution

We focus on the practical registration and join of VANET members and the efficient signing/verification of V2V messages. The solution consists of seven phases: Setup, Registration, Join, Signing, Categorized Verification, Trace, and Revocation.

6.4.1 Setup

In the first part, $\mathcal{T}_\mathcal{A}$ chooses parameters $(\mathbb{G}_1, \mathbb{G}_2, g_1, g_2, \psi, e)$ and generates an ECDSA key pair $sig_{\mathcal{T}_\mathcal{A}}/ver_{\mathcal{T}_\mathcal{A}}$, an ElGamal private key $sk_{\mathcal{T}_\mathcal{A}}$ and a public key $pk_{\mathcal{T}_\mathcal{A}}$, and then releases the public keys and parameters. Every $\mathcal{M}_{\mathcal{G}_i}$ generates group signature keys, ElGamal private $sk_{\mathcal{M}_{\mathcal{G}_i}}$ and public $pk_{\mathcal{M}_{\mathcal{G}_i}}$ keys for the secure V2I communication, and they publish the public keys. Every $\mathcal{M}_{\mathcal{G}_i}$ calls $KeyGen_G$ from §3.4, that is, randomly selects $\xi_1, \xi_2 \xleftarrow{R} \mathbb{Z}_p^*$, $h \xleftarrow{R} \mathbb{G}_1^*$ and sets u, v such that $u^{\xi_1} = v^{\xi_2} = h$. Then, $\mathcal{M}_{\mathcal{G}_i}$ selects random $\gamma \xleftarrow{R} \mathbb{Z}_p^*$ and computes $w \leftarrow g_2^\gamma$. The group public key is $gpk = (g_1, g_2, u, v, w, h)$ and the group manager secret key is $gmsk = (\xi_1, \xi_2)$.

6.4.2 Registration

In the registration phase, the i-th driver (member) $\mathcal{V}_i$ using a vehicle with OBU requests a valid certified pseudonym $\pi_{\mathcal{V}_i}$ from $\mathcal{T}_\mathcal{A}$. For the first time, $\mathcal{T}_\mathcal{A}$ must physically verify the driver's real ID, her driving license and OBU's ID number. Then, $\mathcal{V}_i$ creates an ECDSA key pair $sig_{\mathcal{V}_i}/ver_{\mathcal{V}_i}$, gives the public key to $\mathcal{T}_\mathcal{A}$, which stores $(ID_{\mathcal{V}_i}, ver_{\mathcal{V}_i})$ in the database, and the signed certificate $cer_{\mathcal{V}_i} = sig_{\mathcal{T}_\mathcal{A}}(ID_{\mathcal{V}_i}, ver_{\mathcal{V}_i})$ is given to $\mathcal{V}_i$. After the first successful registration phase, the driver can request her

next pseudonym online. Assuming that $\mathcal{V}_i$ has $pk_{\mathcal{T}_\mathcal{A}}, ver_{\mathcal{T}_\mathcal{A}}$, the two-message of the registration phase consists of these steps:

1. $\mathcal{V}_i$ self-generates ElGamal key pair $(sk_{\mathcal{V}_i}/pk_{\mathcal{V}_i})$ and sends the encrypted request $enc_{pk_{\mathcal{T}_\mathcal{A}}}(pk_{\mathcal{V}_i}\|ID_{\mathcal{V}_i}\|ver_{\mathcal{V}_i}\|cer_{\mathcal{V}_i}\|sig_{\mathcal{V}_i}(pk_{\mathcal{V}_i}\|ver_{\mathcal{V}_i}\|ID_{\mathcal{V}_i}))$ to $\mathcal{T}_\mathcal{A}$.

2. $\mathcal{T}_\mathcal{A}$ decrypts the request and checks if the $ID_{\mathcal{V}_i}$ is not revoked in Global Revocation List (GRL), the certificate $cer_{\mathcal{V}_i}$ and the member's signature, which ensures member's authenticity, and commits the $pk_{\mathcal{V}_i}$ in the certificate with new ElGamal key pair. Then, $\mathcal{T}_\mathcal{A}$ generates a challenge $c \xleftarrow{R} \mathbb{Z}_q$, a timestamp T_l and sends the encrypted response $enc_{pk_{\mathcal{V}_i}}(enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|T_l\|sig_{\mathcal{T}_\mathcal{A}}(T_l\|enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|pk_{\mathcal{V}_i}))$ back to $\mathcal{V}_i$. Finally, $\mathcal{V}_i$ checks the signature by $\mathcal{T}_\mathcal{A}$ and composes the pseudonym $\pi_{\mathcal{V}_i} \leftarrow pk_{\mathcal{V}_i}\|enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|T_l\|sig_{\mathcal{T}_\mathcal{A}}(T_l\|enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|pk_{\mathcal{V}_i})$ and stores it.

6.4.3 Join

A vehicle entering the i-th $\mathcal{M}_{\mathcal{G}_i}$ area (several RSUs) for the first time, requests the group public key and its group member secret key. We assume that RSUs managed by $\mathcal{M}_{\mathcal{G}_i}$ are securely connected through the VANET infrastructure. As a remember, $H()$ is a hash function, and the two-message join phase consists of these steps:

1. $\mathcal{V}_i$ sends $\pi_{\mathcal{V}_i} = pk_{\mathcal{V}_i}\|enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|T_l\|sig_{\mathcal{T}_\mathcal{A}}(T_l\|enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|pk_{\mathcal{V}_i})$, which is encrypted using $pk_{\mathcal{M}_{\mathcal{G}_i}}$, to $\mathcal{M}_{\mathcal{G}_i}$.

2. $\mathcal{M}_{\mathcal{G}_i}$ decrypts $\pi_{\mathcal{V}_i}$ using $sk_{\mathcal{M}_{\mathcal{G}_i}}$, verifies $\pi_{\mathcal{V}_i}$, which is signed by $\mathcal{T}_\mathcal{A}$, and controls if $enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)$ is not in the Group Temporary Revocation List (GTRL) and the validity of the timestamp T_l. If $\pi_{\mathcal{V}_i}$ is ok, $\mathcal{M}_\mathcal{G}$ creates $gsk_{\mathcal{V}_i} \leftarrow (A_i, x_i)$, where $x_i = H(enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c)\|T_l\|\gamma)$, and $A_i = g_1^{\frac{1}{x_i+\gamma}}$, and stores $(enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID\|ver_{\mathcal{V}_i}\|c), A_i, T_l))$ to the join table and sends $gsk_{\mathcal{V}_i}$ encrypted using $pk_{\mathcal{V}_i}$ to $\mathcal{V}_i$.

We note that ElGamal encryption/decryption is probabilistic. Due to this fact, an observer cannot link two or more encrypted messages if $\mathcal{V}_i$ requests $gsk_{\mathcal{V}_i}$ for the second time.

6.4.4 Signature

The signature phase applies the modified short group signature WLZ scheme [Wei 11], which is based on the BBS scheme [Bone 04b], that is, it calls the procedure $Sign_G$ from §3.4. We include a counter k in the OBUs, a member secret key $gsk_{\mathcal{V}_i} = (A_i, x_i)$ and a group public key $gpk = (g_1, g_2, h, u, v, w)$. An OBU with secret key $gsk_{\mathcal{V}_i} = (A, x)$ signs a message $M \in \{0,1\}^*$ and outputs the signature of knowledge $\sigma = (T_1, T_2, T_3, R_2, R_3, R_5, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$.
If $k = 0$, $\mathcal{V}_i$ generates $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p^*$, and
computes

$$\begin{gathered} T_1 \leftarrow u^\alpha, \qquad T_2 \leftarrow v^\beta, \qquad T_3 \leftarrow Ah^{\alpha+\beta}, \\ \delta \leftarrow \alpha x, \qquad \mu \leftarrow \beta x. \end{gathered} \tag{6.1}$$

$$p_1 \leftarrow e(T_3, g_2), \qquad p_2 \leftarrow e(h, w), \qquad p_3 \leftarrow e(h, g_2). \tag{6.2}$$

stores $T_1, T_2, T_3, \delta, \mu, p_1, p_2, p_3$, generates $r_\alpha, r_\beta, r_x, r_\delta, r_\mu \xleftarrow{R} \mathbb{Z}_p^*$, and
computes

$$\begin{gathered} R_1 \leftarrow u^{r_\alpha}, \qquad R_2 \leftarrow v^{r_\beta}, \qquad R_3 \leftarrow p_1^{r_x} \cdot p_2^{-r_\alpha - r_\beta} \cdot p_3^{-r_\delta - r_\mu}, \\ R_4 \leftarrow T_1^{r_x} u^{-r_\delta}, \qquad R_5 \leftarrow T_2^{r_x} v^{-r_\mu}, \end{gathered} \tag{6.3}$$

calculates a self-made challenge

$$c \leftarrow H(M, T_1, T_2, T_3, R_1, R_2, R_3, R_4, R_5), \tag{6.4}$$

and calculates its corresponding response

$$\begin{gathered} s_\alpha \leftarrow r_\alpha + c\alpha, \qquad s_\beta \leftarrow r_\beta + c\beta, \qquad s_x \leftarrow r_x + cx, \\ s_\delta \leftarrow r_\delta + c\delta, \qquad s_\mu \leftarrow r_\mu + c\mu. \end{gathered} \tag{6.5}$$

Finally, $\mathcal{V}_i$ sends the message M with the signature $\sigma = (T_1, T_2, T_3, R_2, R_3, R_5, c, s_\alpha, s_\beta, s_x, s_\delta, s_\mu)$ and increases the counter k++.

If α and β are unchanged every n messages, the short-term linkability is kept because the pseudonyms of group signature T_1, T_2, T_3 are also unchanged. Thus, for n messages, when $1 \leq k \leq n$, $\mathcal{V}_i$ does not need to compute equations 6.1, 6.2, contrary to the WLZ scheme, but only generates random $r_\alpha, r_\beta, r_x, r_\delta, r_\mu \xleftarrow{R} \mathbb{Z}_p^*$ and computes equations 6.3, 6.4 and 6.5. This reduces the 3 bilinear operations to 0, 10 exponentiations to 9 and 14 multiplications to 9. The concrete VANET application can decide when to fix the counter $k = 0$, and $\mathcal{V}_i$ generates new α and β and recomputes the equations 6.1 and 6.2.

6.4.5 Categorized verification

Our solution uses a categorized verification which sorts the incoming signed messages to three levels of credibility. Due to the short-term linkability, $\mathcal{V}_i$ can keep the Temporary List (TL) of known vehicles. Firstly, the received message M_j is checked by $\mathcal{V}_i$ by verifying if it contains a valid timestamp and consistent data. After that, the message with the group signature containing T_3 is verified regarding if T_3 is in TL. If this assumption is true, the recorded T_3 with previous validity $(W = 1)$ is included and sorted in the first batch. The validity W can be a boolean value which indicates valid $(W = 1)$ or invalid (and unknown, $W = 0$) signatures. Otherwise, the signed message with unknown T_3 is sorted to the second batch, which is verified after the first batch verification. The rest of signed messages with T_3 linked with $W = 0$ is verified in the third batch at the end of verification, if OBU has enough time for this. This approach improves the efficiency of the batch verification process and helps when an attacker, who is out of the group, generates unsigned or corrupted messages.

Batch verification

Batch verification is investigated in [Ferr 09], and it verifies n messages in one batch. $\mathcal{V}_i$ uses $gpk = (g_1, g_2, h, u, v, w)$ to verify messages $\sigma_j = (T_{1_j}, T_{2_j}, T_{3_j}, R_{2_j}, R_{3_j}, R_{5_j}, c_j, s_{\alpha_j}, s_{\beta_j}, s_{x_j}, s_{\delta_j}, s_{\mu_j})$, $\forall j \in \{1..n\}$, and performs the following actions:

1. restores $\tilde{R}_{1_j} \leftarrow u^{s_{\alpha_j}} T_{1_j}^{-c_j}$, and $\tilde{R}_{4_j} \leftarrow u^{-s_{\delta_j}} T_{1_j}^{s_{x_j}}$,
2. computes a new control hash c'_j from received parameters:

$$c'_j \leftarrow H(M_j, T_{1_j}, T_{2_j}, T_{3_j}, \tilde{R}_{1_j}, R_{2_j}, R_{3_j}, \tilde{R}_{4_j}, R_{5_j})$$

3. checks if $c'_j \stackrel{?}{=} c_j$. If true, $\mathcal{V}_i$ then continues with verification. Otherwise, the message with the signature is inconsistent and is therefore refused.
4. $\mathcal{V}_i$ randomly selects $\theta_1, \theta_2, ..., \theta_n \stackrel{R}{\leftarrow} \mathbb{Z}_p$ with l_b bit,
5. checks batch if

$$\prod_{j=1}^{n} R_{3_j}^{\theta_j} \stackrel{?}{=} e(\prod_{j=1}^{n} (T_{3_j}^{s_{x_j}} h^{-s_{\delta_j} - s_{\mu_j}} g_1^{-c_j})^{\theta_j}, g_2) \cdot e(\prod_{j=1}^{n} (T_{3_j}^{c_j} h^{-s_{\alpha_j} - s_{\beta_j}})^{\theta_j}, w) \qquad (6.6)$$

6. and if

$$1_{G_1} \stackrel{?}{=} (R_{5_j} R_{2_j})^{-\theta_j} T_{2_j}^{(s_{x_j} - c_j x_j)\theta_j} v^{(s_{\beta_j} - s_{\mu_j})\theta_j}. \tag{6.7}$$

The signed message is valid if equations 6.6 and 6.7 hold. All T_3s from new valid signed messages are added to TL with $W = 1$. In case that the batch verification fails, the divide-and-conquer approach is used to identify the invalid signatures that were added to TL with $W = 0$. The honest messages keep the mark $W = 1$.

Individual verification

At the end of the divide-and-conquer approach, the last two messages are individually verified. The procedure $Verify_G$ for standard verification is called from §3.4.

$\mathcal{V}_i$ restores $\tilde{R}_1 \leftarrow u^{s_\alpha} T_1^{-c}$ and $\tilde{R}_4 \leftarrow u^{-s_\delta} T_1^{s_x}$, computes new control hash c' from received parameters:

$c' \leftarrow H(M, T_1, T_2, T_3, \tilde{R}_1, R_2, R_3, \tilde{R}_4, R_5)$.

and checks if $c' \stackrel{?}{=} c$. If so, then $\mathcal{V}_i$ continues with the verification. Otherwise, the message is inconsistent and it is therefore refused.

Then, $\mathcal{V}_i$ checks if

$$\begin{aligned} R_3 \stackrel{?}{=} e(T_3, g_2)^{s_x} \cdot e(h, w)^{(-s_\alpha - s_\beta)} \cdot e(h, g_2)^{(-s_\delta - s_\mu)} \cdot \\ \cdot (e(T_3, w) \cdot e(g_1, g_2)^{-1})^c \end{aligned} \tag{6.8}$$

and

$$1_{G_1} \stackrel{?}{=} (R_5 R_2)^{-1} \cdot T_2^{(s_x - cx)} v^{(s_\beta - s_\mu)}. \tag{6.9}$$

The signed message is valid if equations 6.8 and 6.9 hold.

We can see from equations 6.6 and 6.8 that individual verification has a cost of 5 pairing operations per one message but batch verification costs only 2 pairing operations per n messages. This is the main reason why we propose to use the categorized batch verification instead of individual verification.

6.4.6 Trace

Every bogus signed message can be opened by $\mathcal{M}_{\mathcal{G}_i}$, using the group manager secret key $gmsk = (\xi_1, \xi_2)$. $\mathcal{M}_{\mathcal{G}_i}$ extracts the part of the member secret group key $gsk_{\mathcal{V}_i} \rightarrow A_i = T_3 / (T_1^{\xi_1} \cdot T_2^{\xi_2})$ and searches the record $(enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID \| ver_{\mathcal{V}_i} \| c), A_i, T_l)$

in the database. This is achieved by calling $Open_G$ from §3.4. The part of the member pseudonym can be sent to $\mathcal{T}_\mathcal{A}$ for revocation.

6.4.7 Revocation

When there are serious circumstances, e.g., an accident, a malicious member is revoked globally by the cooperation of $\mathcal{M}_{\mathcal{G}i}$ and $\mathcal{T}_\mathcal{A}$. $\mathcal{M}_{\mathcal{G}i}$ is able to open a message and extract the member pseudonym that is sent to $\mathcal{T}_\mathcal{A}$. $\mathcal{T}_\mathcal{A}$ broadcasts $rev = (enc_{pk_{\mathcal{T}_\mathcal{A}}}(ID \| ver_{\mathcal{V}_i} \| c), T_l) \| sig_{\mathcal{T}_\mathcal{A}}(rev)$ to other active $\mathcal{M}_{\mathcal{G}i}$ which check the signature and store *rev* to own GTRLs until the lifetime of this pseudonym expires. $\mathcal{T}_\mathcal{A}$ extracts $ID_{\mathcal{V}_i}$ and adds it to GRL so the malicious member cannot refresh her pseudonym in the following registration phase.

6.5 Performance and security considerations

In this section, we outline the evaluation of our solution, the comparison of the signing and verification phases with the related works which are based on group signatures, and the security and privacy considerations of our solution.

6.5.1 Performance and comparison with related work

We compare our proposal based on the BBS04 scheme [Bone 04b] with the related VANETs schemes which use group signatures, the scheme of Wei et al. (WLZ scheme) [Wei 11], GSIS [Lin 07], Zhang et al. [Zhan 10], and Ferrara et al. [Ferr 09]. In the comparison, we omit the WS2010 scheme [Wase 10] due to the problem of message signing that is pointed out in [Wei 11]. The verification of the TAA scheme [Chen 11] takes 5 scalar multiplications and 5 pairing operations but the TAA scheme does not support batch verification.

Generally, the time of bilinear pairing τ_p is considered the most expensive operation (tens times more expensive than exponentiation operation τ_e) and exponentiation is more expensive than multiplication τ_m. Nevertheless, the actual processing time also depends on the input size to those operations. Due to the fact that related works are also based on the BBS04 scheme [Bone 04b], we assume the same lengths of parameters (the MNT curves with $\mathbb{G}_1$ = 176 bits, $\mathbb{G}_T$ = 528 bits and $\mathbb{Z}_p$ = 162 bits). The work [Mali 11] shows that the modular arithmetic operations such as addition and subtraction can be computed more efficiently than

Table 6.1: The comparison of the verification phases.

V2V scheme:	Our scheme & WLZ scheme [Wei 11]	GSIS [Lin 07]	Zhang et al. [Zhan 10]	Ferrara et al. [Ferr 09]
Batch:	yes	no	yes	yes
Length of signature:	$5\mathbb{G}_1, \mathbb{G}_T, 5\mathbb{Z}_p$ (2380 bits)	$3\mathbb{G}_1, 6\mathbb{Z}_p$ (1500 bits)	$7\mathbb{G}_1, \mathbb{G}_T, 5\mathbb{Z}_p$ (2570 bits)	$3\mathbb{G}_1, \mathbb{G}_T, 6\mathbb{Z}_p$ (2032 bits)
Performance of batch verification				
Pairings	**2**	$5n$	2	2
Exponentiation	**11n**	$12n$	$14n$	$13n$
Multiplication	$11n + 1$	**8n**	$17n$	$10n + 1$
Performance of individual verification				
Pairings	5	5	5	5
Exponentiation	**10**	12	12	12
Multiplication	9	8	8	8

multiplication and exponentiation. Due to this fact, we omit these fast operations in this performance evaluation.

The proposal, based on the group signature BBS04 scheme [Bone 04b] and motivated by Wei et al. (WLZ) [Wei 11], reaches more efficient batch verification ($2\ \tau_p + 11n\ \tau_e$), where n is the number of messages, and individual verification ($5\ \tau_p + 10\ \tau_e$), than the compared schemes (see Table 6.1). However, the related solutions like Zhang et al. [Zhan 10], Ferrara et al. [Ferr 09], the WS2010 scheme [Wase 10] and also the WLZ scheme [Wei 11], use uncategorized batch verification that can be negatively affected by malicious and bogus messages (equal or greater than 15% from all messages). To our best knowledge, our proposal applies categorized batch verification with short-term linkability in VANET for the first time. The categorized batch verification with the temporary list of known vehicles reaches the high correctness of the important first batch in case that the bogus or damaged signed messages appear in the V2V communication.

As we can see in Table 6.2, the proposal significantly improves the performance of the signing of several messages with short-term linkability, and it requires less

Table 6.2: The comparison of the signing phases.

V2V scheme:	Our scheme	WLZ scheme [Wei 11]	GSIS [Lin 07] & Zhang et al. [Zhan 10] & Ferrara et al. [Ferr 09]
Short-term linkability:	yes	no	no
Performance of signing for the first message / the next messages			
Pairings	3 / **0**	3 / 3	3 / 3
Exponentiation	12 / **9**	10 / 10	12 / 12
Multiplication	12 / **9**	14 / 14	12 / 12

operations than in the signing phase of the WLZ scheme. Pairing ($3 \Rightarrow 0$), exponentiations ($10 \Rightarrow 9$) and multiplication ($14 \Rightarrow 9$) operations are reduced.

The scheme has been implemented as a proof of concept in Java and uses the Java Pairing Based Cryptography (jPBC) Library [2]. The implementation is tested on a machine with Intel(R) Xeon(R) CPU X3440 @ 2.53GHz, 4 GB Ram, Windows 7 Professional. The signing phase of the scheme with short linkability takes approx. 60 ms per one signature and the signing phase of the related schemes [Ferr 09], [Lin 07], [Zhan 10] and [Wei 11], based on BBS scheme, takes approx. 160 ms per one signature. The verification of a single signature takes approx. 207 ms using our scheme and approx. 224 ms using related schemes. If the batch verification is employed, the verification of one signature then takes approx. 50 ms, so the batch verification of 10 signatures takes approx 500 ms.

6.5.2 Security and privacy considerations

In this section, we outline the security and privacy considerations of the proposal, which is based on the cryptographic primitives which are secure and widely accepted.

Proposition 6.1. *In the registration phase between $\mathcal{V}_i$ and honest $\mathcal{T}_\mathcal{A}$, the scheme preserves message confidentiality, integrity and authenticity.*

[2](available on http://gas.dia.unisa.it/projects/jpbc/index.html)

Claim 6.1.1. *The request and response messages are confidential.*

Security Argument. We suppose that breaking the security of the ElGamal encryption is at least as hard as the decision Diffie-Hellman problem, as is proven in [Tsio 98]. Then, the registration phase keeps confidential communication between $\mathcal{V}_i$ and $\mathcal{T}_\mathcal{A}$ due to the encryption of every message between $enc_{pk_{\mathcal{T}_\mathcal{A}}}$ and $enc_{pk_{\mathcal{V}_i}}$. Only the holder of the ElGamal private key $sk_{\mathcal{T}_\mathcal{A}}$ and $sk_{\mathcal{V}_i}$, respectively, can decrypt the message.

Claim 6.1.2. *The request message is authentic and cannot be modified by an unauthorized entity.*

Security Argument. Message integrity and authenticity are ensured by the ECDSA signature scheme. The request message is unforgeable due to the commitment of the member public key $pk_{\mathcal{V}_i}$ in the member's certificate and in the signed part of request, by $\mathcal{V}_i$ using ECDSA signature key $sig_{\mathcal{V}_i}$. Assumining that the ECDSA signature scheme is secure under the Elliptic Curve Discrete Logarithm Problem (ECDLP), and that the used hash function is preimage resistant and collision resistant, the verification by the stored ECDSA key $ver_{\mathcal{V}_i}$ of the signature would then be incorrect if the request message was modified.

Claim 6.1.3. *The creation of a fraudulent pseudonym is computationally unfeasible nowadays.*

Security Argument. If an unathorized entity wants to create a pseudonym $\pi_{\mathcal{V}_i}$, it needs the ECDSA private key $sig_{\mathcal{T}_\mathcal{A}}$ of $\mathcal{T}_\mathcal{A}$. Supposing that ECDSA is secure nowadays, only trusted $\mathcal{T}_\mathcal{A}$ with its private ECDSA key $sig_{\mathcal{T}_\mathcal{A}}$ can sign $\pi_{\mathcal{V}_i}$. Moreover, if a fraudulent $\pi_{\mathcal{V}_i}$ was sent to $\mathcal{V}_i$, having $\mathcal{T}_\mathcal{A}$'s public ECDSA key $ver_{\mathcal{T}_\mathcal{A}}$, the signature of $\pi_{\mathcal{V}_i}$ would then be invalid.

Result 6.1. *According to the definitions given in §6.3.2 and the Claims 6.1.1, 6.1.2 and 6.1.3, we can assure that the protocol achieves the security requirements of confidentiality, integrity and authenticity for the registration phase.*

Proposition 6.2. *In the join phase between members ($\mathcal{V}_i$) and honest Group Managers $\mathcal{M}_{\mathcal{G}i}$, the proposed scheme preserves message confidentiality, integrity, authenticity and member's privacy.*

Claim 6.2.1. *The request and response messages are confidential.*

Security Argument. Every $\mathcal{V}_i$ who wants to join a group maintained by $\mathcal{M}_{\mathcal{G}i}$, must send the ciphertext ($pk_{\mathcal{V}_i}$ and $\pi_{\mathcal{V}_i}$) encrypted by using the certified ElGamal public key $pk_{\mathcal{M}_{\mathcal{G}_i}}$ to $\mathcal{M}_{\mathcal{G}i}$. $\mathcal{M}_{\mathcal{G}i}$ decrypts and checks if $\pi_{\mathcal{V}_i}$ is valid and sends $gsk_{\mathcal{V}_i}$ encrypted using $pk_{\mathcal{V}_i}$. Only $\mathcal{V}_i$ knows the corresponding ElGamal private key and can decrypt the message with $gsk_{\mathcal{V}_i}$. Assuming that $\mathcal{M}_{\mathcal{G}i}$ is honest, the members joining keep the message confidentiality, integrity and authenticity due to the ElGamal properties.

Claim 6.2.2. *The pseudonym $\pi_{\mathcal{V}_i}$ is anonymous.*

Security Argument. Assuming that ElGamal encryption/decryption is probabilistic, an observer is unable to link two or more request/response messages because ciphertexts are different although $\pi_{\mathcal{V}_i}$ is used several times. The pseudonym $\pi_{\mathcal{V}_i}$, created by $\mathcal{T}_{\mathcal{A}}$, does not contain the plaintext of the user identity (ID) but contains the encrypted fragment $enc_{pk_{\mathcal{T}_{\mathcal{A}}}}(ID)$. $\mathcal{M}_{\mathcal{G}i}$ and other entities are not able to open the member's ID without the private ElGamal key $sk_{\mathcal{T}_{\mathcal{A}}}$. Hence, the privacy protection of members is ensured in the join phase.

Result 6.2. *According to the definitions given in §6.3.2 and the Claims 6.2.1 and 6.2.2, we can assure that the protocol achieves the security requirements of message confidentiality, integrity, authenticity and member's privacy.*

Proposition 6.3. *In the V2V communication between $\mathcal{V}_i$, the proposed scheme ensures message integrity, authenticity, and anonymity for members although being revoked in case of misbehaviour.*

Claim 6.3.1. *Group signatures of messages keep integrity, authenticity and non-repudiation.*

Security Argument. The signing and verification phases employ the group signature with the short-term linkability to ensure the message authenticity and integrity, the driver anonymity in long-term way and non-repudiation. Our scheme modifies the WLZ scheme [Wei 11] based on the BBS04 scheme [Bone 04b], and inherits all its security features, including the correctness. Besides honest $\mathcal{M}_{\mathcal{G}i}$, only a valid group member $\mathcal{V}_i$ can sign a message on behalf of the group. If an attacker without valid $gsk_{\mathcal{V}_i} = (A_i, x_i)$ tries to modify the message, she must recompute hash c and some signature parts. Assuming that the hash function is secure and the Discrete Logarithm problem holds, the computation of the proof of

knowledge $(s_{\alpha_j}, s_{\beta_j}, s_{x_j}, s_{\delta_j}, s_{\mu_j})$ without x_i is then unfeasible nowadays. If the proof of knowledge $(s_{\alpha_j}, s_{\beta_j}, s_{x_j}, s_{\delta_j}, s_{\mu_j})$ is incorrectly computed, then the equations 6.6, 6.7 and 6.8, 6.9 would not hold. The complete formal analysis can be found in [Bone 04b].

Claim 6.3.2. *Drivers are anonymous, untraceable by the all entities besides honest $\mathcal{T}_\mathcal{A}$ and their anonymity is revocable with the collaboration of $\mathcal{M}_\mathcal{G}$ and $\mathcal{T}_\mathcal{A}$.*

Security Argument. The group signatures contain the pseudonyms T_1, T_2, T_3 of the group members, which are a linear encryption of members' secret key A_i and the random values α and β. The short-term linkability of messages does not violate the privacy of drivers. When the counter k is set to 0 and $\mathcal{V}_i$ generates a new α an β, the new signatures are then unlinkable to the old ones because new pseudonyms T_1, T_2, T_3 are generated. Supposing that the Strong Diffie-Hellman assumption holds, every correct message of a malicious member can be only opened by $\mathcal{M}_\mathcal{G}$ with $gmsk = (r_1, r_2)$, and then $gsk_{\mathcal{V}_i} = (A_i, x_i)$ can be extracted. Malicious members can be revoked with the collaboration of both $\mathcal{T}_\mathcal{A}$ and $\mathcal{M}_\mathcal{G}$.

Result 6.3. *According to Claims 6.3.1 and 6.3.2, we can assure that the protocol achieves message integrity, authenticity, and anonymity for members although being revoked in case of misbehaviour.*

Proposition 6.4. *The proposed signature scheme prevents DoS attacks, Sybil attacks and replay attacks.*

Claim 6.4.1. *The categorized verification prevents DoS and Sybil attacks.*

Security Argument. If a malicious driver Eve (E) starts the Sybil attack (a special type of DoS attack), then she broadcasts bogus messages that contain fake pseudonyms and signatures. Meanwhile, the honest drivers $(C, D, F, ...)$ send messages that contain valid pseudonyms and signatures announcing an accident (sent by D) or a traffic jam (sent by C). If existing solutions are used, E can flood the uncategorized batch verification process and paralyze drivers who must discard some messages. Our proposal implements categorized batch verification. Driver Bob (B) has a Temporary List (TL) of honest drivers. We suppose that Bob's TL keeps the list of known and honest drivers like $D, F, ...$ using the property of short-term linkability, which keeps the pseudonym T_3 unchanged for a short time. If B receives all the messages, he checks the TL and collects the messages containing

known T_3 to the first batch and verifies them. Therefore, the warning message referencing the accident from driver D is verified in time. The messages with unknown pseudonyms like C are collected in the second batch. The potentially untrusted messages from E with validity $W = 0$ are verified in the third batch only if Bob's OBU has free time and computational capacity. If Eve tries to replay recent a valid pseudonyms together with false signatures, then the recomputed hash c'_j is not equal to received hash c_j due to timestamps in messages. For this reason, Eve is not able to mount a successful DoS attack against the batch verification of signatures.

Claim 6.4.2. *The proposed signature scheme prevents replay attacks.*

Security Argument. Every message M contains the position, speed, etc. as well as the current timestamp. Before verification, every received message is checked in order to verify that the timestamp is actual and valid. If an attacker without valid $gsk_{\mathcal{V}_i} = (A_i, x_i)$ wants to reply an old message with valid signature of a user, she must modify the timestamp to a valid and actual one, then recomputes hash c_j, and recomputes all parts $s_{\alpha_j}, s_{\beta_j}, s_{x_j}, s_{\delta_j}, s_{\mu_j}$ of the signature. Anyway, recomputing valid $s_{x_j}, s_{\delta_j}, s_{\mu_j}$ without x_i is unfeasible under the Discrete Logarithm problem.

Result 6.4. *According to the Claims 6.4.1 and 6.4.2, we can assure that the protocol achieves the protection against DoS attacks, Sybil attacks and replay attacks.*

6.6 Conclusions and related contributions

We have presented a group signature scheme that includes short-term linkability, as an extension of the AFC proposal, in the previous chapter, including batch verification, and making it practicable for a mass-verification scenario. We presented this solution to an international conference in Montreal (Canada) in 2012, and we submitted an extended version with experimental results tested on mobile devices in an ISI-JCR Journal. We detail the publications below:

- L. Malina, J. Castellà-Roca, A. Vives-Guasch, and J. Hajny. "Short-term Linkable Group Signatures with Categorized Batch Verification". In *Foundations and Practice of Security (FPS), 5th International Symposium, LNCS 7743, pp. 244–260, doi: 10.1007/978-3-642-37119-6_16*, 2012.

Chapter 7

Conclusions

This chapter summarises the contributions, the related publications and describes possible future research lines.

Contents

In this thesis, we have focused on providing security and privacy for new applications based on electronic ticketing systems. More specifically, we have contributed to different types of systems, which includes electronic ticketing (prepaid) systems, automatic fare collection (or electronic toll, postpayment) systems, and a solution for mass-verification systems (for ticketing and mobile/vehicular networks). These systems require adapted solutions, so we have made different contributions to all of them.

We have performed a survey of the related proposals to e-ticketing systems, and classified them depending on the degree of anonymity. Apart from anonymity, we have seen that some security requirements have to be fulfilled, and that depending on the services, some requirements could apply.

7.1 Contributions

We summarize the contributions we have performed, which are:

1. We have presented a secure electronic ticketing system, that is, a prepaid system, which ensures the security requirements of exculpability and reusability while guaranteeing the privacy of users. The novel requirement of exculpability avoids false accusations between all the entities of the system. Furthermore, all the roles of the system can verify if all the steps of the protocol have been performed correctly, and in case of dispute they can contact the

TTP. Moreover, the experimental results performed in mobile devices show that the protocol can be successfully deployed in real environments for its usability.

2. We have presented an automatic fare collection (AFC) system, that is, a post-payment system, where the fare to be paid depends on the points of entrance and exit of the system, being either time- or distance-based. We have introduced the need of short-term linkability, that is, generally the movements of a same user must be unlinkable between them –in order to avoid generation of profiles– except from determined movements related to a same journey –in order to avoid fraud–, e.g. to demonstrate being the same user in the entrance and in the exit. The proposal uses group signatures to preserve the anonymity of users, and we have made an extension to achieve short-term linkability for determined movements related to a journey. Finally, the experimental results have been performed in a set of smartphones, showing then the performance of the protocol in real scenarios.

3. We have presented a solution for the use case of Vehicle Ad-hoc Networks, but can be also generalised as a solution for mass-verification systems, where there is a high frequency of sent messages that have to be verified in short time. The protocol is an evolution of the AFC system with short-term linkable group signatures to preserve privacy (both anonymity and short-term linkability) for users, by performing batch verification for those signatures. This technique allows to noticeably improve the efficiency in the part of verification, which can reduce from linear to logarithmic cost, in the best case.

7.2 Publications

The main publications supporting the content of this thesis are stated below:

ISI-JCR Journals

- A.P. Isern-Deyà, A. Vives-Guasch, M. Mut-Puigserver, M.M Payeras-Capellà, J. Castellà-Roca. "A Secure Automatic Fare Collection System for Time-based or Distance-based Services with Revocable Anonymity for Users". *The Computer Journal, doi: 10.1093/comjnl/bxs033*, 2012.

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

- A. Vives-Guasch, M.M. Payeras-Capellà, M. Mut-Puigserver, J. Castellà-Roca, and J.L. Ferrer-Gomila. "A secure e-ticketing scheme for mobile devices with Near Field Communication (NFC) that includes exculpability and reusability". *IEICE Transactions on Information and Systems, Vol.E95-D No.1, pp. 78–93, doi: 10.1587/transinf.E95.D.78*, 2012.

International Conferences with Core Category

- A. Vives-Guasch, J. Castellà-Roca, M.M Payeras-Capellà, M. Mut-Puigserver. "An Electronic and Secure Automatic Fare Collection System with Revocable Anonymity for Users". In *Advances in Mobile Computing & Multimedia (MoMM), 8th International Conference, pp. 387–392, doi: 10.1145/1971519.1971585*, 2010. Core B. ACM.

International Conferences with LNCS Proceedings

- A. Vives-Guasch, M.M. Payeras-Capellà, M. Mut-Puigserver, and J. Castellà-Roca. "E-ticketing Scheme for mobile devices with exculpability". In *Data Privacy Management and Autonomous Spontaneous Security (DPM), 5th International Workshop, LNCS 6514, pp. 79–92, doi: 10.1007/978-3-642-19348-4_7*, 2011. Springer.

- L. Malina, J. Castellà-Roca, A. Vives-Guasch, and J. Hajny. "Short-term Linkable Group Signatures with Categorized Batch Verification". In *Foundations and Practice of Security (FPS), 5th International Symposium, LNCS 7743, pp. 244–260, doi: 10.1007/978-3-642-37119-6_16*, 2012. Springer.

Patents Also some patents co-authored by the candidate and related to the scope, but not included in this thesis, are also listed below:

- "One-Touch non-NFC Network Access Configuration through NFC". Inventors: X. Pérez-Costa, and A. Vives-Guasch. Assignee: NEC Laboratories Europe Ltd. (Germany). Ref.: *NLE-411-12*. Year: 2012.

  This patent has been the result of the collaboration with Xavier Pérez-Costa during the internship that the candidate did in NEC Laboratories Europe in Heidelberg (Germany) for 4 months, by applying the knowledge of NFC technology that the candidate already had during the PhD.

- "Método para realizar transacciones con billetes digitales". Inventors: J. Castellà-Roca, J. Aragonès, A. Vives-Guasch, J. Domingo-Ferrer, L. Huguet-Rotger, J.L. Ferrer-Gomila, M. Mut-Puigserver, and M.M. Payeras-Capellà. Assignees: Universitat Rovira i Virgili (Spain), and Universitat de les Illes Balears (Spain). Ref.: *PI-04-098-2012*. Year: 2012.

  This patent has been the result of the collaboration between the Universitat Rovira i Virgili (URV) and the Universitat de les Illes Balears (UIB), which allows the transferability of e-tickets as well as preserves the security of the system and the privacy for users. Currently, we pretend to publish the proposal as an article to be presented in an international conference.

7.3 Future work

In this section, we outline some of the possible new projects or open problems in order to make contributions in that area.

In Chapters 5 and 6, we include the need of short-term linkability in these proposals to link the entrance with the exit of a system. Except from this case, any other situation must not allow the linkage of other different movements of a same user. We have made extensions of the group signatures of Boneh et al. in [Bone 04b]. In this line, some cryptographic alternatives could be analysed for

their possible application in new protocols, that, for example, could avoid the existance of optimistic TTPs, or that could provide more efficiency in determined scenarios while preserving the privacy of users.

In Chapter 6, we present a protocol that applies batch verification for multiple signatures, which provides more efficiency in the verification part. Furthermore, this proposal offers security for the system and privacy for users, that is, regarding anonymity and short-term linkability. For the near future, this proposal could be extended with experiments in mobile devices in the part of the user, and analyse the response in cloud verification. This way, we could show the feasibility of such protocols in real scenarios.

Another scope that could be of interest in the near future is the proposal of solutions for Intelligent Transport Systems, as part of the nowadays trend of the Smart Cities project, in which cities can be more efficient and can provide many services for citizens. In this ecosystem, the importance of the protection of the security and the privacy of citizens (users) takes a key role for the acceptance of these systems by the community.

Bibliography

[Amol 10] A. S. Amoli, M. Kharrazi, and R. Jalili. "2Ploc: Preserving Privacy in Location-based Services". In: *IEEE 2nd International Conference on Social Computing/PASSAT'2010*, pp. 707–712, 2010.

[Arna 06] A. Arnab and A. Hutchison. "Ticket based Identity System for DRM". *Proceedings Information Security South Africa*, 2006. Sandton, South Africa.

[Bald 10] D. Bald, G. Benelli, and A. Pozzebon. "The SIESTA project: Near Field Communication based applications for tourism". In: *IEEE 7th International Symposium on Communication Systems Networks and Digital Signal Processing (CSNDSP)*, pp. 721–725, 2010.

[Bao 04] F. Bao. "A Scheme of Digital Ticket for Personal Trusted Device". *15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC04)*, Vol. 4, pp. 3065–3069, 2004. IEEE.

[Bone 01] D. Boneh and M. Franklin. "Identity-Based Encryption from the Weil Pairing". In: J. Kilian, Ed., *Advances in Cryptology - CRYPTO 2001*, pp. 213–229, Springer Berlin Heidelberg, 2001.

[Bone 04a] D. Boneh and X. Boyen. "Short Signatures Without Random Oracles". In: C. Cachin and J. Camenisch, Eds., *Advances in Cryptology - EUROCRYPT 2004*, pp. 56–73, Springer Berlin Heidelberg, 2004.

[Bone 04b] D. Boneh, X. Boyen, and H. Shacham. "Short Group Signatures". In: *Advances in Cryptology–CRYPTO'04*, pp. 41–55, Springer Berlin Heidelberg, 2004.

[Bone 11] D. Boneh and X. Boyen. "Efficient Selective Identity-Based Encryption Without Random Oracles". *Journal of Cryptology*, Vol. 24, No. 4, pp. 659–693, 2011.

[Bone 98a] D. Boneh. "The Decision Diffie-Hellman problem". In: J. Buhler, Ed., *Algorithmic Number Theory*, pp. 48–63, Springer Berlin Heidelberg, 1998.

[Bone 98b] D. Boneh and R. Venkatesan. "Breaking RSA May Not Be Equivalent to Factoring". In: *Advances in Cryptology – EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, pp. 59–71, Springer, 1998.

[Cala 07] G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Lioy. "Efficient and robust pseudonymous authentication in VANET". In: *Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks*, pp. 19–28, ACM, New York, NY, USA, 2007.

[Caro 07] J. Caron, I. Lagrange, and L. Robet. "Contactless cell phone payment and e-ticketing: Japan leads the way at CARTES & IDentification 2007". 2007. CARTES 2007 Press release.

[Chan 06] C. C. Chang, C. C. Wu, and I. C. Lin. "A Secure E-coupon System for Mobile Users". *International Journal of Computer Science and Network Security*, Vol. 6, No. 1, pp. 273–280, 2006. IEEE.

[Chau 83] D. Chaum. "Blind signatures for untraceable payments". *Advances in Cryptology - CRYPTO'82*, pp. 199–203, 1983.

[Chen 07] Y. Y. Chen, C. L. Chen, and J. K. Jan. "A mobile ticket system based on personal trusted device". *Wireless Personal Communications: An International Journal*, Vol. 40, No. 4, pp. 569–578, 2007.

[Chen 11] L. Chen, S.-L. Ng, and G. Wang. "Threshold Anonymous Announcement in VANETs". *IEEE Journal on Selected Areas in Communications*, Vol. 29, No. 3, pp. 605–615, 2011.

[Chim 11] T. W. Chim, S.-M. Yiu, L. C. K. Hui, and V. O. K. Li. "SPECS: Secure and privacy enhancing communications schemes for VANETs". *Ad Hoc Networks*, Vol. 9, No. 2, pp. 189–203, 2011.

[Diff 76] W. Diffie and M. Hellman. "New directions in cryptography". *Information Theory, IEEE Transactions on*, Vol. 22, No. 6, pp. 644–654, 1976.

[Ding 04] R. Dingledine, N. Mathewson, and P. Syverson. "Tor: The second-generation onion router". In: *Proceedings of the 13th USENIX Security Symposium*, 2004.

[Dorn 07] A. von Dörnberg. "The global phenomenon of low cost carrier growth". *Trends and Issues in Global Tourism*, pp. 53–59, 2007. Springer-Verlag Berlin and Heidelberg GmbH & Co. KG.

[ElGa 85] T. ElGamal. "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". In: G. Blakley and D. Chaum, Eds., *Advances in Cryptology*, pp. 10–18, Springer Berlin Heidelberg, 1985.

[Elli 99] J. Elliot. "The one-card trick Multi-application smart card E-commerce prototypes". *Computing & Control Engineering Journal*, Vol. 10, No. 3, pp. 121–128, 1999. IET.

[Fan 98] C. I. Fan and C. L. Lei. "Micro-recastable ticket schemes for electronic voting". *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, Vol. E81A, No. 5, pp. 940–949, 1998.

[Ferr 09] A. L. Ferrara, M. Green, S. Hohenberger, and M. Ø. Pedersen. "Practical Short Signature Batch Verification". In: *Topics in Cryptology - The Cryptographers' Track at the RSA Conference*, pp. 309–324, Springer, April 2009.

[Ferr 10] J. L. Ferrer-Gomila, J. A. Onieva, M. M. Payeras-Capellà, and J. López-Muñoz. "Certified electronic mail: Properties revisited.". *Computers & Security*, pp. 167–179, 2010.

[Fons 07] E. Fonseca, A. Festag, R. Baldessari, and R. Aguiar. "Support of Anonymity in VANETs - Putting Pseudonymity into Practice". In: *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC), Hong Kong*, March 2007.

[Fuji 98] K. Fujimura and Y. Nakajima. "General-purpose Digital Ticket Framework". *3rd USENIX Workshop on Electronic Commerce*, pp. 177–186, 1998. USENIX.

[Fuji 99a] K. Fujimura, H. Kuno, M. Terada, K. Matsuyama, Y. Mizuno, and J. Sekine. "Digital-Ticket-Controlled Digital Ticket Circulation". *8th USENIX Security Symposium*, pp. 229–240, 1999. USENIX.

[Fuji 99b] K. Fujimura, Y. Nakajima, and J. Sekine. "XML Ticket: Generalized Digital Ticket Definition Language". *W3C XML-Dsig'99*, 1999.

[Galb 08] S. D. Galbraith, K. G. Paterson, and N. P. Smart. "Pairings for cryptographers". *Discrete Appl. Math.*, Vol. 156, No. 16, pp. 3113–3121, Sep. 2008.

[Gerl 07] M. Gerlach, A. Festag, T. Leinmuller, G. Goldacker, and C. Harsch. "Security architecture for vehicular communication". In: *The 5th International Workshop On Intelligent Transportation*, March 2007.

[Gran 07] N. Granados, K. Gupta, and R. Kauffman. "IT-enabled transparent electronic markets: the case of the air travel industry". *Inf. Syst. E-Business Management*, pp. 65–91, 2007.

[Hane 02] D. Haneberg. *electronic Ticketing A Smartcard Application Case-Study*. Master's thesis, Institut Für Informatik, 2002. Technical Report 2002-16, http://www.informatik.uni-augsburg.de/lehrstuehle/swt/se/publications/2002-e_ticket_scard_app_stud/2002-e_ticket_scard_app_stud-pdf.pdf.

[Hane 04] D. Haneberg, K. Stenzel, and W. Reif. "Electronic-Onboard-Ticketing: Software Challenges of an State-of-the-Art M-Commerce Application". In: K.Pousttchi and K.Turowski, Eds., *Workshop Mobile Commerce*, pp. 103–113, Gesellschaft für Informatik (GI), 2004.

[Hane 08] D. Haneberg. "Electronic ticketing: risks in e-commerce applications". *Digital excellence*, pp. 55–66, 2008. Springer-Verlag.

[Heyd 06] T. S. Heydt-Benjamin, H. J. Chae, B. Defend, and K. Fu. "Privacy for Public Transportation". In: *6th Workshop on Privacy Enhancing Technologies (PET 2006)*, pp. 1–19, 2006. LNCS 4258.

[Huss 09] R. Hussain, S. Kim, and H. Oh. "Towards Privacy Aware Pseudonymless Strategy for Avoiding Profile Generation in VANET". In: H. Youm and M. Yung, Eds., *Information Security Applications*, pp. 268–280, Springer Berlin Heidelberg, 2009.

[Jorn 07] O. Jorns, O. Jung, and G. Quirchmayr. "A privacy enhancing service architecture for ticket-based mobile applications". In: *2nd International Conference on Availability, Reliability and Security*, pp. 374–383, ARES 2007 - The International Dependability Conference, Vienna, Austria, Apr 2007. vol. 24.

[Joux 00] A. Joux. "A One Round Protocol for Tripartite Diffie-Hellman". In: W. Bosma, Ed., *Algorithmic Number Theory*, pp. 385–393, Springer Berlin Heidelberg, 2000.

[Kref 05] H. Kreft. "Cashing up with mobile Money - The fairCASH way". *Euro mGov 2005*, p. 29, 2005. Sussex University, Brighton (UK), Mobile Government Consortium International LLC.

[Krem 02] S. Kremer, O. Markowitch, and J. Zhou. "An Intensive Survey of Fair Non-Repudiation Protocols". *Computer Communications*, Vol. 25, pp. 1606–1621, 2002.

[Kunt 07] N. Kuntze and A. U. Schmidt. "Trusted ticket systems and applications". *New Approaches for Security, Privacy and Trust in Complex Systems. IFIP International Federation for Information Processing*, Vol. 232, 2007.

[Kura 00] K. Kuramitsu, T. Murakami, H. Matsuda, and K. Sakamura. "TTP: Secure ACID transfer protocol for electronic ticket between personal tamper-proof devices". In: *24th Annual International Computer Software and Applications Conference (COMPSAC2000)*, pp. 87–92, Taipei, Taiwan, Oct 2000. vol. 24.

[Kura 02] K. Kuramitsu and K. Sakamura. "Electronic Tickets on Contactless Smartcard Database". In: *Proceedings of the 13th International Conference on Database and Expert Systems Applications*, pp. 392–402, 2002. LNCS 2453.

[Lin 07] X. Lin, X. Sun, P. han Ho, and X. Shen. "GSIS: A Secure and Privacy Preserving Protocol for Vehicular Communications". *IEEE Transactions on Vehicular Technology*, Vol. 56, No. 6, pp. 3442–3456, 2007.

[Lutg 07] J. Lutgen. "The Security Infrastructure of the German Core Application in Public Transportation". In: *Isse/secure 2007 Securing Electronic Business processes: Highlights of the Information Security Solutions Europe/secure 2007 Conference*, pp. 411–419, Vienna, Austria, 2007. Vieweg&Teubner Verlag.

[Mali 11] L. Malina and J. Hajny. "Accelerated modular arithmetic for low-performance devices". In: *the 34th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 131 –135, Aug. 2011.

[Mana 01] A. Maña, J. Martínez, S. Matamoros, and J. M. Troya. "GSM-Ticket: Generic Secure Mobile Ticketing Service". *Gemplus World Developers Conference*, 2001. Gemplus, Paris (France).

[Mats 03] S. Matsuo and W. Ogata. "Electronic ticket scheme for ITS". *IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences*, Vol. E86A, No. 1, pp. 142–150, 2003.

[McDa 93] R. L. McDaniel and F. Haendler. "Advanced RF Cards for Fare Collection". In: *Commercial Applications and Dual-Use Technology, Conference Proceedings*, pp. 31–35, Telesystems Conference, 1993.

[Mihl 02] F. Mihlberg. *On the Formal Analysis of E-ticketing Protocols*. Master's thesis, School of Computer Science and Engineering, 2002.

[Naka 99] T. Nakanishi, N. Haruna, and Y. Sugiyama. "Unlinkable Electronic Coupon Protocol with Anonymity Control". *Proceedings of the Second International Workshop on Information Security*, pp. 37–46, 1999. LNCS 1729.

[Pate 97] B. Patel and J. Crowcroft. "Ticket Based Service Access for the Mobile User". *Proceedings of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'97)*, pp. 223–233, 1997. Budapest, Hungary.

[Pedo 00] F. Pedone. "A Two-Phase Highly-Available Protocol for Online Validation of E-Tickets". *Hewlett-Packard Labs Technical Reports*, 2000. HPL-2000-116 20000929.

[Qin 11] B. Qin, Q. Wu, J. Domingo-Ferrer, and L. Zhang. "Preserving security and privacy in large-scale VANETs". In: *Proceedings of the 13th international conference on Information and communications security*, pp. 121–135, Springer-Verlag, 2011.

[Quer 05] D. Quercia and S. Hailes. "MOTET: Mobile transactions using electronic tickets". In: *1st International Conference on Security and Privacy*

for Emerging Areas in Communications Networks, Proceedings, pp. 374–383, Athens, Greece, Sep 2005. vol. 24.

[Raya 07] M. Raya and J.-P. Hubaux. "Securing vehicular ad hoc networks". *J. Comput. Secur.*, Vol. 15, pp. 39–68, January 2007.

[Rive 83] R. L. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public-key cryptosystems". *Commun. ACM*, Vol. 26, No. 1, pp. 96–99, Jan. 1983.

[Schn 91] C. P. Schnorr. "Efficient signature generation by smart cards". *Journal of Cryptology*, Vol. 4, pp. 161–174, 1991.

[Serb 08] C. Serban, Y. Chen, W. Zhang, and N. Minsky. "The concept of decentralized and secure electronic marketplace". *Electronic Commerce Research*, Vol. 8, No. 1-2, pp. 79–101, 2008.

[Siu 01a] I. W. Siu and Z. S. Guo. "The secure communication protocol for electronic ticket management system". In: *8th Asia-Pacific Software Engineering Conference (APSEC2001)*, University of Macau, 2001.

[Siu 01b] W. I. Siu and Z. S. Guo. "Application of Electronic Ticket to Online Trading with Smart Card Technology". *Proceedings of the 6th INFORMS Conference on Information Systems and Technology (CIST-2001)*, pp. 222–239, 2001. Miami Beach, Florida (US).

[Song 03] R. Song and L. Korba. "Pay-TV System with Strong Privacy and Non-Repudiation Protection". *IEEE Transactions on Consumer Electronics*, Vol. 49, No. 2, pp. 408–413, 2003.

[Stan 07] N. I. of Standards and Technology. "Special Publication 800-57 Part 1". 2007. Recommendation for Key Management.

[Stud 09] A. Studer, E. Shi, F. Bai, and A. Perrig. "TACKing Together Efficient Authentication, Revocation, and Privacy in VANETs.". In: *SECON*, pp. 1–9, IEEE, 2009.

[Sun 10] J. Sun, Y. Fang, and X. Zhu. "Privacy and emergency response in e-healthcare leveraging wireless body sensor networks". *Wireless Com.*, Vol. 17, No. 1, pp. 66–73, Feb. 2010.

[Tsio 98] Y. Tsiounis and M. Yung. "On the security of ElGamal based encryption". In: H. Imai and Y. Zheng, Eds., *Public Key Cryptography*, pp. 117–134, Springer Berlin / Heidelberg, 1998.

[Vald 03] M. E. G. Valdecasas-Vilanova, R. Endsuleit, J. Calmet, and I. Bericht. *State of the Art in Electronic Ticketing*. Master's thesis, Institut für Algorithmen und Kognitive Systeme, 2003.

[Wang 04a] G. Wang, F. Bao, J. Zhou, and R. H. Deng. "Proxy Signatures Scheme with Multiple Original Signers for Wireless E-Commerce Applications". *Vehicular Technology Conference, VTC2004-Fall*, Vol. 5, pp. 3249–3253, 2004. IEEE.

[Wang 04b] S. C. Wang, K. Q. Yan, and C. H. Wei. "Mobile Target Advertising for Mobile User". *International Workshop on Business and Information (BAI 2004)*, Vol. V2, 2004. Taipei, Taiwan.

[Wase 10] A. Wasef and X. S. Shen. "Efficient Group Signature Scheme Supporting Batch Verification for Securing Vehicular Networks". In: *IEEE International Conference on Communications (ICC)*, 2010.

[Wei 11] L. Wei, J. Liu, and T. Zhu. "On a Group Signature Scheme Supporting Batch Verification for Vehicular Networks". In: *International Conference on Multimedia Information Networking and Security*, pp. 436–440, IEEE C. S., Los Alamitos, CA, USA, 2011.

[Zhan 08] C. Zhang, R. Lu, X. Lin, P.-H. Ho, and X. Shen. "An Efficient Identity-Based Batch Verification Scheme for Vehicular Sensor Networks.". In: *INFOCOM*, pp. 246–250, IEEE, 2008.

[Zhan 10] L. Zhang, Q. Wu, A. Solanas, and J. Domingo-Ferrer. "A Scalable Robust Authentication Protocol For Secure Vehicular Communications". In: *IEEE Transactions on Vehicular Technology 59(4)*, pp. 1606–1617, 2010.