TECHNISCHE UNIVERSITÄT DRESDEN

Europäische Union

Europa fördert Sachsen.
ESF
Europäischer Sozialfonds

Fakultät Informatik Institut für Systemarchitektur

Privacy-preserving E-ticketing Systems for Public Transport Based on RFID/NFC Technologies

Dissertation

Submitted to the Faculty of Computer Science, TU Dresden,
in Partial Fulfillment of the Requirements for the Degree of Dr.-Ing.

Presented by M.C.S. **Ivan Gudymenko**

Born on 15 December 1986 in Mykolaiv, Ukraine

Scientific Advisers:

Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill	TU Dresden, Germany
Prof. Dr. habil. Simone Fischer-Hübner	Karlstad University, Sweden
Dr.-Ing. Katrin-Borcea Pfitzmann	TU Dresden, Germany

Day of Defense: April 20, 2015

Dresden, May 26, 2015

Administrative Support of This Work

This work is financially supported by the Free State of Saxony and the European Social Fund (ESF) under grant number 100097574. The author would like to express his gratitude for the provided support which enabled him to concentrate on the dissertation and exchange experience with other scientists by participating in a number of conferences and workshops.

Statement of Authorship

Herewith I declare that this dissertation is my own work and that to the best of my knowledge it contains no material previously published, written or presented by another person without the proper acknowledgment.

Dresden, May 26, 2015.

Ivan Gudymenko

Abstract

Pervasive digitization of human environment has dramatically changed our everyday lives. New technologies which have become an integral part of our daily routine have deeply affected our perception of the surrounding world and have opened qualitatively new opportunities. In an urban environment, the influence of such changes is especially tangible and acute. For example, ubiquitous computing (also commonly referred to as UbiComp) is a pure vision no more and has transformed the digital world dramatically. Pervasive use of smartphones, integration of processing power into various artefacts as well as the overall miniaturization of computing devices can already be witnessed on a daily basis even by laypersons. In particular, transport being an integral part of any urban ecosystem have been affected by these changes. Consequently, public transport systems have undergone transformation as well and are currently dynamically evolving. In many cities around the world, the concept of the so-called electronic ticketing (e-ticketing) is being extensively used for issuing travel permissions which may eventually result in conventional paper-based tickets being completely phased out already in the nearest future. Opal Card in Sydney [1], Oyster Card in London [2], Touch & Travel in Germany [3] and many more are all the examples of how well the e-ticketing has been accepted both by customers and public transport companies.

Despite numerous benefits provided by such e-ticketing systems for public transport, serious privacy concern arise. The main reason lies in the fact that using these systems may imply the dramatic multiplication of digital traces left by individuals, also beyond the transport scope. Unfortunately, there has been little effort so far to explicitly tackle this issue. There is still not enough motivation and public pressure imposed on industry to invest into privacy. In academia, the majority of solutions targeted at this problem quite often limit the real-world pertinence of the resultant privacy-preserving concepts due to the fact that inherent advantages of e-ticketing systems for public transport cannot be fully leveraged.

This thesis is aimed at solving the aforementioned problem by providing a privacy-preserving framework which can be used for developing e-ticketing systems for public transport with privacy protection integrated from the outset. At the same time, the advantages of e-ticketing such as fine-grained billing, flexible pricing schemes, and transparent use (which are often the main drivers for public to roll out such systems) can be retained.

Acknowledgements

I would like to express my sincerest gratitude to all people who helped me during these three years. At first, I would like to thank Prof. Schill for providing me with his constant support and fast feedback which enabled me to focus on the dissertation and essentially shielded me from many administrative hurdles each PhD student goes through. He constantly motivated me to continue my research at the same time providing me with freedom which I value very much. I learned a lot from Prof. Schill and I am happy to have been supervised by him.

Next, from all my heart I would like to thank Katrin Borcea-Pfitzmann for believing in me from the very beginning when I was still at the crossroads during the final phase of my master's she supervised. Moreover, Katrin supported me enormously during the application process to obtain financing for my PhD project. During my PhD studies she provided great help and advice which was especially crucial at the beginning. Somehow I have always had a feeling that when I really needed her support she has always been there guiding me during the most decisive moments of these three years.

Moreover, my sincerest acknowledgment goes to Prof. Andreas Pfitzmann who lightened up my heart and completely changed my view on how responsive, kind and astonishingly near a full professor can be to the students. He provided me with that very first decisive impulse to enter the field of privacy protection in the IT and motivated me to choose the path I have been taking ever since. I deeply regret that he passed away so early and that we could not continue our cooperation, both at the scientific and, what is arguably even more important for a young person like me, at the human level.

I would also like to express my gratitude to Marius Feldmann who not only pointed me to an attractive program for financing my PhD studies but also greatly supported me during the proposal writing (together with Katrin). I do not know if I would have really made it, if Marius hadn't knocked on my door back then holding a paper with the information on that PhD program...

During the active phase of my dissertation, I got a lot of support from the team of chair of privacy and data security at our university (in German, Lehrstuhl für Datenschutz und Datensicherheit, DuD). With this respect, I would like in the first place to thank Stefan Köpsell, Sebastian Clauß, and Martin Beck. Our discussions and the resulting feedback played a very important role for my research. Moreover, the whole DuD team was very kind to me and it feels they have become some kind of a second family for me (yes, I am perfectly aware that it may sound a bit exaggerated but in this case it is so). Furthermore, I would like to express my gratitude to Florian Kerschbaum for our cooperation during his time at TU Dresden which was very beneficial and fruitful for me. I would also like to thank Prof. Thorsten Strufe for our interesting discussions and for being so kind and cheerful. I wish you would have moved to Dresden earlier!

Towards the end of my dissertation, I was lucky to have a short academic visit at Karlstad university in Sweden. Prof. Simone Fischer-Hübner and the members of her group provided

me with valuable feedback and fruitful discussions. I would like to thank Simone for having me there, for her warm welcome, and for valuable comments. Furthermore, I would like to express my gratitude to Tobias Pulls, Stefan Berthold and Leonardo Martucci for our long discussions.

I was privileged to supervise very prospective students during my PhD studies who proved to be very dedicated and creative. My work with Pavel Plakhin, Manuel Weißbach, and Álvaro Giménez Serrano was definitely of mutual benefit for both parties and I would like to thank them for their input (especially with respect to the practical evaluation) which helped to improve my dissertation.

Additionally, I would like to thank Stefan Köpsell, Martin Beck, Stefanie Roos, Manuell Weißbach, Katrin Borcea-Pfitzmann and Sergei Arnautov for proofreading my dissertation at the end and providing me with valuable comments which helped to improve and polish up the final version. I really appreciate your time and effort!

Last but on no account least: without the massive support of my family I would never have achieved my ”small wins”. Neither would I have become a person I am today (let alone the obvious fact that without my parents I wouldn’t have been brought to this world at all ☺). I would also like to thank my girlfriend and all my friends for being by my side and helping to make my life dynamic and complete.

Contents

List of Figures

List of Tables

Acronyms

(U)SIM	(Universal) Subscriber Identity Module
AFC	Automated fare collection
BIBO	Be in/be out
CICO	Check-in/check-out
CLF	Contactless Front-end
EPT	Electronic paper ticket
ESPT	E-ticketing Systems for Public Transport
HF	High-frequency
IFMS	Interoperable Fare Management System
ITS	Intelligent Transport System
LF	Low-frequency
LLCP	Logical Link Control Protocol
NDEF	NFC Data Exchange Format
NFC	Near Field Communication
NFC-WI	NFC Wired Interface
OTA	Over-the-Air
PDU	Protocol Data Unit
PII	Personally Identifiable Information
RF	Radio frequency
RFID	Radio Frequency Identification
RTD	Record Type Definition
SE	Secure Element
SNDEF	Simple NDEF Exchange Protocol
SPoF	Single Point of Failure
SWP	Single Wire Protocol
TA	Transport authority
TR	Travel record

TTP	Trusted third party
UbiComp	Ubiquitous Computing
WIWO	Walk in/walk out

1 Introduction

Since the last decade, computing has made a giant leap forward and continues to evolve rapidly deeply penetrating into everyday life and accompanying individuals everywhere, anytime. The technological advance has not only lead to the dramatic burst of computational power but, what may be even more important, has made it possible to integrate intelligence into the surrounding artefacts. The latter has become a significant driver for implementing the vision of the so-called Ubiquitous Computing (UbiComp) into life.

Despite the numerous benefits such systems provide, serious concerns over privacy of their users arise, since the ubiquitous properties may pave the way to privacy invasion. This quite often becomes a serious hurdle on the way to public acceptance of ubiquitous systems and thus directly influences their commercial success. Therefore, this dissertation focuses on the development of necessary concepts for making ubiquitous systems privacy-respecting while preserving their core advantages (such as easiness of use, etc.) and by this making such systems more attractive for customers. Since the notion of a ubiquitous system is still quite broad, the more specific case of RFID/NFC-based systems was considered. Radio Frequency Identification (RFID) technology together with well standardized Near Field Communication (NFC) can be indeed regarded to be among the main enablers of UbiComp. More specifically, the focus was made on e-ticketing systems for public transport based on NFC/RFID.

1.1 Motivation

As ubiquitous computing has already transformed itself from a vision to reality, various parts of our daily lives are being rapidly affected by it. The transport infrastructure of the future comprising of the so-called intelligent transport systems (ITS) is one of the tangible examples of this process. The area of public transport being an inherent part of any urban transport system is, therefore, no exception either. More specifically, the trend of replacing the conventional paper-based tickets with their electronics counterparts (e-tickets) is constantly growing. Public transport systems incorporating the e-ticketing paradigm are referred to as e-ticketing systems for public transport (ESPT) within this dissertation. The front-end of ESPT is typically based on RFID/NFC technologies. Such systems have already been in operation for quite a while in many large cities around the world, for example in London [2], Singapore [11] or São Paulo [12]. In fact, the latest advances in mobile technology allowing to integrate NFC into widely spread smartphones have opened new opportunities for ESPT at the same making such e-ticketing systems even more ubiquitous.

Along with the numerous benefits of future public transport systems, serious privacy concerns arise. The utilization of e-tickets integrated into a customer's smartphone with NFC support or stored on a smart card is going to dramatically multiply the digital traces left by people using the system. This paves the way to various misuse scenarios if no mechanisms for privacy protection are explicitly considered. Till now, this issue has not been sufficiently addressed by the industry. The academic solutions developed so far are either based on additional assumptions or are far too inefficient to be integrated into a real-life system. Moreover, the inherent trade-offs between privacy protection and security (especially for a transport authority) as well as between privacy and efficiency pose a significant challenge for system developers. Furthermore, the majority of academic privacy-preserving solutions have a negative side-effect of disabling some important features of ESPT which are, however, an important advantage of such systems. For example, the following features could be referred to: the support for flexible, highly customizable tariff schemes and fine-grained billing, easiness of use, etc. Therefore, a privacy-preserving solution leveraging the full potential[1] of ESPT is highly required.

1.2 Research Area and Dissertation Focus

UbiComp is a hot topic in the scientific community. It is a complex and ambiguous notion incorporating different technologies and is often used in conjunction with other related buzzwords such as pervasive computing, ambient intelligence (AmI) or the Internet of things (IoT). In order to resolve possible ambiguities, the term *ubiquitous systems* is used within this dissertation referring to the (distributed) systems the main parts of which consist of powerful internetworked processing centers (*back-end*) and the intelligence-enabled artefacts (*front-end*). The latter is responsible for the "ubiquity" of a system, whereas the former creates the necessary backbone network which is required to interconnect the front-end devices and keep the system functioning (background processes, implementing system logic, etc.). The third part – *the bridging element* – interconnects front-end and back-end therefore acting as a bridge between the two main components.

The classic and by far the most widespread technology for realization of such ubiquitous systems is RFID. Another promising technology very closely related to RFID is the so-called NFC currently being actively promoted by several leading IT companies. Therefore, in this dissertation, the focus is made on RFID- and NFC-based ubiquitous systems. Since the area of NFC and especially RFID is rather broad being used in different application domains and incorporating several standards, the *e-ticketing* domain was chosen as the primary research target. More specifically, the e-ticketing systems for public transport (ESPT) were considered. The basic architecture of such systems essentially consists of e-tickets residing on a user device (a smart card or an NFC-enabled smartphone), terminals validating e-tickets, and the back-end (a more detailed description is presented in Section 2.3). Along with the numerous benefits this kind of e-ticketing systems provide, serious privacy concerns arise (see the discussion in Section 2.1). The main reason for that is the dramatic increase in the amount of digital traces left by the customers. This thesis, therefore, is dedicated to the subject of privacy preservation in ESPT. More specifically, the solution is devised which on the one hand aims at protecting privacy of system's customers, and on the other hand allows for the inherent benefits of e-ticketing systems, such as transparent usage and benefits from flexible tariffs (and possibly individual discounts) by leveraging the fine-grained billing approach.

[1] The advantages of ESPT are discussed in Section 2.1.1.

1.3 Main Goals of the Work

The main goal of this dissertation is to develop a framework for building a loosely-coupled privacy-preserving e-ticketing system which *(1)* allows for local validation of e-tickets and *(2)* supports fine-grained billing for registered customers. Loosely-coupled architecture implies that terminals can serve check-in/out requests in the front-end without requiring real-time connection to the back-end. This is an important requirement since it to a large extent defines the real-world pertinence of an e-ticketing system for public transport. Based on the main goal described above, the specific research questions (RQ) have been derived for this work which are summarized below.

Research Questions

RQ 1. How to provide for a privacy-preserving local validation at the terminal side such that:

a) valid e-tickets remain anonymous to the terminal;

b) invalid e-tickets are rejected.

RQ 2. How to allow for privacy-preserving travel records processing in the back-end such that:

a) fine-grained billing for the registered tickets is possible;

b) customer identification is prevented.

Main Challenges

The aforementioned research questions introduce the following challenges for system developers:

Challenge 1. Terminals are widely distributed in the system and therefore can not be entrusted to manage privacy-sensitive information pertaining to e-tickets and owners thereof. Therefore, it should be possible for terminals *(1)* to authenticate e-tickets and *(2)* to check their validity without being able to identify them or even to distinguish between them.

Challenge 2. In order to enable fine-grained billing for registered customers, different rides pertaining to a single customer have to be correlated in some way. This, however, has privacy implications, since e-tickets (and hence their owners) could be a subject to illegal tracking.

1.4 Main Contributions

The contribution of this thesis is a privacy-preserving framework which explicitly addresses the issues of privacy protection in e-ticketing systems for public transport (ESPT). On the one hand, it allows for the implementation of flexible pricing schemes and fine-grained billing. A transport authority, therefore, would be able to fully leverage the whole potential of such systems. On the other hand, our solution addresses privacy protection from the outset through the specific system design. Moreover, in contrast to several other solutions, the developed framework is based on a loosely-coupled architecture implying that terminals do not have to maintain permanent real-time connection to the back-end in order to serve check-in/check-out requests in the front-end. The main contributions of the thesis are summarized below.

1. Design and evaluation of a framework allowing to develop privacy-preserving e-ticketing systems for public transport.
    a) The developed approach enables for the implementation of fine-grained billing and transparent tariff schemes in a natural way.
    b) At the same time, privacy properties can be retained.
2. Important findings with respect to practical realization of interactive privacy-preserving protocols on end user devices (RFID smart cards and NFC-enabled smartphones):
    a) The issues concerning the implementation of interactive protocols via an NFC interface on Android-based smartphones have been discussed.
    b) The hurdles encountered while applying the protocols requiring the non-standard cryptographic primitives to conventional smart cards were described.
3. Classification of the field with respect to privacy preservation in the context of public transport systems based on e-ticketing was presented.

The core findings of the thesis were published at the following conferences:

- Ivan Gudymenko. A Privacy-Preserving E-Ticketing System for Public Transportation Supporting Fine-Granular Billing and Local Validation. In *Proceedings of the 7th International Conference on Security of Information and Networks, Glasgow, UK*, SIN '14, New York, NY, USA, 2014. ACM. Best paper award in section "Assuarance and Trust"

- Ivan Gudymenko, Felipe Sousa, and Stefan Köpsell. A Simple and Secure E-Ticketing System for Intelligent Public Transportation based on NFC. In *The First International Conference on IoT in Urban Space, Rome, Italy*, Urb-IoT, New York, NY, USA, 2014. ACM

- Ivan Gudymenko. On Protection of the User's Privacy in Ubiquitous E-ticketing Systems Based on RFID and NFC Technologies. In *PECCS 2013 - Proceedings of the 3rd International Conference on Pervasive Embedded Computing and Communication Systems.* SciTePress, February 2013

- Florian Kerschbaum, Hoon Wei Lim, and Ivan Gudymenko. Privacy-Preserving Billing for e-Ticketing Systems in Public Transportation. In *Proceedings of the 12th ACM workshop on privacy in the electronic society, WPES'2013* , WPES '13, pages 143–154, New York, NY, USA, July 2013. ACM

1.5 Thesis Structure

After the introduction, the necessary background on e-ticketing systems and underlying technologies is presented in Chapter 2. Then, the privacy issues in the e-ticketing domain are introduced in Chapter 3 together with the discussion of generic threats and countermeasures as well as with the analysis of the degree to which privacy is considered in the respective RFID and NFC standards. Chapter 3 is concluded by the discussion of the core requirements for a privacy-preserving e-ticketing system together with the adopted attacker model. A detailed related work review is presented in Chapter 4. Chapter 5 is devoted to the discussion of our solution. The respective evaluation is provided in Chapter 6. Chapter 7 concludes the thesis.

Chapter Summary

In this chapter, the target area of research together with the respective motivation and problem statement were presented. The main goals and research questions for the dissertation have been discussed and the main challenges outlined. The core contributions of the thesis together with the respective publications have been summarized

2 Background: E-Ticketing Systems and Underlying Technology

In this chapter, the e-ticketing concept is discussed in detail. Firstly, its advantages and disadvantages are highlighted in Section 2.1. Main functional types of an e-ticket together with the most relevant application areas are presented in Section 2.2. The general application scenario of public transport employing the e-ticketing concept which is the focus of this dissertation is outlined in Section 2.3. Different fare collection approaches in e-ticketing are discussed in Section 2.4 highlighting the ones based on check-in/check-out. The underlying technologies (RFID and NFC) enabling check-in/check-out in e-ticketing systems for public transport (ESPT) together with the respective standards are discussed in Section 2.5. The chapter is concluded with raising the concerns over privacy-friendliness of ESPT which serves as the logical transition to Chapter 3 where these issues are discussed in detail.

2.1 E-ticketing in Public Transport: Pros and Contras

E-ticketing is an attractive approach to fare management since it has a decent potential of providing a set of unique long-term benefits both for the customers of an e-ticketing system and for the service providers (public transport companies).

2.1.1 E-ticketing: Advantages

For **customers**, an e-ticketing system can potentially offer the following advantages:

- Faster and more convenient verification of a ticket [10];
- Saving on travel expenses due to the ”pay-as-you-go” feature (paying for the actual distance travelled);
- The ability to profit from a flexible fare pricing scheme (possibly with individual discounts and special offers);
- Revocation of lost tickets and their replacement [10];
- Increased usability:
  - no need to have change for local ticket issuing machines[1] (for instance, for customers only sporadically using the transport service or while being in another city);
  - no need to learn complex fare pricing schemes: the system can automatically choose the best option and possibly suggest a discount (e.g., based on the customer’s travel habits);

[1] This is possible if some form of electronic payment is supported. For convenience, the e-ticket could even be linked to the bank account of a customer who can then transparently use the transport service in another city, for example, without having to buy a single trip ticket from a local ticket issuing machine.

For **public transport companies**, the adoption of the e-ticketing concept can be beneficial due to the following reasons:

- Decrease in system maintenance costs [10, 17];
- Infrastructure optimization through the analysis of detailed statistics (a very important source of feedback);
- Significant reduction of payment handling costs [18];
- Improvement of the fare dodgers rate through more efficient ticket verification [10];
- Mitigation of the ticket forgery problem by leveraging various digital tools (e.g., based on digital signatures, authentication primitives, etc.) [10];
- The ability to create highly flexible fare pricing schemes and innovative ticketing solutions [17];
- The opportunity to create innovative multi-application schemes combining the transit with the non-transit[1] functions [17];
- The possibility to create interoperable solutions between cooperating transport service providers with subsequent revenue sharing[2];
- Consequently, transport service providers can attract more customers and therefore generate more revenue.

2.1.2 E-ticketing: Disadvantages

Along with tangible benefits, the concept of e-ticketing raises several concerns. For a **customer**, it can be the possible reluctance to using a conventional transport system in a new way (especially among the elderly). Moreover, the privacy-related issues arise, namely:

- Ubiquitous customer identification;
- The possibility of customer profiling (creation of location patterns, etc.);
- The resulting privacy violation through the increased surveillance potential (”The Big Brother” problem).

If the privacy concerns outweigh the possible benefits for the user, the latter is likely to prefer using the public transport system in a conventional way, which my eventually hinder the successful roll-out of the respective e-ticketing system. For example, in the Netherlands a similar situation occurred in the area of smart metering. Namely, the increased concerns over privacy and the resulting public outcry introduced a serious hurdle for the successful deployment of the local smart metering system [19].

[1] An example of this would be using an e-ticket for a discount in food vending machines deployed at stations, etc.

[2] The e-ticketing concept enables to create an interoperable architecture of public transport services from different transport companies (e.g., in different cities or even countries). This allows a customer to use a single e-ticket with different providers in a transparent way. At the same time, the providers of transport service can share the profits from collaborative business relations.

For **transport companies**, moving to an e-ticketing system may raise the following concerns:

- Relatively high system development costs;
- Lack of mature interoperable solutions and standardization in the area as a whole (many of the developed e-ticketing systems are proprietary);
- The necessity to invest in the creation of a new infrastructure and its deployment (which might involve high risks for a relatively low-profit public transport business [17]);
- Possible reluctance to using the system from the customer side due to privacy reasons. Moreover, due to constantly growing privacy awareness, public transport companies may be pushed to additionally invest in privacy to retain current customer base and attract new customers.
- To ensure interoperability between different service providers, the respective architecture must be developed to provide for efficient, secure, and privacy-respecting sharing of the respective billing information, which introduces yet another challenge to transport companies.

2.2 Main Functional Types of an E-ticket and Application Areas

There exist several fundamentally different management schemes for e-ticketing systems which by analogy with the classification presented in [20] can be roughly divided into *account-based* and *card-based*. The notion of an electronic ticket (or an e-ticket), therefore, can be respectively understood in several ways, namely as *(1)* a widely used *online ticket* or as *(2)* a *"smart ticket"* – an electronic medium (e.g., an RFID or NFC chip) holding the digitalized version of rights to claim a certain service (e.g., a travel permission). The former has already become a conventional way of issuing tickets for public transport, events, etc., through creation and registration of the respective digital record at the service provider side. A user then simply prints out a copy of the ticket itinerary which usually contains the reservation number essentially serving as a *reference* to the respective digital record in the back-end. Upon check-in, the itinerary is presented possibly together with the document certifying identity (e.g., a passport).

To the contrary, the second type of an e-ticket (a "smart ticket") is stored on a user device in the form factor of a smart card or a smart phone, for example. Such e-ticket can by *itself* prove the validity of a service request to an electronic validator by this rendering the need of personal ID check unnecessary in many cases.

E-ticketing systems using the latter type of an e-ticket are in the focus of this dissertation (see Figure 2.1). In order to avoid ambiguities, the term e-ticket is further used solely with respect to the second type of an e-ticket.

It should be mentioned, that the e-ticketing concept can be utilized in several application domains such as public transport, event ticketing, fitness studious, etc. Each of them imposes specific requirements on the e-ticketing system which determines the eventual design of the latter. In this dissertation, however, the focus is made on public transport domain, see Figure 2.1.

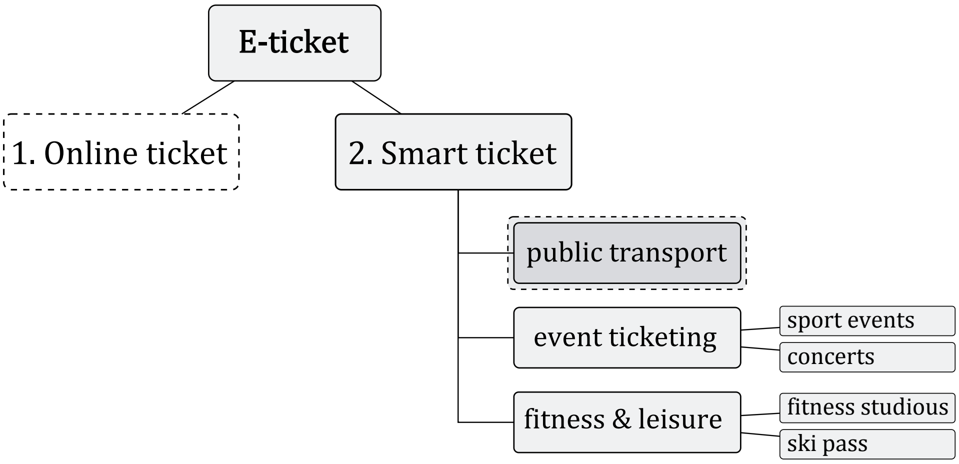


Figure 2.1: E-ticket taxonomy depicting the main application areas. The dissertation focuses on public transport which is respectively marked.

2.3 E-ticketing in Public Transport: A General Application Scenario

The e-ticketing concept can be implemented in a number of ways. The general application scenario, however, can be described as follows (see Figure 2.2). A customer acquires an e-ticket possibly registering himself to enable fine-grained billing and flexible pricing schemes with individual discounts, for example (step 1 in Figure 2.2). The trip begins when the customer enters the transport vehicle and checks in (step $2a$). The check-in procedure is performed through the reading device (reader) installed in the vehicle. On successful validation, the reader forwards the e-ticket ID to the on-board processing unit which registers the check-in time, the geographical coordinates[1], etc. When the customer has reached the final destination, he/she checks out (using the on-board reader) and leaves the transport vehicle (step $2b$). Similarly to the check-in case, the time, location, etc. are registered again and the co-called travel record is eventually formed. The latter is then communicated to the back-end system for processing purposes, statistics analysis and possibly for the application of individual fare pricing schemes and fine-grained billing (step 3). All interchanges performed by the customer during the trip are, therefore, registered, respectively processed, and subsequently included into the bill. Note that depending on the system implementation, the reader functionality and subsequent event processing may be encapsulated in a single device which is often referred to as a terminal. Moreover, terminals may be deployed at the stops (i.e. stationary terminals) and not on the vehicle. Furthermore, the adopted scheme for fare collection determines if and how travel records are created (see the respective discussion in Section 2.4).

[1] The coordinates determination can be performed through the GPS technology or, for example, by registering the stop where a customer entered the transport vehicle. The combined approached is used in a so-called Vehicle Location System (VLS) which was deployed in Singapore public transport system [21].

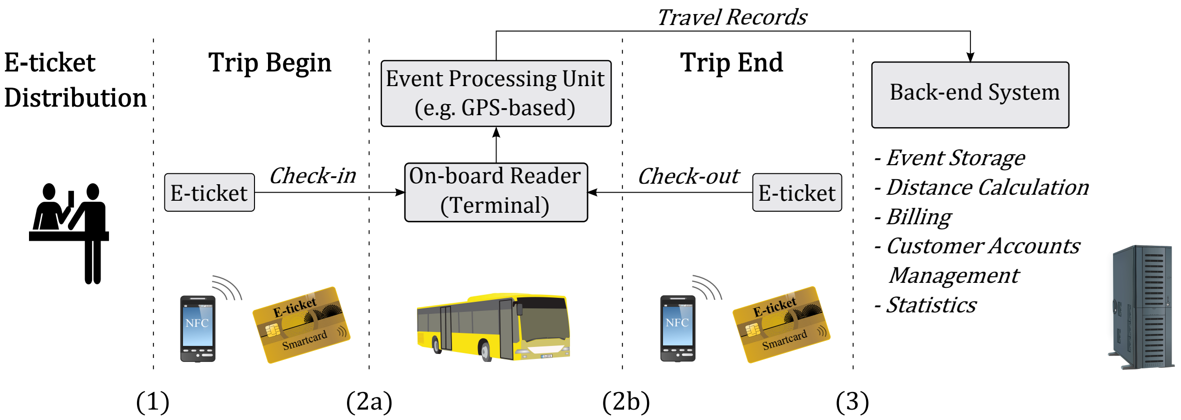


Figure 2.2: E-ticketing: a general application scenario.

2.4 Fare Collection in E-ticketing: Main Approaches

One of the main advantages of e-ticketing is the ability to collect transport fares in an automated fashion which is often referred to as automated fare collection (AFC). There are several approaches which enable AFC, see Figure 2.3. The chosen approach determines how price calculation schemes are going to be developed and greatly affects the overall system design.

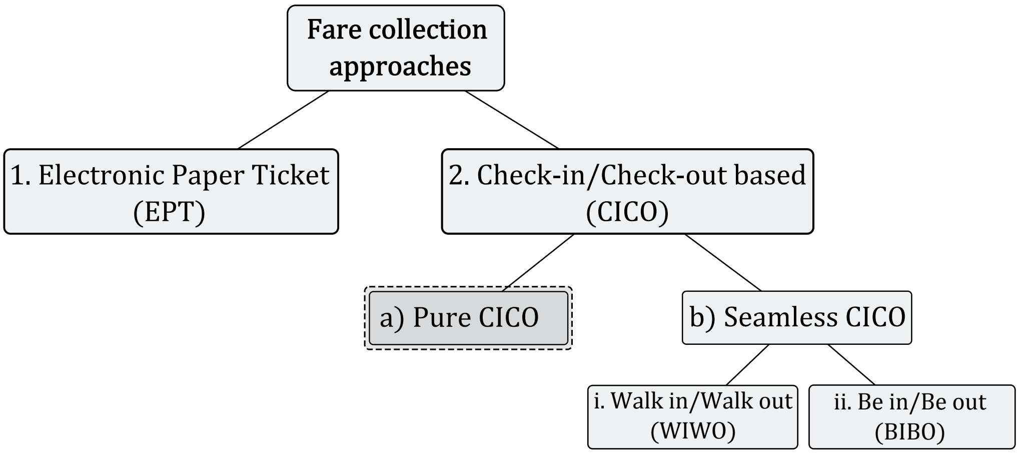

Fare collection approaches
1. Electronic Paper Ticket (EPT)
2. Check-in/Check-out based (CICO)
a) Pure CICO
b) Seamless CICO
i. Walk in/Walk out (WIWO)
ii. Be in/Be out (BIBO)

Figure 2.3: Approaches to fare collection in e-ticketing.

The first class of approaches collectively referred to as electronic paper ticket (EPT) considers the customer device (e.g., a smart card or a smartphone) essentially acting as an electronic purse, i.e. being a cashless alternative to the conventional paper-based approach [22]. EPT is considered to be rather inflexible compared to other approaches since in this case the price for a trip is usually zone dependent and cannot be directly inferred from the actual travel distance and other relevant parameters. Therefore, the check-out procedure (step *2b* in Figure 2.2) is usually not necessary[1]. An example of the public transport system using the EPT approach

[1] Some modified versions of this approach involve the check-out procedure to provide for rebates and to

is OstalbMobil [23] in Aalen, Germany.

In case of fare collection based on check-in/check-out (CICO), the events of a passenger entering and leaving the transport vehicle are registered by terminals. It enables the creation of flexible pricing schemes which operate on the respective travel information pertaining to a customer. Moreover, different loyalty programs can be developed more easily supporting fine-grained billing and other services.

In *pure CICO* (see approach 2*a* in Figure 2.3), *explicit* check-in/check-out on respectively entering and leaving the transport vehicle is carried out. This is usually performed by a user putting his/her device with an e-ticket in the vicinity of the dedicated area of a terminal (e.g., shortly touching the active surface). For example, the Octopus Card [24] in Hong-Kong and the Dutch OV-Chipkaart [25] are the public transport systems utilizing pure CICO for fare collection.

The approaches to fare collection termed *seamless CICO* within this dissertation in essence differentiate themselves from pure CICO mainly by the technical means of registering check-in/check-out events. As described in [26], walk in/walk out (WIWO) implies that the system automatically recognizes when a user gets in (walks in) and leaves (walks out) the vehicle. The procedure is performed in a wireless fashion and involves two electromagnetic fields. The low-frequency (LF) field is used for e-ticket activation as well as for determining the check-in/check-out events and for distinguishing between the two. The high-frequency (HF) field enables the activated e-ticket to subsequently send its ID, stop name, time stamp, etc. to the terminal. The duration of HF communication is determined by the time necessary to send the aforementioned information to the terminal and receive the acknowledgement (usually in the order of milliseconds). In case of be in/be out (BIBO), to the contrary, the on-board HF reader maintains the connection to the customer device with an e-ticket throughout the whole journey, i.e. since the e-ticket has been activated [26], hence the name "be in/be out". Both WIWO and BIBO aim at making the travel even more seamless and comfortable for a passenger. Therefore, they are referred to as seamless CICO in this dissertation.

Among the aforementioned approaches, seamless CICO (approach 2*b*) is the least standardized and not widespread despite several completed pilot projects (see [26, 27]). EPT (approach 1) is relatively inflexible and is likely to be replaced by pure CICO. The latter is widely used in ESPT around the world, for example, in the Netherlands (OV Chipkaart), Hong-Kong (Octopus card), etc. Moreover, CICO is backwards-compatible to EPT. Therefore, the dissertation will primarily focus on pure CICO (approach 2*a*). It should be further mentioned that for higher layers, it is essentially transparent whether pure CICO or seamless CICO is deployed since each of them is based on the same idea (processing the check-in/check-out events).

Non-interactive vs. Interaction-based Systems. Additionally, the following two types of ESPT can be distinguished with respect to the degree of user involvement into system information flow: *(1)* non-interactive and *(2)* interaction-based. The first type can be also referred to as an "honor-based" system implying that a user is not required to prove the possession of a valid travel permission each time on entering and leaving the vehicle (or alternatively, a transport system as in the case of a subway). Therefore, a customer is rather obliged to present the corresponding proof of possession of a respective travel permission in case of a spot check performed by a conductor. This model is fairly widespread in Germany, for example. Unlike

enhance loyalty schemes.

the non-interactive systems, the ones based on interaction (type 2) require that a customer explicitly performs check-in/check-out each time on entering/leaving a transport vehicle by this proving the validity of a travel permission. The latter type inherently allows for the implementation of flexible fare polices supporting on-demand payment (that is, a customer pays only for the transport service actually used). Moreover, an interaction-based ESPT is compatible to the non-interactive one but not the other way around. For this reason, the dissertation focuses on interaction-based systems.

The next section briefly discusses the underlying technology and the respective standards (mainly considering the front-end of an ESPT) which are most commonly used for implementing the e-ticketing paradigm in public transport scenario.

2.5 Underlying Technology and Standards

As already mentioned in the introduction, two technologies can be distinguished as the main enablers of e-ticketing systems for public transport (especially with respect to the front-end of ESPT), namely RFID and NFC. Despite many similarities enabling interoperability between these technologies, there also exist certain differences (especially concerning the protocol layer, data exchange formats and the respective standards).

The architecture of e-ticketing systems for public transport based on RFID/NFC essentially adheres to the three-tier model described in Section 1.2, namely it consists of the back-end, front-end, and the bridging element. For the back-end system, it is transparent how the respective information concerning each individual journey was acquired from the front-end part, i.e. whether the data were received from an RFID device (e.g., a contactless smart card) or from an NFC one (e.g., an NFC-capable smartphone). Therefore, the aforementioned differences between these two technologies are further discussed with respect to the front-end of an e-ticketing system.

2.5.1 Radio Frequency Identification (RFID)

The term RFID encompasses the whole family of contactless identification systems with the applications ranging from simple theft prevention systems to relatively profound electronic payment systems utilizing contactless smart cards (see, for example, [28] for more details). With respect to the e-ticketing systems discussed in this dissertation, the class of the so-called *proximity-coupling smart cards* is of particular relevance. Front-end of the majority of modern e-ticketing systems is based on this class of RFID systems most of which[1] comply with ISO14443 (A, B) [30]. The latter considers the first two layers of the OSI model [31], operates in the range of up to 10cm (hence proximity coupling) and consists of 4 parts:

Part 1: Physical characteristics;

Part 2: Radio frequency interface power and signal interface;

Part 3: Initialization and anticollision;

Part 4: Transmission protocol.

[1] There exists another important proprietary standard Felica from Sony Corporation [29]. It is used by the systems based on Sony FeliCa Cards. Despite the standardization attempts, Felica's recognition as an international standard failed.

The most widespread proximity smart card technologies used in e-ticketing systems comply with ISO14443, namely:

- *MIFARE* family: Developed by NXP Semiconductors and used in the following e-ticketing sytems: Dutch OV-chipkaart, London's Oyster Card, Hong Kong's Octopus Card, the Puget Sound ORCA Card in the US, etc. According to [32], more than 80% of all contactless smart cards in the world (as for 2011) are based on MIFARE technology.
- *Calypso* standard: Used in Calypso e-ticketing system (Belgium, Canada, China, France, Israel, Italy, Portugal, etc.) [33].

It should be mentioned that there is another prominent smart card technology which was initially applied to be included into the ISO14443 (as type C) but failed, namely *FeliCa* from Sony Corporation [34]. It is mainly used in e-ticketing systems deployed in Singapore (EZ-Link) and Hong-Kong (Octopus Card) and complies with Japanese Industrial Standard (JIS) X 6319 Part 4 [29]. Every smart card technology mentioned above is supported by NFC for compatibility and interoperability reasons[1] (see the discussion in Section 2.5.2).

On top of the communication interface, a number of higher level standards further defining data exchange and security-relevant functions is required (see Figure 2.4). Each of them is going to be briefly discussed during the overview of privacy issues in e-ticketing systems in Chapter 3. It should be mentioned that at the architecture layer (represented by ISO EN 24014-1), the conceptual issues with respect to the organisation of Interoperable Fare Management System (IFMS) are considered. The ISO EN 24014-1 standard, therefore, pertains to the back-end part of an e-ticketing system. The actual system implementation is left, however, to the public transport company and is not directly standardised. For the architecture layer, it is transparent which underlying technology is used for e-ticketing at the lower layers (e.g., RFID or NFC).

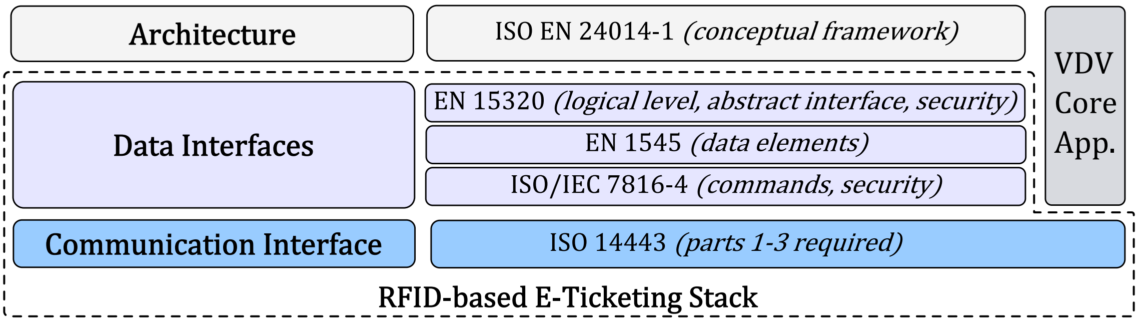

Architecture
ISO EN 24014-1 *(conceptual framework)*
VDV Core App.
Data Interfaces
EN 15320 *(logical level, abstract interface, security)*
EN 1545 *(data elements)*
ISO/IEC 7816-4 *(commands, security)*
Communication Interface
ISO 14443 *(parts 1-3 required)*
RFID-based E-Ticketing Stack

Figure 2.4: Standards supporting interoperable RFID-based e-ticketing systems.

In order to facilitate the development of *interoperable* systems for public transport and incorporate all the necessary standards in a single technical specification [35], the so-called Core Application (Kernapplikation) was created in Germany [36]. According to its developers, Core Application is generic enough to enable interoperability between different e-ticketing systems which comply with it (even interoperability on an international level is claimed to be possible). Core Application is, therefore, a generic middleware developed on top of the

[1] Therefore, an NFC-capable device (e.g., a smartphone) can be used in (e-ticketing) systems based on any of the three aforementioned influential smart card technologies.

communication interface (ISO 14443) which should provide users with the ability to seamlessly travel (e.g., using the same e-ticket in different cities) and enable transport companies to keep such an interoperable system in operation (inlcuding revenue sharing). Moreover, it is claimed to greatly improve the process of migrating from a set of proprietary solutions to a global one (see Figure 2.4).

2.5.2 Near Field Communication (NFC)

NFC is a technology closely related to RFID, namely to RFID operating at 13.56 MHz (proximity coupling systems due to ISO/IEC 14443, see Section 2.5.1, and vicinity coupling systems due to ISO/IEC 15693 [37]). The communication takes place within the near field region with respect to the operating electromagnetic field (13.56 MHz), hence the name Near Field Communication. The operating distance of NFC is up to 10cm[1]. NFC is compatible with the RFID technologies described in Section 2.5.1, namely MIFARE, Calypso and FeliCa [28].

NFC is being actively promoted by the NFC Forum [4] which is an important non-commercial organisation in the area of NFC. It develops the necessary technical specifications to ensure interoperability and worldwide acceptance of NFC and defines the so-called NFC Forum Architecture incorporating relevant standards and enabling the development of interoperable NFC applications (see further).

2.5.2.1 The Cornerstones of NFC

The radio frequency (RF) layer of the NFC Forum Architecture together with initialization schemes (involving data collision control) and transport protocol (including protocol activation and data exchange methods) are defined in ISO/IEC 18092 standard as NFC Interface and Protocol (NFCIP-1) [38]. This standard defines two communication modes: *active* and *passive*, see Table 2.1. The passive mode of NFCIP-1 is compatible with MIFARE and FeliCa (see Section 2.5.1) [39].

Table 2.1: The communication modes defined in NFCIP-1 (ISO/IEC 18092).

Modes	**Description**
Active mode:	Both the initiator (e.g., an NFC reader) and the target (e.g., an NFC smartphone) use their own RF field for communication
Passive mode:	The initiator generates the RF field, starts the communication and supplies the target with energy needed for generating the response sequence (using the load modulation scheme).

The NFCIP-2 standard (ISO/IEC 21481) [40] further combines the functionality of proximity-coupling devices (ISO/IEC 14443), vicinity-coupling devices (ISO/IEC 15693) together with the communication modes defined in NFCIP-1, and enables the compliant NFC device to

[1] NFC is in principle compatible with ISO/IEC 15693 (vicinity coupling) the communication range of which is around 1 m. This feature, however, is not the part of the NFC Forum Architecture being actively promoted by the NFC forum [4] (see further).

switch between these communication modes. The distinct communication modes supported by NFCIP-2 are listed in Table 2.2.

Table 2.2: The communication modes supported by NFCIP-2 (ISO/IEC 21481).

Mode	Initial standard
Proximity Coupling Device (PCD) *Proximity Integrated Circuit Card (PICC)*	ISO/IEC 14443
Vicinity Coupling Device (VCD)	ISO/IEC 15693
NFC	NFCIP-1 (ISO/IEC 18092)

The PCD/VCD mode defined in NFCIP-2 (see Table 2.2) is analogous to the initiator (e.g., a reader), see Table 2.1. The PICC mode is analogous to the target (e.g., a smart card) passively waiting for the initiator's requests and energy supply. The NFC mode refers to the communication modes (active and passive) defined by NFCIP-1.

NFCIP-2 specifies the mechanisms for detection and selection of one of the communication modes described above as well as the subsequent behaviour "... as specified in the standard specifying the selected communication mode" [40]. Therefore, an NFCIP-2 compliant device can act both as the initiator and the target depending on the device being communicated with (and on the agreed communication protocol).

2.5.2.2 The NFC Forum Architecture

The aforementioned core standards (NFCIP-1,2) are the basis for the development of further cross-layer specifications for NFC communication. In order to enable the interoperability for the NFC-based systems and backward compatibility with the proximity-coupling RFID systems (ISO/IEC 14443, FeliCa), the NFC Forum [4] developed the so-called *NFC Forum Architecture* (see Figure 2.5). It should be mentioned, however, that currently the vicinity-coupling technology (ISO/IEC 15693) is not included into the NFC Forum Architecture. The possible reason may be to keep the operating range within around 10 *cm* (since the operating range of vicinity-coupling systems is in the order of 1 *m*) and by this to ensure the "touch-to-activate" principle for service discovery, decrease the collision probability between NFC devices as well as to provide better security.

The NFC Forum Architecture supports 3 operation modes:

1. **Peer-to-Peer mode**. Defines the communication between two NFC devices.

   Underlying basics: Defined by NFCIP-1.

   Applications: data transfer between two NFC devices, exchange of configuration parameters (e.g., WiFi/Bluetooth pairing), etc.

2. **Reader/Writer mode**. Defines the communication between an NFC device and a passive transponder. The latter may be compatible with the 4 types of NFC Forum Tags

(see Table 2.3 further in the section) or with other contactless chip cards. This mode is backwards-compatible with existing smart card infrastructures [39].

Underlying basics: Defined by NFCIP-1 and RFID standards: ISO/IEC 14443, JIS X 6319-4 (FeliCa).

Applications: content distribution, information access (smart posters), smart advertising [41] (reader mode), custom tag writing (writer mode).

3. **Card Emulation mode**. Enables an NFC device (appearing as a contactless smart card) to communicate with RFID readers. It is, therefore, backwards-compatible with existing smart card infrastructures.

   Underlying basics: The same as for the Reader/Writer mode.

   Applications [41]: mobile payment, ticketing, access control, top-ups, toll-gate, etc.

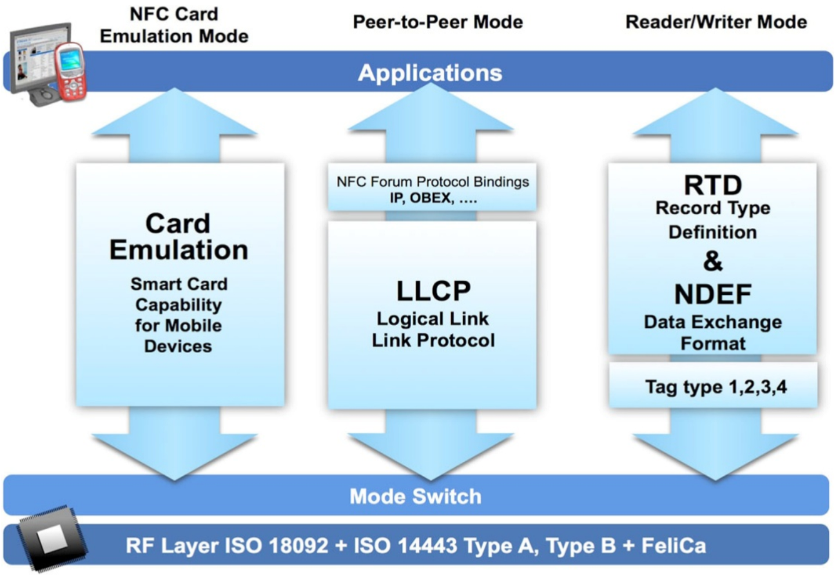


Figure 2.5: The NFC Forum Architecture. Taken from [4].

The selection of the required operation mode (and consequently the required mode at the RF layer) is performed by the *mode switch* procedure at the respective Mode Switch layer of the NFC Forum Architecture (see Figure 2.5).

In order to provide the necessary prerequisites for the interoperable behaviour of the mode switch procedure, the so-called *Digital Protocol Specification* [42] was developed by the NFC forum. This specification is applied between the RF layer and the Mode Switch layer of the NFC Forum Architecture (being a part of both of them) and considers the digital part of the physical layer (the digital interface) and the MAC layer (of the OSI architecture). It, therefore, implements NFCIP-1 incorporating ISO/IEC 14443 and narrows down the options in the underlying base specifications to ensure interoperability [43].

At the Mode Switch layer, the *Activity Specification* [44] of the NFC Forum describes how the NFC Digital Protocol can be used to set up the communication protocol with the other device through the definition of *Activities*. They combine elementary blocks of the Digital Protocol into the *functional* ones [44]. Activities are then combined in *Profiles* each of which in turn has specific *Configuration Parameters* and covers a particular use case. Therefore, according to [44], the combination of Activities and Profiles defines a predictable, deterministic behaviour of the NFC Forum device. This however does not impose constraints on the implementation of other building blocks (or the definition of other Profiles) for the use cases other than the existing ones. The following Activities are considered in the Activity Specification [44]:

- Technology Detection Activity;
- Collision Resolution Activity;
- Device Activation Activity;
- Data Exchange Activity;
- Device Deactivation Activity;

The two lower layers of the NFC Forum Architecture (the RF layer and the Mode Switch layer) define a common interface incorporating the underlying communication technologies and making it transparent for the upper layers which particular technology (e.g., due to ISO/IEC 14443-A or FeliCa) and in which mode is used for connection establishment (i.e. for low-level communication). The further higher-level specifications are targeted at each particular operating mode.

For the ***Peer-to-Peer (P2P)*** mode, the *Logical Link Control Protocol (LLCP)* Specification [45] was developed to establish and maintain the logical link between two NFC Forum devices. The main features provided by LLCP are:

- Link Activation, Supervision, and Deactivation;
- Asynchronous Balanced Communication (for providing peer-to-peer capabilities in contrast to the default "Initiator-Target" behaviour, hence "balanced" communication);
- Protocol Multiplexing (the ability to accommodate several instances of higher level protocols at the same time);
- Connectionless/Connection-oriented Transport (unacknowledged/acknowledged data transmission, respectively).

The LLCP resides between the Mode Switch and the Application layer of the NFC Forum Architecture [43]. On top of LLCP, different protocols may be operating:

- *Simple NDEF Exchange Protocol (SNDEF).* This NFC Forum protocol [46] can be used to exchange the so-called NFC Data Exchange Format (NDEF) messages in P2P mode. The NDEF Specification [47] defines the NDEF data structure format, the rules to construct a valid NDEF message as well as the mechanisms for specifying the types of application data encapsulated in NDEF records (in an interoperable manner and completely transparent with respect to the type of NFC device or tag in use). The detailed description of record types being exchanged through NDEF is defined in separate specifications.
- *Protocol bindings.* Provide standard bindings to NFC Forum protocols and allow interoperable use of registered protocols [32], for example, OBEX, IP [32].
- *Other protocols.* Other protocols which may run over the link layer provided by LLCP and for which no LLCP bindings from the NFC Forum are provided.

Further up the P2P stack, at the application layer, the NFC Forum reference applications may be run (over SNDEF) together with other P2P applications (e.g., printing an image taken by the smartphone camera, third party NDEF applications) [48].

The ***Reader/Writer*** mode is used for communicating with *(1)* NFC Forum Tags (involving NDEF Applications) or *(2)* for other vendor-specific applications which are not based on the NDEF Specification (e.g., reading a balance of an electronic purse or getting information from an e-ticket). In the first case, on top of the Digital Protocol, the NFC Forum specifies 4 tag types (see Table 2.3) with the respective operations [49]. Information can be read/written from/to these tags using the NDEF data format with records defined according to the Record Type Definition (RTD) specification [50]. The latter defines how to construct records in NDEF messages (e.g., records of type Text, URI, Smart Poster, etc.). Various NDEF applications (both NDEF Reference and third-party ones) can run on top of the NFC Tag Specifications (e.g., smart poster, reading product information from products/product flyers equipped with NFC tags, etc.)

Table 2.3: The NFC Forum tag types. Based on [7].

Tag Type	**Standard**	**Memory**	**Com. Speed**, kbit/s
Type 1	ISO14443-A	Read and re-write capable, can be configured to be read-only; memory size 96 byte (expandable to 2 kB)	106
Type 2	ISO14443-A	Read and re-write capable, can be configured to be read-only; memory size 48 byte (expandable to 2 kB)	106
Type 3	FeliCa	Pre-configured (at man. time) to be either read and re-writeable, or read-only; memory is variable, theoretically limited by 1 MB per service	212/424
Type 4	Fully compatible with ISO14443-A,B (”open tags”)	Pre-configured (at man. time) to be either read and re-writeable, or read-only; memory is variable, theoretically limited by 32 kB per service	≤ 424

In ***Card Emulation*** mode, the proprietary contactless card applications (e-ticketing, access control, payment, etc.) are executed directly on top of the Digital Protocol (between the RF layer and the Mode Switch Layer in Figure 2.5) enabling to emulate a contactless smart card based on ISO/IEC 14443 of Type A,B and JIS X 6319-4 (FeliCa). These applications are usually security-critical (payment information, etc.). Therefore, the sensitive part of the application (e.g., secure storage of cryptographic keys, execution in the secure environment) usually resides in the so-called Secure Element (SE). The latter can be of the following types [28]:

- *(1) Hardware embedded into the phone* (i.e. soldered into the NFC mobile phone's circuitry);
- *(2) Secure Memory Card* (e.g., Secure Digital (SD) card). It is removable and therefore not bound to a single device;
- *(3) (Universal) Subscriber Identity Module ((U)SIM).* The SE based on (U)SIM is compliant with smart card standards and can host multiple applications issued by different application providers [32]. Moreover, the secure NFC applications stored in the (U)SIM-based SE can be remotely managed via the so-called Over-the-Air (OTA) technology (using the commands encapsulated in SMS messages) [32].

The most widespread technical means of implementing the interface between the SE and the NFC controller are the following [28]:

- *Single Wire Protocol (SWP).* Interconnects an NFC Contactless Front-end (CLF) as master and a SE as slave via a single-wire transmission. It is standardized in [51] and mainly intended for (U)SIM (see above, SE type 3) since there is a single free contact left on a (U)SIM card (out of the standard 8) which can be utilized for this function [28]. The transmission rates range from 212 kbit/s to 1.6 Mbit/s [32].
- *NFC Wired Interface (NFC-WI)* (also known as S^2C interface). Interconnects the SE with the NFC front-end utilizing two wires. It is standardized in [52] and supports the transmission rates of 106, 212, and 424 kbit/s.

In case a (U)SIM is used as a SE, the power required for SWP to operate can be supplied via the NFC interface (i.e. from the reader side). It, therefore, renders the need for an external energy source (e.g., a smartphone battery) unnecessary, in order to be able to run the respective applications (e.g., e-ticketing) residing in the (U)SIM memory in card emulation mode [28].

2.5.3 Underlying Technology and Standards: A Short Summary

The main enablers of the e-ticketing paradigm were reviewed in this section, namely RFID and NFC. More specifically, the respective standards have been reviewed and analyzed above. A holistic overview of the standardisation effort in the area of e-ticketing systems based on RFID and NFC is provided in Figure 2.6.

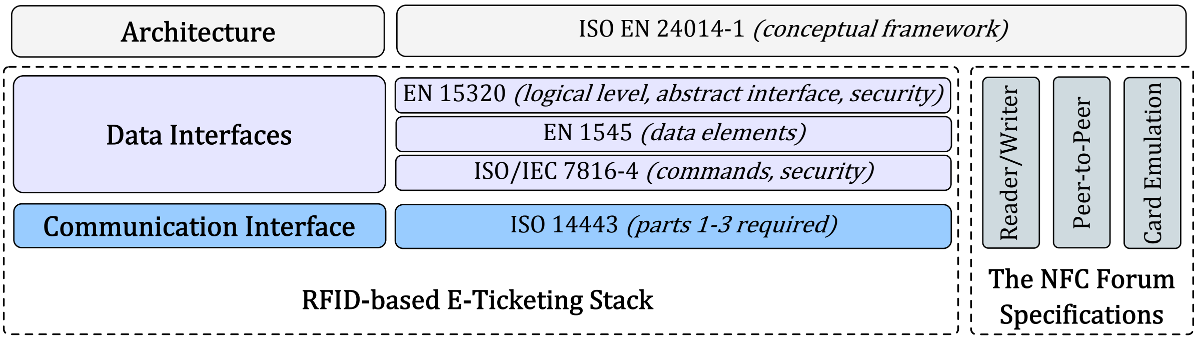

Architecture
ISO EN 24014-1 *(conceptual framework)*
Data Interfaces
EN 15320 *(logical level, abstract interface, security)*
EN 1545 *(data elements)*
ISO/IEC 7816-4 *(commands, security)*
Communication Interface
ISO 14443 *(parts 1-3 required)*
RFID-based E-Ticketing Stack
Reader/Writer
Peer-to-Peer
Card Emulation
The NFC Forum Specifications

Figure 2.6: Standards stacks for e-ticketing based on RFID and NFC.

2.6 Focus on Customer Privacy

Having briefly analysed the implications of e-ticketing systems, the following conclusion regarding security and privacy can be made. The interoperability goal implies the existence of common security and privacy measures (e.g., an agreement on mutually recognized and accepted security and privacy policies). The need for security is widely acknowledged by transport companies, since insecure solutions may result in substantial revenue losses (e.g., due to ticket forgery or system blackouts) and even lead to the eventual phase-out of the system.

Privacy, namely the customer privacy, to the contrary, is not in direct interest of service providers. The reason for this is that possible risks associated with privacy violation have far less serious implications for company business compared to security. However, the constantly rising privacy-awareness of customers and the ever growing likelihood of public outcry induced by the cases of privacy violation may stimulate transport companies to invest in privacy in order to remain competitive. The interoperability goal poses a further challenge to privacy since sharing of privacy-critical data, which is needed for a proper delivery of transport services by cooperating companies, should be performed in a privacy-preserving way.

As a consequence, the concerns over the customer privacy (see Section 2.1.2) have a negative impact on public acceptance of interoperable e-ticketing systems which inevitably influences the revenues of public transport companies. A privacy-preserving solution, therefore, would be beneficial for both customers and transport service providers. Thus, this thesis aims at providing such a solution which can be directly applied for the development of e-ticketing systems with privacy-preserving mechanisms being integrated from the outset.

The next chapter focuses on privacy issues in e-ticketing systems discussing the privacy threats together with the privacy-preserving mechanisms foreseen by the respective standards and implemented in a number of currently deployed e-ticketing systems.

Chapter Summary

In this chapter, the e-ticketing paradigm together with its advantages and disadvantages was discussed. Moreover, main functional types of an e-ticket together with the most relevant application areas were presented. The general application scenario of ESPT which is in focus of this dissertation was discussed as well. Moreover, different fare collection approaches in e-ticketing were presented focusing on the check-in/check-out concept. The main enablers of the latter (RFID and NFC) in ESPT scenario together with the respective standards were also discussed. The chapter is concluded with raising a question on privacy friendliness of such systems which is going to be discussed in detail within the next chapter.

3 Privacy Issues in E-ticketing Systems: Overview and Requirements

Having discussed the foundations of e-ticketing systems for public transport (ESPT) in the previous chapter, the thesis focuses on the issues of privacy protection in this area. In this chapter, the notion of privacy is shortly formalized at first in Section 3.1. The generic privacy threats endemic in ESPT systems are discussed in Section 3.2 together with the respective countermeasures presented in Section 3.3. The availability of privacy-preserving mechanisms within the respective standards comprising RFID and NFC stacks is discussed in Section 3.4. Core requirements for ESPT as well as the adopted attacker model are discussed in Section 3.5 and Section 3.6 respectively.

3.1 Privacy Analysis: Prerequisites

The user's privacy is a fairly ambiguous notion which is often understood in a number of different ways depending on the culture, country's political system, etc. Much effort from different fields of science has been spent within the context of privacy *per se* and its perception in society. In order to approach the problem from a technical point of view, a set of generic privacy properties similar to the classic CIA triad in information security (confidentiality, integrity, availability) is required. This would further enable to derive the privacy requirements and to perform the privacy analysis of a system under concern. Therefore, the following notions considering the user's privacy from a technical perspective are defined below:

- *Pseudonymity* (to ensure accountability in contrast to Anonymity);
- *Confidentiality*;
- *Unlinkability.*

Definition 1: *Pseudonymity* enables the communicating entities to perform the necessary information exchange without disclosing their Personally Identifiable Information (PII) during the communication session. In case the exchanged information is persistently stored, its pseudonymised form should prevent malicious parties from illegal identification of the respective entities. It is, however, possible to perform *subsequent identification* by a special entity with respective authorisation to ensure accountability (e.g., for billing purposes).

Definition 2: *Confidentiality* of information exchanged between communicating entities, which can also be persistently stored, ensures that the content of such a conversation (especially with respect to the identifying information) is disclosed only to the legitimate parties possessing the respective authorisation (e.g., the ones being authorised to use the respective message decryption key).

Definition 3: *Unlinkability* prevents a malicious party from performing linkage of different pieces of information[1] which pertain to a certain entity and are distributed in time and/or space and therefore from illegally obtaining the entity's PII.

A set of the three notions defined above describes the basic technical aspects of the user's privacy and can be used, therefore, as the necessary basis for privacy analysis of an e-ticketing system.

3.2 Generic Privacy Threats in E-ticketing Systems

Analysing the e-ticketing systems under concern (see Section 2.3) against pseudonymity, confidentiality and unlinkability defined in the previous section, the following generic threats to the user's privacy can be identified in the e-ticketing environment:

1. Unintended customer identification:
    a) Exposure of the customer ID:
        i. Personal ID exposure (direct identification),
        ii. Indirect identification through the relevant object's ID[2] [53].
    b) Exposure of a static identifier (e.g., during the anti-collision session [35]);
    c) Physical layer identification (RFID fingerprinting[3]).
2. Information linkage;
3. Illegal customer profiling.

The first subset of the aforementioned privacy threats (*Unintended customer identification*) pertains to the front-end of an e-ticketing system. It mainly considers the process of e-ticket authentication during check-in/check-out events (see Figure 2.2) when the respective trip tuples are formed (containing user and terminal IDs, location, time, etc.). Moreover, the vulnerabilities implied by these threats can be potentially exploited by any properly equipped passer-by (e.g. with a portable reader).

The possible attack scenarios in this case are the following:

- Intervening with the RF interface between the e-ticket medium and the (honest) terminal:
    - Communication eavesdropping;
    - Relay attacks;
- Unintended interaction with the e-ticket medium (also outside the specifically designed locations for check-in/check-out) in order to compromise the privacy of its owner (Threats 1b, 1c).

[1] Such pieces of information can originate *inter alia* from the digital traces left by a person.

[2] The notion of object ID (OID) encompasses the following ID set: medium ID (e.g., unique card number), application ID (the unique identifier of an application instance installed), etc. OID can, therefore, become an indirect personal identifier [53].

[3] For example, using the deviations in the backscatter frequency of an RFID chip as a distinguishing factor, see [54].

- Spoofing the e-ticket medium into interacting with a malicious reader presenting itself as a legitimate terminal.
- Compromising the legitimate terminal (e.g., to mount replay attacks).

The user's privacy violation resulting from *information linkage* (Threat 2) can take place when various pieces of information directly and indirectly pertaining to a user are combined together in order to obtain an *identifiable* piece of information. The latter can be subsequently used for committing privacy violating actions. The information traces "left" by a user partially originate as a result of the vulnerabilities collectively referred to as *unintended customer identification* (Threat 1) in this section.

Illegal customer profiling (Threat 3) considers the creation of users' profiles which is not prescribed by the system specification and therefore is not required for maintaining system functionality and proper service delivery. In contrast to the customer profiling needed for e.g., the implementation of different loyalty schemes (which should be well documented, certified and implemented in a privacy-respecting way), the processes collectively referred to as illegal customer profiling violate the privacy regulation and endanger the users' privacy. One of the examples of this would be selling of the sensitive user data to the third parties for marketing purposes.

3.3 Generic Privacy Threats: Possible Countermeasures

The aforementioned privacy threats endemic to e-ticketing systems can be mitigated against using specific countermeasures. In order to approach the problem and to consider specific countermeasures against each privacy threat described in Section 3.2, a preliminary analysis was conducted resulting in the countermeasures set, see Table 3.1. However, an efficient application of such countermeasures during the system development as well as their optimal distribution across system components is to a large extent an open research question. Currently, privacy-preserving techniques being developed by the research community as well as the ones implemented in real public transport systems are for the most part tailor-made and solve a specific privacy-related problem. An approach which would target the privacy of users from the outset in a *holistic way* treating all components[1] of a public transport system in their entirety is, however, still missing. Moreover, it is questionable whether a solution addressing all possible privacy threats ranging from the physical layer (e.g., RFID fingerprinting) to the application layer (e.g., user ID exposure) can stay efficient and what is even more important compatible with the existing technologies currently embraced by the industry.

Therefore, it would be desirable to analyze to which extent privacy protection is addressed within the widely adopted standards specifying the implementation of core enablers of e-ticketing systems (namely, RFID and NFC, see Section 2.5). This is performed in the next section and presented in a top-down manner beginning from the upper layers of the respective standards stack.

[1] The focus is made on the functional components of a public transport system which can be generally divided into back-end, front-end, and the so-called bridging element (e.g., readers/terminals).

Table 3.1: Countermeasures against the generic privacy threats identified in the e-ticketing systems under concern.

Threats	Countermeasures
1. Unintended customer identification:	
a) *Exposure of the customer ID:*	
i. Personal ID exposure (direct)	Privacy-respecting authentication; ID encryption/randomization; access-control functions [55]
ii. Indirect identification	ID encryption
b) *Unencrypted ID during anti-collision*	Randomized bit encoding [56]; bit collision masking [57, 58] (protocol dependent)
c) *PHY-layer identification*	Shielding; switchable antennas [59]
2. Information linkage	Anonymization (in the front-end and in the back-end): threat 1 countermeasures; privacy-respecting data processing
3. Illegal customer profiling	Privacy-respecting data storage (especially in the back-end); the same as in threat 1

3.4 Privacy Issues Considered in the Respective Standards

The e-ticketing systems under concern are based on RFID and NFC technologies (see Section 2.5). Despite obvious similarities between them, each technology possesses its own set of distinctive features and, therefore, needs to be separately analysed for privacy issues. Thus, Section 3.4.2 examines the RFID-based standards stack for e-ticketing systems (see Figure 2.4). Section 3.4.3 contains the analysis of the generic privacy-enhancing mechanisms implemented in the NFC Forum Architecture (see Figure 2.5). We start this analysis, however, with the architecture layer common to both technologies, which is considered in the next section.

3.4.1 Architecture Layer

The Architecture layer of an interoperable e-ticketing system for public transport is specified in ISO EN 24014-1 [18]. The standard introduces a conceptual framework for developing an interoperable architecture for transport fare management systems, which are collectively called Interoperable Fare Management System (IFMS). It describes the structure of an interoperable platform, its main actors, and general flows of information exchange. Privacy is considered at a conceptual level by requiring the definition of a security scheme that should provide for privacy protection (along with ”integrity and confidentiality between the actors to ensure fair and secure data flow within the IFMS” [18]). The security-related measures are defined in the respective security policy. Security management is performed by the Security Manager entity who is responsible for the implementation of the security policy by all actors concerned.

The standard prescribes that the privacy of a customer must be protected ”as required by applicable laws” specifying the following rules [18]:

- Only relevant personal data needed for the operation of the IFMS shall be requested from the customer [the classic data minimization principle];
- The itemised disclosure of service consumption on an invoice shall be an option that can be chosen by the customer;
- An IFM actor may not disclose customer-related information to third parties without specific authorisation from the customer [user consent].
- Within the IFMS, the customer-specific data shall be handled only in connection with the identification number of the contract (implicit or explicit) between the customer and product owner. A link between the contract number and the name of the customer may only be achieved by the contractual partner at the request of the customer.

As it can be seen, the ISO EN 24014-1 standard rather coarsely specifies the privacy-related requirements which partially cover information linkage and illegal customer profiling (privacy threats 2 and 3, see section 3.2). The standard, however, does not provide any detailed recommendations concerning the further implementation of these requirements.

3.4.2 RFID-based Standards Stack

The assessment of privacy issues considered at each layer of the RFID-based standards stack (see Figure 2.4) is performed in this section in a "top-down" manner. Firstly, the uppermost layer – *the Data Interfaces layer* – is considered.

3.4.2.1 Data Interfaces Layer

In this section, the Data Interfaces Layer which consists of three sub-layers represented by the respective standards (see Figure 2.4) is considered.

EN 15320 The standard defines the logical structure of the data residing in the card, specifies an abstract interface for interaction between the card and the terminal (which consists of two logical interfaces: the Card Data Interface and the Data Group Interface) and considers security through specification of the Security Subsystem (SSS). The latter is divided into the Card Security Management System and the Data Group Security Management System in order to correspond to the two logical interfaces. Security-related operations are defined in profiles (Card Profiles and Data Group Profiles respectively), see Figure 3.1.

The privacy-related issues are considered only indirectly in EN 15320 through the description of data groups containing privacy-relevant information (e.g., the *card holder* data group). If such data group is present, the necessary access control mechanisms together with encryption should be implemented in order to protect the customer's privacy (i.e. be included into the respective profiles of the SSS).

The division into two logical interfaces (Card Data Interface and Data Group Interface) provides flexible implementation of access control schemes[1]. Specific security-related

[1] For example, in order to quickly and efficiently perform a ticket validation procedure, only the Card Data Interface is used, which speeds up the processes and saves resources.

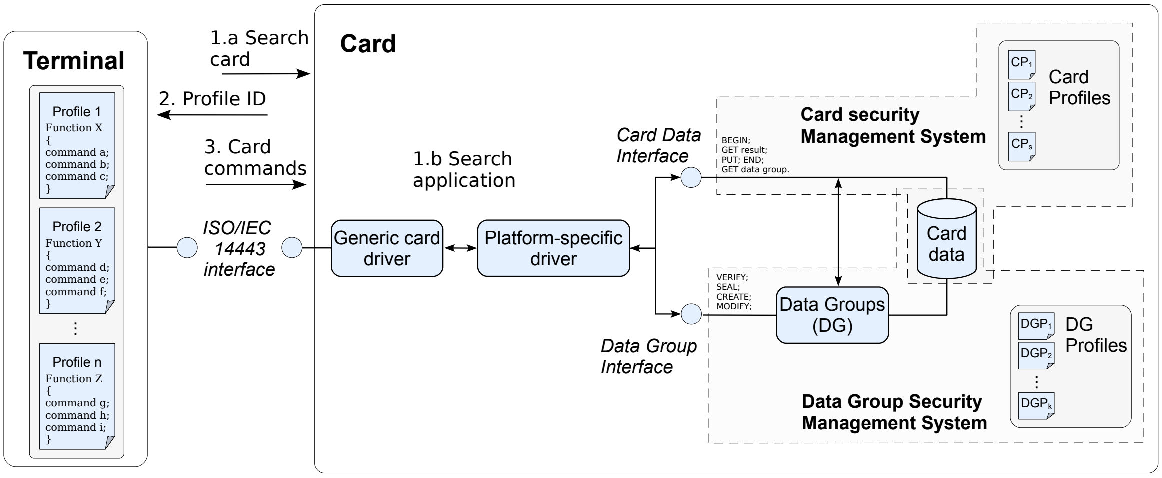


Figure 3.1: Interaction between a terminal and a card. Based on the processes description specified in EN 15320.

operations can be defined in the respective profiles and called whenever it is necessary to ensure proper execution of application commands. This mechanism may be used for processing of personal data in a privacy-respecting way, therefore, providing protection against unintended customer identification (namely, personal ID exposure and object's ID exposure, see section 3.2, threats 1(a)i and 1(a)ii respectively). The standard, however, does not explicitly address customer privacy and focuses solely on security issues.

EN 1545 **Part 1** The structure of data elements residing in the card is considered, which is expressed according to ASN.1 (Abstract Syntax Notation 1). Privacy-relevant information is contained in several data elements presented in Table 3.2. These data can be protected by applying encryption and access control schemes defined at a logical level in the respective profiles of the security subsystem (SSS, see EN 15320 above) therefore covering the issues of personal ID exposure and object's ID exposure (privacy threats 1(a)i and 1(a)ii respectively, see section 3.2).

Part 2 Data structures residing in the card are further specified according to the requirements of an interoperable fare management transport system (i.e. the requirements specified at higher layers of the standards stack, see Figure 2.4). This part of the standard focuses solely on the functional issues of a transport system and does not consider privacy and security.

ISO/IEC 7816-4 The standard considers the issues of commands exchange as well as the retrieval of data structures and data objects residing in the card. Security and privacy are taken into account by specification of methods for secure messaging and a security architecture which defines access rights to files and data in the card. Access methods to the algorithms processed by the cards are considered as well [60].

Table 3.2: Privacy-relevant fields in EN 1545-1.

Privacy-relevant field	**Description**
birth date	-
birth name	-
birth place	-
customer number	*customer reference number*
device ID	*can be linked to a particular customer*
e-mail address	-
telephone number	-
postal address	-
location ID	-
customer profile ID	*e.g. student, military, resident, etc.*
user data	*additional information about a customer*

3.4.2.2 Communication Interface Layer

The Communication Interface layer is represented by ISO 14443, which consists of four parts [30]. Parts 1-3 are required for connection establishment between the card and the terminal. Part 4 is optional and usually used for the cards with relatively high processing power. The standard does not consider any security- or privacy-related issues and focuses solely on functionality. Therefore, the issues of unintended customer identification during the anti-collision session as well as physical layer identification[1], which could be covered within the communication interface level, remain unconsidered.

Privacy Issues Considered in the RFID-based Stack: A Short Summary

Summarizing, the standards composing the generic e-ticketing system based on the RFID technology primarily consider security for protection of transport companies' assets and maintaining the proper and reliable system functionality. The issues of customer privacy are seen more as a by-product of security without the additional measures specifically targeted at ensuring the privacy-respecting behaviour of the system. Table 3.3 summarizes the security and privacy measures considered by each standard in the standards stack depicted in Figure 2.4.

3.4.3 The NFC Forum Architecture

There are two NFC modes defined by the NFC Forum Architecture (see Figure 2.5) which are usually considered in the first place for e-ticketing applications: *card emulation* mode and *peer-to-peer* mode. The former incorporates the existing RFID standards and therefore is succumb to similar privacy threats as in the case of RFID stack described above. The P2P mode, to the contrary, is NFC-specific having its own security (and hence privacy[2]) relevant mechanisms considered in the respective standards, which are going to be addressed in this section.

[1] See privacy threats 1b and 1c in section 3.2.

[2] Security can be viewed as the underlying basis for implementing privacy-preserving mechanisms.

Table 3.3: Security and privacy measures available in the e-ticketing standards stack based on RFID.

Standard	**Security**	**Privacy**
ISO EN 24014-1	- definition of security policy; - security management (by the Security Manager entity).	coarsely specified privacy requirements, targeted at compliance with the regulation
EN 15320	- Security Subsystem (SSS); - security-related operations are defined in profiles.	- privacy-relevant data groups; - protection through access control (AC) and encryption.
EN 1545	- security-relevant fields	privacy-relevant fields, see Table 3.2
ISO/IEC 7816-4	- secure messaging; - security architecture with AC	security mechanisms can be applied to privacy-critical data
ISO 14443 (1-3)	not considered	not considered

Legend:
⬜ Architecture Layer
🟪 Data interfaces Layer
🟦 Communication Interface Layer

3.4.3.1 P2P Mode

In order to address security issues in NFC communication, a series of NFC security standards (NFC-SEC) was developed on top of NFCIP-1. It defines an application independent security protocol stack which enables link layer encryption and offers a key exchange mechanism for upper layer (application-specific) encryption schemes [61]. NFC-SEC standards series consist of a Common framework defined in ECMA-385 standard [62] and Cryptographic mechanisms (used in the framework) defined in ECMA-386 standard [63]. The Common framework referred to as NFCIP-1 Security Services and Protocol considers the security services (Shared Secret Service and Secure Channel Service), the Protocol Data Units (PDUs), and the security protocol itself [61]. This framework is complemented by cryptographic mechanisms specified in [63] as NFC-SEC-01: NFC-SEC Cryptography Standard using ECDH and AES[1]. Further cryptographic mechanisms may be standardized for the NFC Common Security Framework, which will then complement it with the respective number, e.g., NFC-SEC-xy. NFC Security Stack is depicted in Figure 3.2.

NFC-SEC: The Services

NFC-SEC considers two services: the Shared Secret Service (SSE) and the Secure Channel Service (SCH). The former provides a generic mechanism for key exchange which can be used by application-specific encryption algorithms at higher levels. The SCH Service, however, along with the key exchange establishes a secure session (on the link layer) between two NFC peer devices.

[1] ECDH and AES stand respectively for the Elliptic Curves Diffie-Hellman (ECDH) protocol for key agreement and the Advanced Encryption Standard (AES) algorithm for data encryption and integrity.

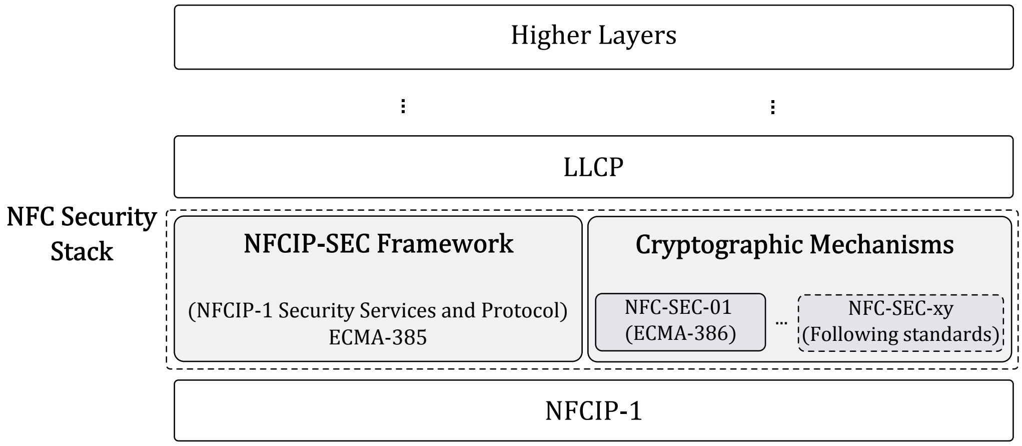


Figure 3.2: NFC Security Stack.

NFC-SEC: The Protocol

The protocol of NFC-SEC defines a sequence of steps for key agreement and confirmation (common for SSE and SCH) as well as the "PDU security" – the actual data encryption and integrity checks (specific to SCH). PDUs of NFC-SEC are encapsulated in Data Exchange Protocol (DEP) packets of a lower layer NFCIP-1 standard (see Figure 3.2).

NFC-SEC-01: Cryptographic Mechanisms

NFC-SEC-01 further specifies [61]:

- Message contents with concatenation rules for keys and other fields;
- Key primitives;
- Random number requirements;
- Conversion and transformation rules;
- Cryptographic algorithms and methods.

For key exchange, the Elliptic Curve Diffie-Hellman (ECDH) mechanism is used (security parameter 192 bit). Advanced Encryption Standard (AES) is used for *(1)* key derivation and confirmation, *(2)* data encryption, and *(3)* data integrity (security parameter 128 bit).

The mechanism, however, does not protect from the Man-in-the-Middle attack since no entity authentication can be provided by NFC-SEC [61].

3.4.3.2 Open NFC Security Stack

Another interesting approach towards NFC security rather complementing the standardised framework considered in the previous section is a Security Stack from Open NFC™ Open Source Project [64]. The Open NFC Security Stack was designed to protect access to the SE (such as (U)SIM, etc.) from unauthorized device applications. It essentially enforces a specified access policy to the SE which is done using Access Control Lists derived from the so-called PKCS#15 applet, see the respective specification [65]. This applet contains an access control file (ACF) where access rights are specified. The security issues pertaining to the RF interface of NFC are beyond the scope of the Open NFC Security Stack [66].

Privacy Issues Considered in the NFC Forum Stack: A Short Summary

Summarizing, privacy issues are indirectly addressed within the NFC Forum Stack by introducing the NFC-SEC standard series. It enables to secure the communication channel between two NFC devices as well as to exchange a secret for custom cryptographic algorithms at the application layer. Therefore, unintended customer identification at the link layer and higher can be prevented (threats 1(a)i, 1(a)ii, see Table 3.1). The problems of ID exposure during the anti-collision session and PHY-layer fingerprinting are, however, not covered (threats 1b, 1c in Table 3.1). Since NFC-SEC is vulnerable to Man-in-the-Middle attacks, entity authentication should be additionally implemented on top of it in order to protect the privacy-critical data from misuse during the communication session.

Privacy Issues Considered in the Respective Standards: A Summary

Privacy protection has been neither directly addressed within the RFID stack nor within the NFC one. A common trend is that security issues are explicitely considered (ISO 7816-4 with secure messaging in case of RFID and NFC-SEC specification in case of NFC) with privacy protection being rather a by-product (if at addressed all).

The aforementioned analysis provides a motivation to explicitly address the issues of privacy protection in the area of e-ticketing systems for public transport (ESPT). In order to devise an appropriate solution, privacy requirements have to be determined at first taking into account the specifics of ESPT as well as other important functional requirements (e.g., the support for fine-grained billing) which comprise a set of core advantages of e-ticketing. The next section focuses on this issue.

3.5 Core Requirements

Fully fledged requirements analysis is a complicated process which should be carefully considered in its own right. For the current thesis which specifically focuses on privacy protection in ESPT, only the most essential (technical) requirements are considered enabling analysis of the related work and development of the solution. Therefore in what follows, the discussion of the most pertinent requirements is provided leaving the holistic process of fully fledged requirement analysis out of scope. The requirements discussed below are summarized in Section 3.5.5.

3.5.1 Privacy

Privacy protection should be applied across system components taking into account the system in its entirety as well as the external entities which are not a part of the system, see Figure 3.3. Accordingly, three core privacy requirements can be distinguished:

1. Privacy against external observers;
2. Privacy against terminals (front-end);
3. Privacy against the back-end.

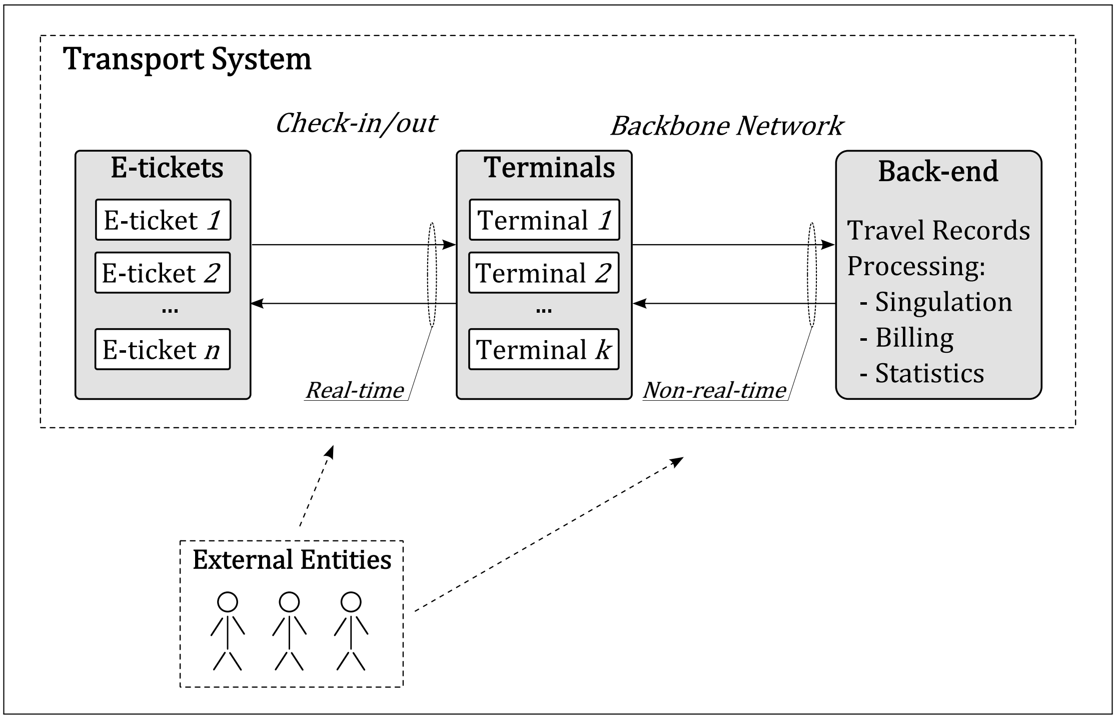


Figure 3.3: A high-level overview of the system architecture together with external entities.

3.5.1.1 Privacy Against External Observers

A substantial set of attacks targeted at privacy invasion are carried out by external entities not directly involved into the system information flow. Such adversarial entities exogenous to the system are collectively referred to as external observers in this dissertation, see the attacker model in Section 3.6. The question of privacy protection against such kind of attackers is especially acute in system front-end due to the wide distribution of terminals and contactless communication between e-tickets and terminals (which is inherently easy to intercept). Therefore, no external entity should be able to derive any identifying or tracking information pertaining to an e-ticket or its user by observing the communication session between terminals and e-tickets. It is implicitly assumed that the communication between terminals and the back-end (see Backbone Network in Figure 3.3) is reliably secured using well-established techniques such as TLS, etc. Therefore, with this respect, the dissertation focuses mainly on the front-end of the system.

3.5.1.2 Privacy Against Terminals (Front-end)

Due to the wide distribution of terminals across the transport network, they are not always under physical control of a transport authority (TA). Moreover, the wireless interface used for ticket validation can be misused by third parties representing an additional attack vector (e.g., a buffer overflow attack covertly mounted via NFC or RFID). This is further backed up by the claim made in [67] that transport authorities are willing to store as minimum critical data at the terminal side as it is possible. The aforementioned stimulates to greatly reduce the amount of private information disclosed from the e-ticket side to a terminal during check-in/out sessions. More specifically, terminals must be prevented from *tracking* valid e-tickets, *distinguishing* between them as well as from *identifying* the customers associated with e-tickets. For the invalid ones (e.g. blacklisted) this requirement may be relaxed (see the discussion on revocation/black list checks later).

3.5.1.3 Privacy Against the Back-end

Ideally, the back-end should also be prohibited from both identifying and tracking individual transactions between terminals and e-tickets, similarly to the previous case (privacy requirement 2). It is highly desirable, however, that *fine-grained* billing is enabled at the back-end side[1] (see fine-grained billing support in Section 3.5.2). Therefore, a reasonable trade-off is allowed in this case, namely the back-end is prohibited from identifying the customers associated with e-tickets but it can correlate different front-end sessions (registered as travel records) to a single e-ticket *pseudonym* for billing purposes.

3.5.2 Fine-grained Billing Support

3.5.2.1 The demand for fine-grained billing

One of the core advantages of ESPT is the ability to offer highly flexible fare policies to customers. This makes the transport service more attractive and enhances the competitiveness of the public transport company on the market (see Section 2.1.1). In contrast to upfront and flat rate payments, the approaches involving fine-grained billing can offer a personalized fare scheme which takes into account the customer's travel pattern. The customer does not have to go through the tedious process of carefully studying complex pricing schemes any more and automatically gets a regular bill (e.g. once a month), which is optimally computed according to the service actually consumed during the billing period.

Fine-grained billing directly supports additional services and loyalty programs which are attractive for customers. For example in [68], it was stated that a considerable number of customers choose personalized cards since they provide more services. Several ESPT systems have already reacted to this trend offering a fine-grained billing approach [69, 70].

Thus, the support for fine-grained billing plays an important part in the process of fully leveraging the potential of e-ticketing systems and making the optimal use of their peculiar advantages over the conventional systems for public transport. In this dissertation, therefore,

[1] For completeness, it should be mentioned that client-side billing is also possible. This, however, has several important drawbacks that impede the acceptance of such an approach in practice, see the respective discussion in Section 3.5.2.2.

supporting this feature is considered an important requirement for ESPT. A negative side-effect of it is, however, that it can pave the way to privacy invasion from the transport companies' side. The respective discussion together with the methods for privacy protection (alongside fine-grained billing) are discussed later in this work.

3.5.2.2 Thin vs. Rich Client for Fare Calculation

Another important issue with respect to fine-grained billing is the actual method for fare calculation. With this respect, two main approaches[1] can be distinguished: *(1)* centralized fare calculation (in the back-end) and *(2)* the decentralized one (at the client side). The former implies that the price is calculated in the back-end of a system therefore allowing a "thin" client approach for users, enabling the utilization of relatively simple user devices for e-tickets, etc. The decentralized model, to the contrary, considers delegation of (a certain part of) fare calculation to the user side, which requires additional computational overhead for customers (i.e. beyond simply using an e-ticket for the transport service itself) and introduces more stringent requirements to the trustworthiness and complexity of the user-side equipment[2]. The second approach is considered beneficial for user privacy since the most sensitive pieces of information (visited locations together with time, user ID, etc.) are processed at the user side and are only partially disclosed (if at all) to the TA for correctness enforcement (e.g., for the so-called spot checks, see [71].) The substantial shortcoming of the decentralized approach, however, are the more stringent requirements for the user device and more importantly additional overhead for a user who is responsible for maintaining the processes pertaining to local fare calculation and result submission to the back-end. In the e-ticketing scenario considering the decentralized model, further efforts are likely to be required for a customer to compute travel fees locally and submit the result to the back-end by, for example, regularly interacting with a card reader connected to the home computer (with the latter running a specialized piece of software). Moreover, such model is more error prone than the centralized one, since it is much harder to trace the source of possible failures and determine their nature (e.g., system malfunction, malicious user device behavior, etc.). Lastly, the decentralized approach implies additional costs for users due to the necessity to obtain extra pieces of hardware (such as NFC/RFID reader, etc.). Therefore, while focusing on privacy protection in ESPT, we resort to the *centralized model* for fare calculation as it is much more likely to have higher acceptance both among customers (less overhead, simpler to use) and public transport companies (billing correctness).

3.5.3 Loose-coupling

E-ticketing systems for public transport are widely distributed and have to process thousands of check-in/out events in parallel. Having the back-end serve all such events originating from terminals in real time would render the overall system highly impractical and excessively costly. Therefore, in the real-world scenario, terminals do not maintain a constant real-time connection to the back-end but rather receive regular updates which do not have to be performed in real

[1] This classification was initially introduced in the context of road pricing in [71].

[2] Since a user device where an e-ticket resides is usually fairly constrained in terms of computational power, additional equipment at the user side may be required for fare calculation and subsequent result submission to the back-end (for example, a card reader connected to a personal computer running a piece of software certified by TA).

time. Our discussions with the representatives of public transport companies of Metrô São Paulo in Brazil and Dresdner Verkehrsbetriebe in Germany have shown that tightly-coupled systems are highly impracticable and are not likely to gain acceptance in practice. Therefore, privacy-preserving solutions being developed for ESPT must be compatible with a loosely-coupled architecture which is considered among the core requirements in this dissertation.

3.5.4 Efficiency

To successfully deploy a privacy-preserving system in a real world scenario, it must comply with efficiency requirements. In ESPT, it is especially critical in the system front-end where check-in/check-out events must be handled in a timely fashion to avoid long passenger queues. In practice, the maximum tolerated time frame within which a terminal must meet the decision of whether to accept or reject an e-ticket, ranges from 0.2 sec (London Oyster Card) to 2.0 sec (early versions of Singapore EZ-Link) [72]. Quite often, a side-effect of many privacy-preserving protocols is their suboptimal efficiency and demand for additional resources (see the related work discussion in Chapter 4). An architecture of a target privacy-preserving ESPT, therefore, should not by design entail prohibitive performance and enable its deployment in the real world.

3.5.5 Multilateral Security

The security goals of a TA as well as the ones of each user must be carefully addressed in each ESPT system. For the TA that would imply, for example, that customers cannot cheat and use the transport service without delivering the appropriate payment. Moreover, the effort of forging e-tickets must be high enough to render it practically infeasible to perform. For customers, it should be guaranteed that protection measures against framing or incorrect billing are implemented. The aforementioned issues are collectively referred to as multilateral security in this dissertation following the terminology discussed in [73].

Core Requirements: Summary and Discussion

The above discussed core requirements for a privacy-preserving ESPT are summarized in Table 3.4. Note that *Requirements 1* and *2* are particularly conflicting. On the one hand, fine-grained billing requires capture and analysis of user travel patterns[1]. On the other hand, this immediately poses a serious privacy threat if implemented without taking the issues of privacy protection into account. A straightforward approach to solve this requirements conflict would be to completely abandon *Requirement 2* and therefore to develop a fully anonymous system where travel patterns cannot be collected and analyzed by default. However, as already mentioned in Section 3.5.2, the support for fine-grained billing is essential in the current state of the market in public transport area. Therefore, for the target ESPT system a reasonable trade-off should be found to enable fine-grained billing at the same time ensuring that customer privacy is protected.

[1] It should be mentioned that there are some methods from the area of private information retrieval (PIR) which by design allow a privacy-preserving pattern analysis for billing purposes, see for example [16]. They are, however, relatively inefficient and inflexible

Table 3.4: A summary of core requirements

1. *Privacy*
    a) *Privacy against external observers.* An external attacker (see the adopted attacker model in Section 3.6) must be prevented from deriving any PII from interaction between e-tickets and terminals in the front-end.
    b) *Privacy against terminals.* Terminals must be prohibited from tracking and distinguishing between valid e-tickets as well as from identifying the users associated with them.
    c) *Privacy against the back-end.* The back-end is allowed to correlate travel records related to a single e-ticket while being prohibited from identifying the users associated with e-tickets.
2. *Fine-grained billing support*
3. *Loose-coupling*
4. *Efficiency.* Check-in/out events handling must comply with the timing requirements.
5. *Multilateral security*

3.6 The Attacker Model

Along with the discussion of core requirements, further assumptions about the relevant adversary types (collectively referred to as an attacker model) have to be made in order to develop a privacy-preserving solution. For the case of ESPT, the attacker model defined in Table 3.5 is considered within this dissertation.

The division into outsider/insider is made with respect to the attacker's involvement into system information flow. That is, the attacker of *type 1* is an entity exogenous to the system. It is assumed that an observing attacker is polynomial-time bounded and is not able to physically tamper with the device carrying an e-ticket.

Terminals are widely distributed within the transport network and are, therefore, situated for the most part in an unsecured area and in certain cases may be even subject to compromisation. Moreover, the wireless interface used for ticket validation can be misused by third parties representing an additional attack vector (e.g., a buffer overflow attack covertly mounted via the RFID/NFC interface). Should an adversary manage to obtain information from terminals in such a way (represented by *type 2* attacker), privacy of users as well as the overall security of the system is likely to be endangered. Furthermore, in practice, transport authorities have proved to be reluctant to have terminals process security and privacy relevant pieces of information [67]. Therefore, in this attacker model, terminals are prohibited from tracking e-tickets, distinguishing between them as well as from identifying customers associated with e-tickets.

Privacy protection against the back-end is represented by the attacker of *type 3.* Even though

Table 3.5: The adopted attacker model

1. *(Outsider)* **External observers** can tap the wireless communication between terminals and e-tickets in the front-end.

   → No derivation of PII should be possible in this case.

2. *(Insider)* **Terminals** can perform additional analysis of communication data and logs maintained, may leak information (e.g., due to a buffer overflow attack mounted via the wireless interface).

   → No tracking of valid e-tickets, distinguishing between them or identification of user identities associated with e-tickets should be allowed.

3. *(Insider)* **Back-end** can process all information pieces under its control, relate them together and analyze.

   → No identification of users associated with e-tickets should be possible.

the back-end is in the possession of large amounts of information pertaining to check-in/out events and has a global view on the system (which enables fine-grained billing for specially registered e-tickets in particular, see Section 3.5.2), it must be prohibited from learning the user identity associated with e-tickets.

3.6.1 Additional Assumptions

In order to clearly define and refine the scope of threats considered in this dissertation, the following assumptions with respect to the adopted attacker model need to be explicitly mentioned.

First, it is assumed that external observers (attacker of *type 1*) are not able to act at the physical layer by mounting, for example, a radio frequency (RF) fingerprinting attack to track user devices managing e-tickets.

Second, a relay attack during which a terminal is spoofed into communicating with the rogue device relaying messages to and from a legitimate user medium is out of scope. Relay attacks are extremely difficult to mitigate against and this is still an open research question in the area of e-ticketing and micropayment. Some known techniques such as distance bounding protocols are likely to have only a limited effect due to high volatility and fluctuations of the NFC/RFID communication interface.

Third, it is assumed that it is not possible for an attacker to gain access to auxiliary sources of information beyond the context of the described public transport system. An example of such information sources could be video surveillance data which would allow for inference of additional pieces of information pertaining to users and would imply a qualitatively new attack vector with respect to privacy invasion. Moreover, an omnipotent attacker (e.g., major governments), which is fairly likely to actually have additional access to such auxiliary sources of information, is beyond the scope of this work as well.

Furthermore, it is assumed that due to the quantized nature of stops in the context of a public transport network in a large city, multiple users are going to get on and off each station simultaneously (as opposed to walking travel patterns). Therefore, it is assumed that user travel patterns with respect to public transport do not leak enough information to be able to identify an individual "in the crowd" with sufficiently high probability. Observe, however, that unlike gathering check-in/out information from public transport (i.e., stops/station data), personal travel patterns originating from GPS or cellular data are far more privacy invasive (see [74] for instance).

Lastly, the attacks leveraging physical tampering with user device, malicious code injection, etc., are out of scope of this work. Moreover, the omnipotent attacker being able to break established encryption mechanisms or, for example, having power to combine different pieces of information from distinct non-colluding entities, is not considered either.

Chapter summary

In this chapter, an overview of privacy issues in e-ticketing systems for public transport (ESPT) was performed. Firstly, the generic privacy threats endemic to ESPT together with the respective countermeasures were discussed. Then the core technological stacks (RFID and NFC) were analyzed for the availability of built-in privacy-preserving mechanisms accessible in the off-the-shelf fashion. Summarizing, it can be stated that unlike the security mechanisms, the ones explicitly addressing privacy are either completely absent in the respective standards or only rudimentary considered. The chapter is finished by the discussion of core privacy requirements for ESPT as well as of the adopted attacker model. This influences the choice of evaluation criteria used during the detailed analysis of the related work presented in the next chapter.

4 Related Work

The issues of customer privacy protection in ESPT are rapidly gaining importance. There have been only a few papers, however, which explicitly consider this problem for the target domain. Therefore, along with the work specifically targeted at systems for public transport, the other research results pertinent[1] to ESPT are considered within this chapter. The material is reviewed and evaluated according to the set of criteria presented in Section 4.1. The solutions falling into the category of tightly-coupled systems are discussed in Section 4.2. The approaches based on the decoupled architecture are analyzed in Section 4.3. The summary of the most important solutions reviewed is presented in Section 4.4.

4.1 Evaluation Criteria

The core evaluation criteria have been essentially derived from the core system requirements (see Section 3.5) and the attacker model (see Section 3.6). They are further extended with other relevant criteria. The result is summarized in Section 4.1.1. The respective discussion follows in Section 4.1.2.

4.1.1 Evaluation Criteria Summarized

The complete criteria set used for evaluation of the related work is listed below. The most decisive criteria are marked in bold.

1. **Back-end coupling:**
    a) Tightly-coupled;
    b) Semi-coupled;
    c) Loosely-coupled.
2. **E-ticket anonymity against terminals;**
3. **E-ticket untraceability against terminals;**
4. **Mutual authentication between terminals and e-ticket;**
5. **Fine-grained billing support;**
6. **Trust assumptions:**
    a) Back-end;
    b) Terminals.
7. Explicit consideration of public transport area;

[1] For example, lightweight protocols for mutual authentication in RFID environment.

8. Possibility of e-ticket revocation;
9. Dynamic extensibility allowing to accommodate new e-tickets without re-initializing the system-wide parameters.
10. Tamper resistance requirement for the e-ticket carrier medium;
11. Involvement of an external personal device to provide privacy properties;
12. Cryptographic primitives used;
13. Type of fare calculation:
    a) Centralized (in the back-end);
    b) Client-side;

4.1.2 Evaluation Criteria: Discussion

Criteria subset 1 (Back-end coupling). Among all the criteria listed in previous section, the one indicating the type of connection maintained between the back-end and terminals (*Criteria subset 1*) is decisive for the current evaluation. Three main system types can be distinguished in this context, namely *(a)* tightly-coupled, *(b)* semi-coupled, and *(c)* loosely-coupled. The last two ones, namely types *b* and *c*, are collectively referred to as decoupled systems in this dissertation. Below each of them is discussed in more detail.

(a) Tightly-coupled Systems. The first case (*Criterion 1a*) represents the "always on-line" scenario where the back-end maintains constant connection to the terminals and actively participates in real-time e-ticket validation protocols taking place in system front-end. Such systems would be beneficial for a number of reasons. Firstly, this would substantially limit the set of possible attack vectors both in terms of e-ticket privacy and security (for TA and for the customer, i.e. in a multilateral sense) since in such a case terminals could simply be used as a plain relay component without further involvement of any more intelligent tasks. Terminals, therefore, would be solely responsible for relaying messages between the e-ticket attempting to be validated and the back-end. Secondly, this greatly simplifies infrastructure management and overall system architecture. The decisive drawback of such an approach, however, lies in the stringent requirement to network connectivity (small delays, etc., leading to dramatic increase in infrastructure costs) and the back-end component being a single point of failure (system reliability and scalability). Taking into account the number of e-tickets circulating in the system[1] and terminals in operation (simultaneous validation requests) our suggestion would be *to decouple* the back-end functionality from e-ticket validation processes and to resort to one of the two other architecture types discussed below. Further discussion on tightly-coupled systems and their properties is provided in Section 4.2.1.

(b) Semi-coupled Systems. Semi-coupled systems (*Criterion 1b*) rely on the semi-online connection to the back-end for regular updates of black lists, certificate information, and other parameters required to keep the system operating. These updates are usually carried out on a nightly basis. Terminals, however, do not have to consult the back-end during the time-critical e-ticket validation and are able to serve check-in/check-out requests locally.

[1] In the currently largest ESPT in the world – Octopus Card Limited in Hong Kong – there are around 11 million cards in circulation which are used in over 8.3 million daily transactions [75].

(c) Loosely-coupled. Loosely-coupled systems (*Criterion 1c*) are based on the infrastructure where terminals are able to validate e-tickets without the need to get regular updates from the back-end (such as black lists updates, etc.). In this case, however, a certain portion of private information would have to be stored on the terminal side in order to provide for proactive validation of e-tickets and enforcement of a travel policy of a TA. Moreover, such systems are inherently inflexible and do not provide support for efficient accommodation of new e-tickets (expressed as *Criterion 9*).

Further discussion on advantages and drawbacks of decoupled systems (i.e., incorporating both semi-coupled and loosely-coupled ones) is provided in Section 4.3.1.

Criteria 2, 3 (Anonymity/Untraceability against terminals). It is of high importance to ensure that e-tickets are anonymous and untraceable against terminals following the attacker model discussed in Section 3.6. *Criteria 2, 3* indicate if these properties are met.

Criterion 4 (Mutual authentication). The communication partners should be properly authenticated to each other prior to the session begin. This provides protection against man-in-the-middle attacks and protects e-tickets against unauthorized interaction. Mutual authentication (*Criterion 4*) is especially critical in the constrained front-end environment due to contactless communication and much weaker capabilities to secure terminals and especially carrier mediums for e-tickets. The back-end communication is, in contrast, considered relatively secure due to the plethora of well established fully fledged security mechanisms which are available.

Criterion 5 (Fine-grained billing support). Another important criterion for evaluation is the support for fine-grained billing (*Criterion 5*). Whereas this might be viewed as an optional feature for ESPT, there is a strong argument presented in Section 3.5.2 which underlines the importance of the privacy-preserving fine-grained billing support in such systems and outlines its decent potential for the future.

Criteria subset 6 (Trust assumptions). Following the attacker model presented in Section 3.6, the reviewed literature has been analyzed according to the trust assumptions with respect to the terminals and to the back-end. Namely, in several solutions it is assumed, for example, that terminals are trusted to store identifying information about the circulating e-tickets and consequently to fully identify them on each interaction (check-in/out). Furthermore, a substantial body of literature is based on the assumption that the back-end is fully trusted meaning it can identify and trace individual e-tickets, create travel patterns and link them to each user. There are approaches, however, that do not require terminals and/or back-end to be trusted with respect to processing of customer private information. These trust assumptions are of high importance for the evaluation of related work in this thesis. They are reflected in *Criteria subset 6*.

Criterion 7 (Explicit consideration of ESPT). Only a few pieces of the related work reviewed explicitly consider the area of public transport and therefore take into account the specifics of the underlying system model and the respective context. *Criterion 7* reflects this issue.

Criterion 8 (Revocation). For TA, it is important to be able to (efficiently) revoke the issued e-tickets. In a non-privacy-preserving setting, it can be done in a fairly straightforward way. For several privacy-preserving solutions, however, it may impose additional challenges, since their primary goal is to protect customer privacy and often the revocation issue is treated as an optional feature if at all. *Criterion 8* captures the fact if revocation is addressed in the reviewed solution.

Criterion 9 (Dynamic extensibility). Several privacy-preserving solutions for ESPT require complete re-initialization of the system parameters in order to accommodate new e-tickets. This may render them highly impractical in the real life scenario. This issue reflected in *Criterion 9.*

Criterion 10 (Tamper resistance). E-tickets issued to customers can be viewed as a part of the TA's security domain similarly to the smart cards issued for banking purposes. The latter possess a tamper resistant module to securely store critical information such as authentication and signing keys, etc. Until recently, the costs associated with such smart cards were considered to be nearly prohibitive to be used in public transport. However, along with the rapid evolution of smart card technology and the resultant price fall, a growing number of transport operators are adopting the cards with a tamper resistant module for annual and monthly passes. Therefore, there are some privacy-preserving solutions that require a certain level of tamper resistance for e-tickets which is reflected in *Criterion 10.* It should be mentioned that a substantial part of the early work on privacy-respecting authentication protocols in RFID area consider the issue of tag compromisation. RFID devices were initially thought to be inherently susceptible to compromisation attacks, i.e. when an attacker can obtain *all* the tag's secrets by physically intervening with it. Therefore, quite a few approaches based on symmetric cryptography and hash functions additionally consider the notion of backward traceability which suggests that if a tag gets compromised at a certain point in time, an adversary must be prevented from tracking it in the past (e.g., based on the previously recorded sessions between other tags (including the victim) and terminals). Moreover, as it is further discussed in Section 4.2.1, some solutions aiming at optimization of tag identification procedure have a specific privacy weakness called the "tag compromise vulnerability". It is caused by the fact that the compromisation of a certain tag may endanger the privacy of the other uncompromised ones (especially their tracking may become possible). In this dissertation, we assume that relatively profound RFID tags (let alone NFC-enabled smartphones) can provide for a decent protection of their secrets (i.e. they possess a tamper-resistant memory area). Therefore, in our further analysis the issue of tag compromisation is not going to be explicitly considered as one of the main criteria.

Criterion 11 (Involvement of an external device). RFID devices have been historically considered to be inherently resource constrained with very limited computational capabilities let alone the ones providing security. Since one of the types of carrier medium for ESPT are RFID-based devices, the area of privacy-respecting protocols in RFID environment is reviewed along with other solutions. Due to severe computational constraints, several analyzed solutions for this area utilize an external (personal) device (e.g., a smartphone) to which computationally demanding tasks such as cryptographic computations are outsourced. *Criterion 11* captures this issue.

Criteria subset 12 (Cryptographic primitive used). Various privacy-preserving protocols have different assumptions with respect to the capabilities of front-end devices and therefore utilize different cryptographic primitives to achieve certain privacy properties. In the e-ticketing domain, each type of carrier medium (smart card, NFC smartphone) has its own cryptographic capabilities and therefore imposes respective constraints on the underlying cryptographic primitives that can be used. Therefore, *Criteria subset 12* reflects the type of cryptographic primitives on which each of the reviewed approaches is based. Evolution in the area of RFID and smart card manufacturing, not to mention NFC-enabled smartphones, has made it possible to efficiently support relatively advanced cryptographic operations like public-key cryptography on constrained devices. Therefore, this criteria subset is not considered to be critical for the current evaluation.

Criteria subset 13 (Fare calculation). As it was already mentioned in Section 3.5.2, in the e-ticketing scenario two main approaches can be distinguished with respect to the user involvement into the process of fare calculation: *(1)* centralized (performed in the back-end) and *(2)* decentralized (client-side fare calculation). Due to the reasons discussed in Section 3.5.2, this dissertation focuses on the first approach. However, the decentralized billing is concisely discussed in Section 4.3.4 for completeness.

4.1.3 Evaluation Criteria: Summary

The evaluation criteria which are subsequently used during the review of privacy-preserving solutions for ESPT (as well as for the other neighbouring domains) have been presented and discussed in this section. The following analysis can be divided into two main categories according to the core *Criteria subset 1* (back-end coupling), namely *(1)* the solutions based on tightly-coupled systems and *(2)* the ones considering the decoupled architecture (semi- and loosely-coupled systems), see Figure 4.1.

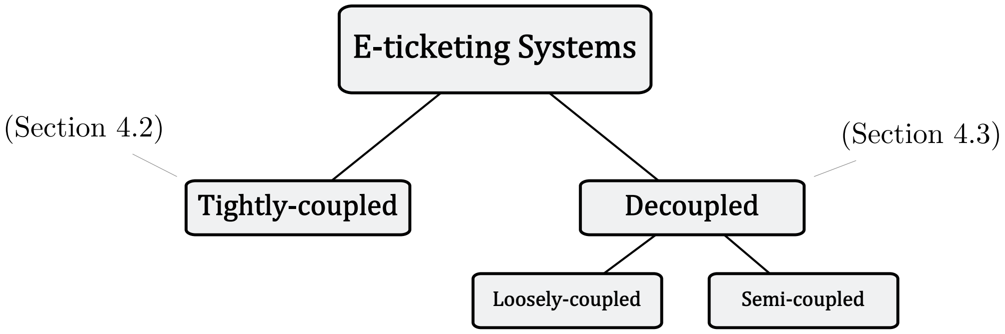


Figure 4.1: Taxonomy of the reviewed solutions: an outline

In the next section, the related work is analyzed according to the evaluation criteria discussed above. The reviewed material is structured according to the two main categories presented in Figure 4.1. It should be mentioned that in the following analysis, the RFID devices are often referred to as tags due to the initial terminology considered in the papers reviewed. In this dissertation, RFID tags (or simply tags) are viewed as potential carrier media for e-tickets (together with NFC-enabled smartphones). If the reviewed approaches explicitly consider public transport, we refer to e-tickets. Otherwise, the term ”tag” is used as in the corresponding paper being reviewed.

4.2 Related Work Analysis: Tightly-coupled Systems

4.2.1 An Introductory Discussion

A common property of the solutions pertaining to this category is the existence of a centralized database of tag IDs which is maintained by the back-end. In order to identify/authenticate a tag, the respective request is forwarded to the back-end by a terminal. In order to provide for privacy properties, a pointer to the tag’s ID (or other authentication parameter of a tag) delivered to the terminal is concealed in some way usually involving cryptographic operations. Therefore, in order to authenticate a tag, the central database quite often has to perform an exhaustive search to find a match. The reason for this is that authentication usually requires applying corresponding cryptographic operations by the back-end *sequentially* to every element in the back-end database. After authentication is performed in the back-end, the answer (accept/reject) is communicated back to the terminal. This approach has several advantages:

1. *Terminal simplicity.* All intensive computations are performed in the back-end.
2. *Less trust in terminals.* The terminals do not need to maintain any secret/private information pertaining to tags and are essentially relaying messages between tags and the back-end.
3. *Simple infrastructure.* The overall system infrastructure is simple and centralized. There is no need to consider revocation in the front-end (e.g., using terminal-side black lists) since everything is done in the back-end.
4. *Easiness of e-ticket revocation.* Since on each check-in/out the back-end is consulted, e-ticket revocation can be straightforwardly implemented in the centralized way without the necessity to distribute the respective information (e.g., revocation lists) to the terminals.

However, the crucial disadvantage of this centralized fully online approach is that it does not scale well. If n is the number of all tickets circulating in the system, the authentication complexity in the front-end is often linear in n, i.e. $O(n)$. In a real world scenario, n can be in the order of millions (e.g. roughly 8 million in Hong-Kong Octopus system [75]). Additionally, the search procedure involves some sort of cryptographic operations (e.g., applying a cryptographic hash function for each database entry, possibly several times to recover a hash chain). What may be even more critical, the back-end must always (24/7) maintain a reliable connection to all terminals in the system which can be quite expensive from the system architecture point of view, especially in case of mobile on-site terminals (e.g., on buses). A further issue is that in order to provide untraceability against terminals, many approaches falling into this category (e.g., [76, 77]) consider updating a tag pseudonym on each successful authentication which requires the back-end database and a tag to be in the synchronized state. Otherwise, a valid ticket can be rejected due to inconsistency (for example, as a consequence of an attacker having queried a tag between two successful authentication sessions). Moreover, in such scenario, the back-end can become a bottleneck and represent a single point of failure (SPoF) impairing the system reliability. The aforementioned disadvantages are listed below for clarity.

1. *Scaling issues.* The back-end often must perform exhaustive search (rendering $O(n)$ complexity) under stringent timing requirements.

2. *The back-end is online 24/7.* The back-end must maintain a constant connection to all terminals in a system (can be rather expensive, especially in case of mobile on-site terminals).

3. *Synchronization (Statefullness, DoS).* Many privacy-preserving approaches require tag pseudonyms to be updated on each successful authentication which requires that synchronization is reliably maintained between the back-end and tags. In case of desynchronization (e.g. due to an attack), a valid tag can be rejected.

4. *The back-end is a bottleneck and a SPoF.* In a simple case of fully centralized infrastructure, the back-end can become a single point of failure and a bottleneck.

Regarding point 1 (scaling issues), additional mechanisms can be considered to enhance search performance. One of the most prominent approaches to optimize the average tag identification time are:

1. *Precomputation-based.* Utilizing some sort of precomputation which requires a considerable expansion of the size of the back-end database (i.e. trading off storage for real-time identification efficiency).

2. *Secrets ordering.* Organizing the secrets pertaining to tags in some ordered way[1], for example, in a tree structure (where leaves correspond to tags' IDs) and having a reader engage in multiple request-response rounds with a tag to traverse the tree to perform authentication. An important (negative) property of the protocols falling into this category is that parts of tags' secrets are *shared* in some way (tree traversal or matrix search) to optimize the average search time.

As an example of the first class of optimization approaches, a special time-memory trade-off presented in [79] can be considered. It is targeted at enhancing the performance of the hash chain-based OSK protocol [80] (see further). The main idea is to use precomputed tables (units of memory, hence the name) to boost the search operation. The performance gain increases with the square of available memory (used for precomputed tables) and depends on system parameters and some further assumptions peculiar to each particular case (see [79] for details). The optimized identification complexity achieved is $O(n^{\frac{2}{3}})$ as opposed to $O(n)$ of the exhaustive search (n is the number of tags in a system). Another concept to enhance the tag identification time was developed by Nohara *et al.* [81] who suggested to use the precomputation scheme based on bloom filters [82] to store the hash chains for each tag (one bloom filter for each tag in a system). Alomair *et al.* [83] provided their own efficient solution for

[1] The prominent example is a tree structure. However, organizing the tag secrets in some other form, e.g. in a matrix as was done in [78], is possible as well.

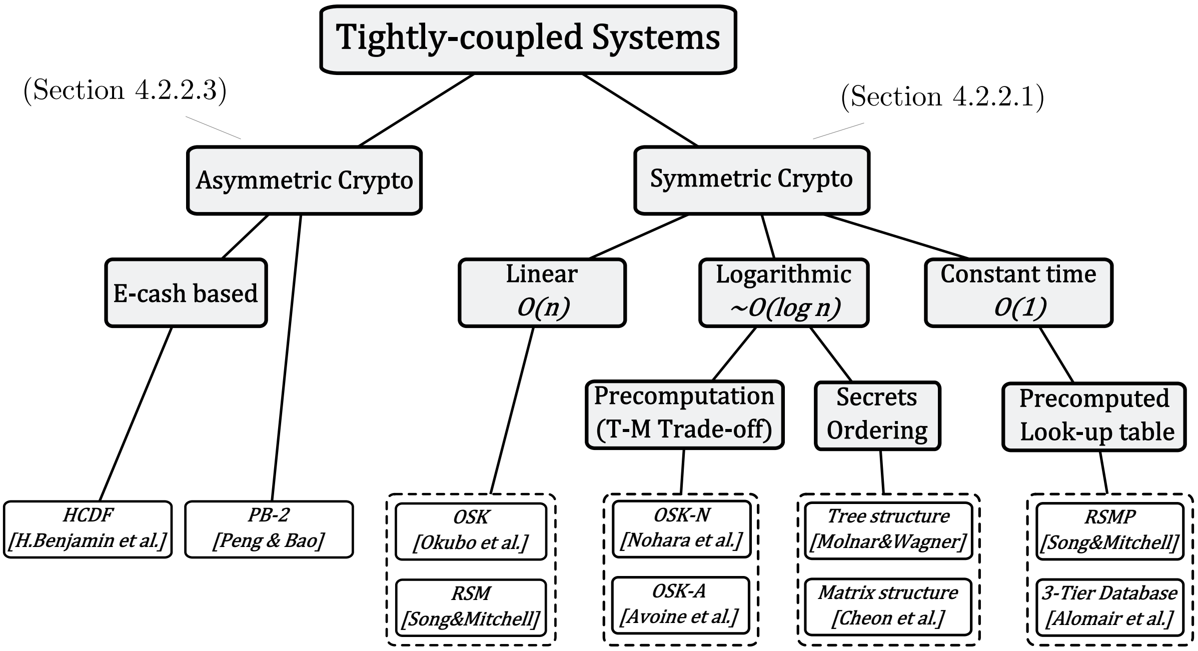


Figure 4.2: Tightly-coupled systems. Solutions taxonomy.

privacy-preserving tag identification which is not based on OSK (see the taxonomy presented in Figure 4.2). Similarly to the optimization approaches discussed above, a time-memory trade-off is used for precomputations. The precomputed back-end database consists of 3 tiers with the tag ID information residing in the third one (reminding a linked list structure). On each successful authentication session with a tag, its pseudonym gets updated from a pool of random pseudonyms maintained by the back-end and securely transferred to a tag (for details, see [83]). The protocol provides mutual authentication between terminals and tags with the constant time identification complexity. However, the addition of new tags to the system requires recomputing the whole database which is a costly operation. Moreover, an attacker model assumed in this paper is rather specific.

Among the approaches falling into the second category (ordering tag secrets in a structure), Molner and Wagner's optimization based on tree-based identification can be considered [84] (see Figure 4.2). It reduces the identification complexity from $O(n)$ to $O(\log n)$. Their method, however, is potentially susceptible to probabilistic tracing attacks in case several tags get compromised[1] as demonstrated in [85]. Another method to enhance the search performance was developed by Cheon *et al.* [78] and considers organizing tag secrets in a matrix form (the so-called "meet-in-the-middle" strategy). The search complexity in this case is reduced to $O\left(\sqrt{n}\log n\right)$. This approach suffers from a similar problem as the Molner and Wanger's one in case of tag compromisation (see "indirectly compromised tags" in [86]).

[1] This privacy threat is called "tag compromise vulnerability" in [83]. The reason for this is that a compromised tag reveals its keys from the root to the corresponding leaf of the tree. The keys in each path are *shared* with other tags which poses a privacy threat, since the more tags get compromised, the higher is the likelihood that other non-compromised tags can be traced and even identified. See [85] for details.

Therefore, the solutions aiming at enhancing the search performance require careful consideration of the arising trade-offs. Moreover, further disadvantages of having the back-end manage the whole tag authentication process (see points 2–4 above) render this type of systems hardly practicable for privacy preservation in real-life ESPT.

If it is required that terminals can not trace and identify tags, the only viable solution which remains is to have the back-end handle the authentication procedure while terminals simply relay messages between the tag being authenticated and the back-end. Therefore, resorting to a tightly-coupled system seems to be unavoidable if no asymmetric cryptography is used. However, in large systems the scaling issues are very likely to impede the overall performance and render such ESPT highly impractical. For this reason, a certain kind of decentralization is required leading to the fact that the pieces of information pertaining to tags's secrets are stored either at local centers, which are able to cope with the required load, or even at the terminal side. In the former case, the system costs are likely to grow up dramatically. In the latter one, terminals must ensure the proper protection of sensitive information or otherwise the privacy concerns arise.

The solutions shortly mentioned above are based on symmetric cryptography and do not leverage the potential of asymmetric cryptography to enhance its privacy properties. The main reason for this is the severely constrained environment of low-cost RFID tags which does not allow computationally expensive asymmetric encryption to be efficiently implemented on a simple tag. However, the newer generations of middle class RFID devices are already capable of efficiently performing asymmetric cryptographic operations [87, 88]. For example, in one of the solutions based on e-cash [8], it is already assumed that an e-ticket is able to perform public key encryption.

In what follows, the most relevant approaches from the ones mentioned above are going to be reviewed in more detail. They are presented in a structured way in Figure 4.2 for clarity.

4.2.2 Tightly-coupled Systems: A Detailed Review

4.2.2.1 Solutions Based on Symmetric Cryptography

Ohkubo et al. (OSK) [80]

Protocol Description. The protocol due to Ohkubo, Suzuki and Kinoshita (OSK) [80] was developed for privacy-respecting RFID identification and therefore it is particularly suited for lightweight RFID tags. OSK only requires two hash operations on the tag side for a single protocol run and possesses the following privacy properties. Firstly, tag's answers to each query from a terminal are indistinguishable[1] from random values thus providing untraceability against an observing attacker (see the attacker model in Section 3.6). Secondly, the transactions are unlinkable for the observing attacker. Lastly, should the adversary be able to acquire the tag's secret, it would be still impossible to trace the previous transactions of this compromised tag (in [80] the latter property was referred to as forward security).

The protocol is relatively simple and essentially requires a tag to be capable of performing two hash operations using two distinct hash functions:

[1] Provided that the underlying hash function has the respective properties.

- $H(\cdot)$ is used to internally update the tag's secret on each query from the terminal side;
- $G(\cdot)$ is used to further randomize the tag's answers and make them untraceable for observing attackers.

After the initialization phase, each tag j shares its initial secret[1] s^j_{init} with the back-end. The latter stores tags' IDs with their corresponding initial secrets in the database. On each query, the tag hashes the value of its current secret state s^j_i using G and sends the result $G(s^j_i)$ as a response to the terminal. Afterwards, the tag uses the other hash function H to internally update its secret by hashing the current secret value: $s^j_{i+1} = H(s^j_i)$. The terminal in turn forwards the tag's answer $G(s^j_i)$ to the back-end where an exhaustive search is performed for all entries in the database (i.e. for all registered tags) in order to reconstruct the hash chain and find a match. More specifically, for each DB entry l, the back-end compares the current tag's answer $a^j_i = G(s^j_i)$ to the value computed at the back-end side according to the function $F(l,k)$:

$$\forall l \in T, k = [1,n] : F(l,k) = G\left(H^{(k-1)}\left(s^l_{init}\right)\right),$$

where s^l_{init} denotes the initial secret of the l-th tag, n is the maximum length of the hash chain, T represents the set of all tags circulating in the system.

Each tag in the system, therefore, can be identified and authenticated at most n times, which is determined by the maximum length of the hash chain. The main steps of OSK protocol are depicted in Figure 4.3.

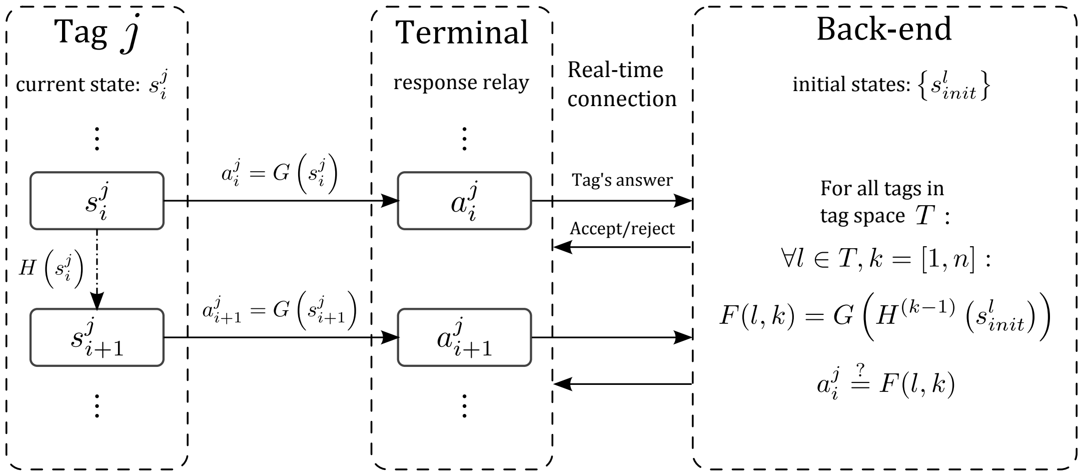

Tag j
current state: s^j_i
s^j_i
$H\left(s^j_i\right)$
s^j_{i+1}
$a^j_i = G\left(s^j_i\right)$
$a^j_{i+1} = G\left(s^j_{i+1}\right)$
Terminal
response relay
a^j_i
a^j_{i+1}
Real-time connection
Tag's answer
Accept/reject
Back-end
initial states: $\{s^l_{init}\}$
For all tags in tag space T :
$\forall l \in T, k = [1,n]$:
$F(l,k) = G\left(H^{(k-1)}\left(s^l_{init}\right)\right)$
$a^j_i \overset{?}{=} F(l,k)$

Figure 4.3: OSK Protocol. Tag j is queried by a terminal and validated in BE. j in superscript denotes the relation to tag j. The expression $H^{(k-1)}$ denotes hash operation applied $(k-1)$ times.

[1] j in superscript denotes the tag to which the current secret s_{init} pertains.

Protocol Assessment. The protocol is fairly simple and has the generic properties of tightly-coupled protocols presented in Section 4.2.1. Among the positive features are tags' anonymity and untraceability against terminals, easiness of revocation procedure. The protocol does not require that terminals are entrusted to maintain any secret information pertaining to tags. Even though fine-grained billing was not directly considered in [80] (the protocol was not initially targeted at ESPT), it would be fairly straightforward to implement this feature, since the back-end keeps track of every transaction with each tag. In its initial version, OSK is stateless with respect to updates of tags' initial secrets s_{init} (therefore, a hash chain must be restored each time to find a match). Whereas it is a positive feature as long as DoS attacks aimed at desynchronizing the respective states of tag secrets between the back-end and each tag are considered, it imposes a hard limit on the maximum number of times a tag can be validated. Moreover, the complexity of a single protocol run is linear in the number of tags n circulating in the system and the parameter k determining the maximum size of a hash chain, i.e. $O(nk)$. No mutual authentication is performed, therefore any tag can be queried by any external or internal entity in the system. Moreover, according to the protocol, the back-end must be fully trusted. Therefore, OSK provides protection only against the first two attacker types of the attacker model discussed in Section 3.6 (outsider, terminals). Another drawback of the OSK protocol is its susceptibility to replay attacks. Indeed, any adversary intercepting the communication between a reader and a tag can impersonate the latter by simply re-sending the tapped tag's response a_i^j on request of a reader, see Figure 4.3.

The most important properties of OSK protocol are summarized in Table 4.1 below. A holistic summary along with the other reviewed approaches is presented in Table 4.19 at the end of this section.

Table 4.1: Main properties of OSK [80]. The features advantageous to ESPT are marked with a ✓sign. k represents the maximum size of the hash chain. n denotes the general number of tags circulating in a system.

Criterion	Assessment	
tight coupling between terminals and the back-end	yes	
anonymity and untraceability against terminals	yes	✓
mutual authentication in the front-end	no	
fine-grained billing is feasible	yes (as add-on)	✓
terminals are trusted	no	✓
the back-end is trusted	yes	
limited number of validations (at most k times)	yes	
serious scalability issues: $O(nk)$	yes	
susceptibility to replay attacks	yes	

OSK-A: The OSK Protocol Improved by Avoine et al. [5]

Gildas Avoine in his PhD thesis [5] suggested three improvements for the conventional version of the OSK protocol, namely *(1)* resistance to replay attacks, *(2)* mutual authentication, and *(3)* optimization of search efficiency.

The first improvement (against replay attacks) can be achieved by additionally generating a nonce[1] r at the back-end side on each request to the tag. The tag's answer a_i^j then must include the generated nonce value: $a_i^j = G(s_i^j \oplus r_i^j)$, see Figure 4.4.

In order to provide mutual authentication, the reader can authenticate itself to the tag at the end of the session by additionally sending the hashed value of the next secret state of the tag XORed with some publicly known non-zero binary string[2] $w : G(s_{i+1}^j \oplus w)$. If the answer received from the terminal is correct, the back-end has successfully authenticated itself to the tag and by this mutual authentication has been achieved (tag authentication takes place earlier in the protocol). In Figure 4.4, an improved OSK protocol is depicted.

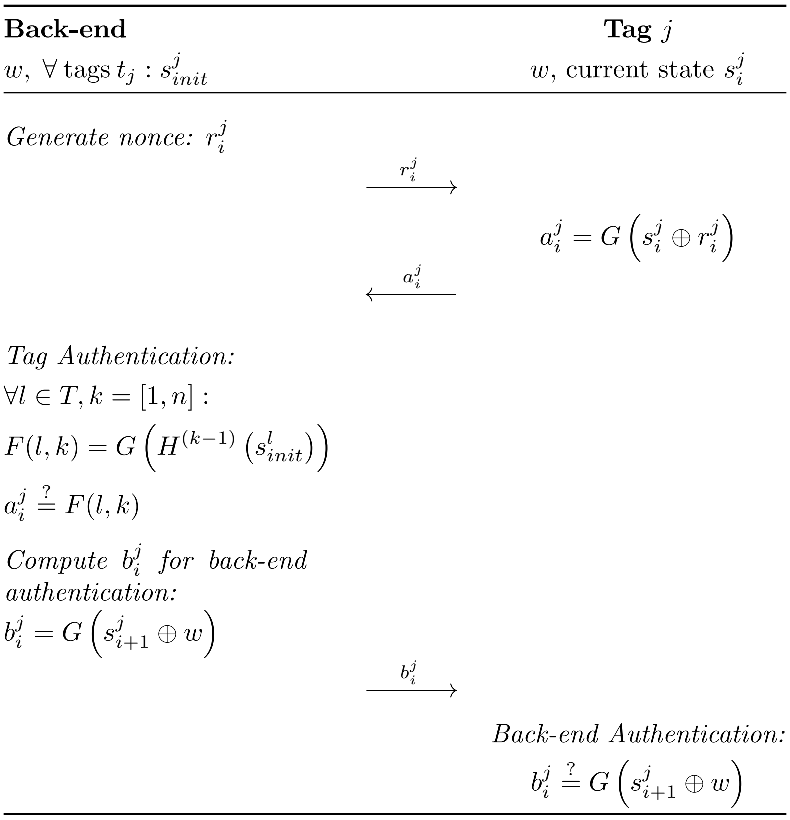

Back-end
Tag j
$w, \; \forall \, \text{tags} \, t_j : s_{init}^j$
w, current state s_i^j
Generate nonce: r_i^j
r_i^j
$a_i^j = G\left(s_i^j \oplus r_i^j\right)$
a_i^j
Tag Authentication:
$\forall l \in T, k = [1, n] :$
$F(l,k) = G\left(H^{(k-1)}\left(s_{init}^l\right)\right)$
$a_i^j \stackrel{?}{=} F(l,k)$
Compute b_i^j *for back-end authentication:*
$b_i^j = G\left(s_{i+1}^j \oplus w\right)$
b_i^j
Back-end Authentication:
$b_i^j \stackrel{?}{=} G\left(s_{i+1}^j \oplus w\right)$

Figure 4.4: OSK-A: an improved version of the OSK protocol by Avoine [5]. The i-th session with tag j is depicted.

In order to optimize the search efficiency (the third improvement for OSK), Avoine suggested that a time-memory (T-M) trade-off initially proposed by Hellman in [89] and further optimized by Oechslin in [90] is used. In order to apply this technique to the improved protocol OSK-A depicted in Figure 4.4, a tag has to additionally answer with $G(s_i^j)$ (according to the conventional version of OSK) along with $G(s_i^j \oplus r_i^j)$. The complexity of the commensurately optimized protocol depends on the memory size available for trade-off purposes. The former,

[1] "Nonce" stays for a number used once.

[2] Similarly to the common reference string model.

therefore, can theoretically be varied from $O(1)$ to $O(n)$ depending on the available memory size used for the time-memory trade-off (n represents the number of tags circulating in a system). For example, according to [79], the amount of work required to identify a tag can be reduced from N to $N^{2/3}$ using $N^{2/3}$ units of memory, respectively. This implies the search complexity of $O(n^{2/3})$ with the respective memory size complexity of $O(n^{2/3})$.

Protocol Assessment The version of OSK improved by Avoine in [5] and referred to as OSK-A in this dissertation has similar properties as OSK (see Table 4.1) with the exception of three enhancements: resistance to replay attacks, mutual authentication support, and optimization of search efficiency. Concerning the latter property, the efficiency gain is achieved using precomputed tables according to the time-memory trade-off. The downside of this is that additional memory is required for this at the back-end side and more importantly the respective recomputation must be made whenever a new tag is registered in the system.

The properties of OSK-A are summarized in Table 4.2.

Table 4.2: Main properties of OSK-A [5]. The features advantageous to ESPT are marked with a ✓sign. k represents the maxim size of a hash chain. n denotes the general number of e-tickets circulating in a system.

Criterion	**Assessment**	
tight coupling between terminals and the back-end	yes	
anonymity and untraceability against terminals	yes	✓
mutual authentication in the front-end	yes	✓
fine-grained billing is feasible	yes (as add-on)	✓
terminals are trusted	no	✓
the back-end is trusted	yes	
limited number of validations (at most k times)	yes	
complexity: tuneable from $O(1)$ to $O(n)$, T-M trade-off	yes	✓
susceptibility to replay attacks	no	✓

OSK-N: A Version of OSK Improved by Nohara et al. [81]

Similarly to OSK-A, Nohara *et al.* [81] proposed another form of time-memory trade-off based on bloom filters [82] in order to improve search efficiency of the initial OSK protocol. Similarly to OSK-A, OSK-N is not susceptible to replay attacks due to the fact that the corresponding tag's secret at the back-end side gets updated on each successful authentication,. However, in contrast to OSK-A, OSK-N does not consider mutual authentication and resistance to impersonation attacks.

The authors of [81] suggest that the issue of search complexity can be solved using bloom filters for storing the hash chains for each tag. On receiving the j-th tag's answer a_i^j (see Figure 4.3 of the initial protocol), the back-end performs the following operations. Firstly, it queries all bloom filters to find a match with the received answer a_i^j. For each match[1], the

[1] The classic bloom filter has false positives (but no false negatives).

back-end checks its accuracy by ”sequentially” computing the hash chain. After this, it can be decided if each occurrence has been a correctly identified tag or it has been a false positive and the search has to be continued in the set of bloom filters that signalize a match. After correct identification, the respective bloom filter is updated so that the state of the corresponding tag's secret is correctly maintained. The properties of OSK-N are summarized in Table 4.3.

Table 4.3: Main properties of OSK-N [81]. The features advantageous to ESPT are marked with a ✓sign. k represents the maxim size of the hash chain. n denotes the general number of e-tickets circulating in a system.

Criterion	**Assessment**	
tight coupling between terminals and the back-end	yes	
anonymity and untraceability against terminals	yes	✓
mutual authentication in the front-end	no	
fine-grained billing is feasible	yes (as add-on)	✓
terminals are trusted	no	✓
the back-end is trusted	yes	
limited number of validations (at most k times)	yes	
complexity: sublinear, depends on the setting	yes	✓
susceptibility to replay attacks	no	✓

Revised Song and Mitchell's Protocol (RSM) [91]

Along with OSK and its improvements, another important protocol developed by Song and Mitchell is reviewed in this dissertation. In their initial paper [92], the authors proposed a protocol for privacy-preserving mutual authentication between the back-end and an RFID tag. In [91], Song and Mitchell further improved and extend their solution responding to the security attacks revealed in [93]. Therefore, in this dissertation the revised version of the initial protocol is reviewed.

In contrast to classic OSK, the revised Song and Mitchell's protocol (RSM) is stateful with respect to the maintenance of secrets shared between tags and back-end and synchronization of their state. During the initialization phase, the back-end assigns each tag T a secret s and computes a tag-side pseudonym $t = h(s)$ where h is a hash function (all notations are summarized in Table 4.4). t is then uploaded to the corresponding tag T while s stays at the back-end side. The back-end in turn maintains a quadruple $(s, t, \hat{s}, \hat{t})$ for each tag, where $\hat{s}$ and $\hat{t}$ are the most recent previous values of the current s and t pertaining to the respective tag T. RSM protocol requires that RFID tags are capable of performing the following operations: XOR $\oplus$, right and left circular shift operations $\gg, \ll$. The main protocol steps are depicted in Figure 4.5 where $\leftarrow$ denotes substitution, $\|$ concatenation, and $f_t(\cdot)$ is a keyed hash function.

As it can be seen from Figure 4.5, mutual authentication between the back-end and a tag is achieved using the shared secret t and the corresponding value of s (the latter is persistently stored in the the back-end), see *Tag authentication* and *Back-end authentication* in Figure 4.5. On each successful authentication, the state of the commensurate secret is updated at both

Table 4.4: Notations used during the discussion of RSM and RSMP.

Notation	**Meaning**
T	tag identifier;
s	tag secret shared with the back-end;
$h(\cdot)$	cryptographic hash function;
t	tag-side pseudonym: $t = h(s)$;
$\hat{s}, \hat{t}$	the most recent previous values of s and t respectively;
$\gg, \ll$	right and left circular shift operations;
$\oplus$	XOR operation;
$\leftarrow$	substitution;
$\|\|$	concatenation;
$f_t(\cdot)$	keyed hash function;
$\{x_i\}$	one-time tag-specific pseudonyms;
$e_t(\cdot)$	keyed hash function (to create a pseudonyms chain);
k	the maximum length of a pseudonym hash chain.

sides, which is denoted as *Update state* in Figure 4.5. The complexity of identifying and authenticating a tag is linear in the overall number of tags in the system: $O(n)$.

In order to improve scalability and reduce complexity to the constant time $O(1)$, the authors suggest that one-time tag-specific pseudonyms $\{x_i\}$ are additionally used for tag identification. These pseudonyms form a tag-specific hash chain which is created using another keyed hash function $e_t(\cdot)$ with a tag-specific shared secret t as a key. At the tag side, the initial pseudonym x_1 is stored which is updated on each authentication request from the reader: $x_{i+1} = e_t(x_i)$. Therefore, each authentication session consists of two parts: *(1)* Tag identification on the basis of the current one-time identifier x_i (used for table look up, ensures the $O(1)$ search complexity) and *(2)* Namely authentication for which a check is performed if the shared secret t corresponding to x_i satisfies the equality $f_t(r||x_i) \stackrel{?}{=} M_T$, where $M_T = f_t(s)$ was previously computed by tag, and r is a request nonce previously sent to the tag by a reader. This version of RSM protocol is referred to as **RSM with one-time pseudonyms (RSMP)** within this dissertation. The pseudonym pool is formed by the hash chain. The length k of this chain determines the number of pseudonyms assigned to each tag for identification. Therefore, the maximum number of tag identification attempts (and subsequent authentication sessions) is limited by k. When the pool of one-time tag-specific pseudonyms is exhausted, the update procedure must be performed to update the tag-side initial pseudonym x_1 and to recompute the look-up table in the back-end. RSMP additionally requires each tag to maintain a decreasing counter c the initial value of each is set to k corresponding to the length of a hash chain (and consequently to the size of the pseudonym pool). The counter is decreased on each request from a reader and serves as an indicator when the respective pseudonyms must be updated. If for some reason the update of the shared secret fails (which should not normally happen), RSMP behaves as RSM. Moreover, the counter is used for delegation of tag authentication rights to external parties (see [91] for details). Interestingly, unlike the initial RSM protocol, RSMP does not consider back-end authentication at the tag side (compare to *Back-end authentication*

Back-end		**Tag** T
$\forall T : (s, t, \hat{s}, \hat{t})$		$t : h(s) = t$
Generate random $r_1, \lvert r_1 \rvert = l$	$\xrightarrow{r_1}$	
		Generate random $r_2, \lvert r_2 \rvert = l$ (session secret) $M_1 = t \oplus r_2$ mask the tag pseudonym t with r_2 Prepare challenge for BE: $M_2 = f_t(r_1 \parallel r_2)$
	$\xleftarrow{r_1, M_1, M_2}$	
Tag authentication: In the back-end DB, find $t : M_2 = f_t\left(r_1 \parallel (M_1 \oplus t)\right)$ Recover $r_2 = M_1 \oplus t$ Prepare response to the tag's challenge: $M_3 = s \oplus f_t(r_2 \parallel r_1)$		
	$\xrightarrow{r_1, M_3, M_2}$	
Update state: $\hat{s} \leftarrow s$ $\hat{t} \leftarrow t$ $s \leftarrow (s \ll l/4) \oplus (t \gg l/4) \oplus r_1 \oplus r_2$ $t \leftarrow h(s)$		*Back-end authentication:* Recover $s = M_3 \oplus f_t(r_2 \parallel r_1)$ if $h(s) = t$, *Update state:* $t \leftarrow h\left((s \ll l/4) \oplus (t \gg l/4) \oplus r_1 \oplus r_2\right)$

Figure 4.5: The revised Song and Mitchell's protocol (RSM).

in Figure 4.5). Therefore, RSMP does not provide mutual authentication.

The authors claim that both RSM and RSMP can withstand several attacks, such as tag impersonation, replay attack, man-in-the-middle attack as well as the DoS attack (in the sense of selective blocking of messages between the the back-end and the tag). Moreover, the protocols are backward untraceable in case of tag compromisation. Forward untraceability is provided by RSMP after the update procedure is performed.

Protocol Assessment. Similarly to the OSK protocol, RSM and RSMP provide tag's anonymity and untraceability against terminals. The latter are not required to maintain any private information pertaining to tags. Revocation can be easily performed as well. Fine-grained billing can be straightforwardly implemented, since each interaction in the front-end is controlled and registered by the back-end. As in the case of OSK, the back-end fully identifies every tag on each authentication session. Therefore, RSM and RSMP only address the first two attacker types from the attacker model in Section 3.6. Unlike OSK, RSM and RSMP are stateful by design with respect to the state of the secret t shared between tags and the back-end. Moreover, a very important property of RSM is that it enables mutual authentication. Similarly to OSK, RSM suffers from scalability issues, since the complexity of tag authentication is linear with the number of tags in the system: $O(n)$. However, unlike OSK, RSM in its initial version does not impose a hard limit on the number of sessions for tag authentication. The extended version of RSM – RSMP – is far more efficient in terms of tag authentication (constant time complexity) but it requires a system update when the pseudonym pool is exhausted. Moreover, in contrast to RSM, RSMP does not provide mutual authentication.

The main properties of RSM and its extension RSMP are summarized in Table 4.5.

Table 4.5: A summary of the main properties of RSM and RSMP [91]. Advantages are marked with a ✓sign. The general number of e-tickets circulating in a system is denoted n.

	RSM	**RSMP**
tight coupling between terminals and the back-end	yes	yes
anonymity and untraceability against terminals	yes ✓	yes ✓
mutual authentication in the front-end	yes ✓	no
fine-grained billing is feasible (as add-on)	yes ✓	yes ✓
terminals are trusted	no ✓	no ✓
the back-end is trusted	yes	yes
scalability issues: $O(n)$	yes	no ✓

4.2.2.2 Solutions Based on Symmetric Cryptography: A Short Summary

In this section, the protocols based on symmetric cryptography where back-end is assumed to be always online (tightly-coupled systems) were discussed. The most important representatives of this class of protocols were presented and analyzed in detail, namely the ones due to Okubo *et al.* (**OSK**) [80] and to Song and Mitchell (**RSM**) [91]. The enhanced versions of the aforementioned basic protocols were discussed as well: **OSK-A** due to Avoine *et al.* [5], **OSK-N** due to Nohara *et al.* [81] as well as **RSMP** due to Song and Mitchell [91]. The other approaches based on symmetric cryptography presented in the taxonomy in Figure 4.2 (Alomair *et al.* [83], Molnar and Wagner [84], and Cheon *et al.* [78]) were shortly discussed in the introduction (see Section 4.2.1).

RSM(P) and **OSK-A** deserve special attention as the protocols satisfying most of the core requirements for ESPT discussed in Section 3.5. Unlike OSK-A, RSM(P) is stateful with respect to synchronization of a tag secret between an e-ticket and the back-end. Both approaches do not require terminals to be trusted. Back-end, however, is a fully trusted entity in the system. Therefore, the third attacker type (see attacker model in Section 3.6) is addressed in neither protocol. In order to enhance the front-end performance, a certain type of precomputation is used in each approach. In case of RSMP, it is a pool of one-time pseudonyms. In OSK-A, the search time is optimized using a special time-memory trade-off.

4.2.2.3 Solutions Based on Asymmetric Cryptography

Most of the privacy-preserving solutions in the area of RFID authentication are based on symmetric cryptography and hash functions since these cryptographic primitives are more efficient and feasible to implement on a constrained device. However, the latest advances in technology have made it possible to leverage certain public key operations on RFID tags by this vastly expanding the design choice for new privacy-preserving solutions. In what follows, two approaches, which were specifically designed for ESPT, are going to be reviewed and analyzed.

Peng and Bao (PB) [94]

In [94], Peng and Bao addressed the issue of privacy protection (in the first place, tracking resistance) by proposing two token-based protocols (PB-1, PB-2). Similarly to RSMP discussed in the previous section, both PB-1 and PB-2 are stateful and based on one-time tokens which are presented to the terminal on each check-in/out for validation. There is an important caveat, however, concerning PB protocols: they are specifically tailored to provide support exclusively for one way trips. Moreover, each one way trip must be equally billed (one token for check-in and one for check-out) regardless of the distance traveled, trip duration or time of day. There are four main actors that are considered in [94]: e-ticket, vending machine, terminal (validator), and finally the back-end.

Both protocols developed by Peng and Bao mainly consist of three stages: *(1)* e-ticket acquisition (or alternatively e-ticket top-up) at a vending machine, *(2)* e-ticket registration in the back-end database (or propagation of the top-up event to the back-end), and finally *(3)* direct usage of transport services (check-in/out events handling). The first protocol (PB-1) is solely based on a one-way hash function which is used to mask the secret tokens (pertaining to a certain e-ticket) stored in a hashed form in the back-end database. In this case, the vending machine must be fully trusted since the secret travel tokens are generated by it. The back-end, however, is not able to profile individual users, since the tokens from different users are delivered to it in bulk and in a mixed form. It learns, therefore, only the mere fact that a certain one-time token is valid and does not know to which particular e-ticket is pertains to. The second protocol (PB-2) is claimed to be more privacy friendly, since it does not require vending machines to be trusted. At the same time, the back-end is able to link check-in/out events (but still is not aware of the e-ticket ID to which these events pertain to). PB-2 is based on hash chains (using a cryptographic hash function) and asymmetric probabilistic encryption with homomorphic properties (the latter is performed with the assistance of a vending machine). In what follows, PB-2 is reviewed and analyzed in more detail.

The privacy-preserving solution presented in [94] is mainly based on one-time tokens generated as the members of a hash chain and their further asymmetric probabilistic encryption with a public key of a transport authority. The latter is performed according to Paillier cryptosystem [95] with assistance from the vending machine, since it is assumed that an RFID chip is not able to perform all operations required for this type of encryption. As it already has been mentioned above, PB-2 can be divided into three stages according to which the protocol is reviewed below.

1. E-ticket acquisition (or top-up). Actors involved: *(a) user (e-ticket)*, *(b) vending machine.*

In order to obtain an e-ticket with travel permission for t trips (or top-up an existing one for t trips), a user pays the respective price at a vending machine. The following steps are performed afterwards. Firstly, the vending machine generates $2t$ encryptions of 0 under the public key of a TA according to Paillier's probabilistic[1] scheme: $E = \{e_i : e_i = E_i(0)\}_{2t}$. The notation $E_i(0)$ means the i-th encryption of 0 under the public key of a TA (the latter is assumed to be implied and therefore not explicitly shown in the formula). The e-ticket in turn randomly chooses a seed s to generate $2t$ secret tokens a_i using a cryptographic hash function h:

[1] Therefore, the obtained ciphertexts are indistinguishable (IND-CPA security) under the so-called decisional composite residuosity assumption.

$a_0 = h(s), a_i = h(a_{i-1}), i = (0, 2t]$. The set of secret tokens $A = \{a_i : a_i = h(a_{i-1}), a_0 = s\}_{2t}$ never leaves the e-ticket space. Then the tokens from A are encrypted using the cryptographic material in E what is referred to as vending machine assistant encryption due to Paillier in this section. According to Peng and Bao [94], it can be performed by simple element-wise multiplication of tag-side generated tokens A by precomputed cryptographic material E received from the vending machine:

$B = \{b_i : b_i = a_i \cdot e_{\pi(i)}\}_{2t}, i = [0, t-1]$, where $\pi(i)$ denotes some permutation used to randomize the order of the resultant encrypted tokens (against the vending machine). The authors of [94] claim that the elements of B thus constitute a valid encryption according to Paillier's scheme (under the public key of a TA). The e-ticket, therefore, does not have to perform all the necessary operations required by this partially homomorphic scheme (including exponentiation) but rather a single modular multiplication for each token. However, it is not clear how exactly the authors make use of the additive homomorphic property of the Paillier's scheme, since in order for the encryption[1] $B = A \cdot E$ to be valid (and therefore to correspond to the sum of respective plaintexts, i.e. $A + 0$) the elements of the set A (tag-side generated tokens) should be either already transformed into a valid Paillier encryption or exponentiated accordingly (for details see homomorphic properties of Paillier's Cryptosystem [95]).

Afterwards, an e-ticket stores the encrypted tokens B in its secret memory in a stack with b_0 being at the bottom and b_{2t-1} at the top. The last element of the initial token hash chain a_{2t} is then submitted to the vending machine.

2. E-ticket registration in the back-end database (alternatively, propagation of the top-up event to the back-end). Actors involved: *(a) vending machine*, *(b) the back-end database.*

Having received a_{2t} from the e-ticket, the vending machine forwards it together with the number of acquired trips ($2t$) to the back-end. In the back-end, the respective record is registered: $R_j = (a_{2t}, 2t), j = [1, n]$ (n is the number of all tickets registered in the system) and the user can start the journey. Note that no PII are required (and hence stored) by the system to be able to operate. The notations used during protocol description are summarized in Table 4.6 for clarity.

3. Usage of transport service. Actors involved: *(a) user (e-ticket), (b) terminal (validator)*, and *(c) the back-end database.*

On entering the transport system, a user checks in by presenting his/her e-ticket to a terminal[2] (validator). Check-in technically implies the following operations. *(1)* A terminal reads out the encrypted token b_j currently stored at the top of the e-ticket token stack[3] and forwards it to the back-end database. *(2)* The latter decrypts the received token and hashes the decrypted value r_j to find a match in the database. If the respective record with $h(r_j)$ is found, the user is allowed to enter the vehicle. Therefore, in order to validate itself, an e-ticket delivers a pre-image of the corresponding hashed value stored in the back-end database to the terminal

[1] Multiplication in the cipthertext space is performed modulo n^2 with the corresponding addition in plaintext modulo $n, n = pq$ (see [95] for details). In the notations used during the protocol analysis, this is omitted for brevity.

[2] Note that a terminal (or validator) is not a vending machine. The former validates travel permissions stored on an e-ticket. The latter serves to issue these to a user.

[3] Only one element (the outer one) in the stack can be read from the ticket's memory. The other ones are read and write protected.

Table 4.6: The second protocol suggested by Peng and Bao (PB-2): main constituents.

Notation	Role in the protocol
$E = \{e_i : e_i = E_i(0)\}_{2t}$	the set of probabilistically encrypted zeros (due to Paillier) generated by a vending machine (used for vending machine assisted encryption);
$A = \{a_i : a_i = h(a_{i-1}), a_0 = s\}_{2t}$	ticket-side secret tokens; they never leave the e-ticket space unencrypted except of a_{2t} which is used for bootstrapping at the back-end side;
$B = \left\{b_i : b_i = a_i \cdot e_{\pi(i)}\right\}_{2t}, i = [0, t-1]$	Encrypted tokens (calculated and stored at the e-ticket side in a stack) after vending machine assisted encryption;
$R_j = (a_{2t}, 2t), j = [1, n]$	a record sent to the back-end by the vending machine upon registration of an e-ticket j (n is the total number of the e-tickets registered in the system);

each time a user performs check-in/out. On successful validation, the corresponding "old" record in Rj maintained by the back-end is substituted with its preimage and the trip counter (initially $2t$) decreased by one. *(3)* After being read out, the transmitted token is deleted from the e-ticket memory (to prevent tracking) and is substituted by the next one from the stack: b_{j-1}. Thus, on each validation, the e-ticket moves the tokens up the stack which corresponds to traversing the respective hash chain[1] backwards (to its first element).

Protocol Assessment. Both protocols (PB-1 and PB-2) can withstand replay attacks (due to one-time tokens). In case of a more complex protocol (PB-2), neither vending machines nor terminals (validators) can track e-tickets. Vending machines solely possess the knowledge of the first token a_{2t} (which is the end of a hash chain). Therefore, provided that the underlying hash function is secure, no tracking is possible even in case the vending machine, which assisted in issuing the tokens, and terminals collude. The back-end, however, is able to link different check-in/out events pertaining to a single e-ticket. However, provided that no exogenous knowledge about the user and the respective e-ticket[2] is available to the back-end, no PII can be inferred from check-in/out events.

In case of the simpler protocol (PB-1), which considers random tokens issued to for each trip ($2t$ tokens for t trips) and does not require asymmetric encryption, the back-end cannot link different check-in/out events. The vending machine which issued the respective tokens must be trusted, however (since the tokens are not further encrypted or randomized at the e-ticket side). The terminals which do not collude with the respective vending machines, are not able

[1] Initial set of unencrypted tokens $\{a_j\}$ form a hash chain, see the process of tokens creation described earlier as well as Table 4.6.

[2] For example, observing a user performing check-in/out and recording the respective token used for this. The token will point to the respective record R_j in the database pertaining to e-ticket j.

to link transactions and track e-tickets.

The complexity in terms of required cryptographic operations is similar for both protocols and is constant time. In case of PB-1, the back-end performs a simple look-up (involving no additional cryptoghraphic operations) to find the token forwarded by the terminal in the database. PB-2, however, requires one decryption (Paillier) and one hash operation for each token received from a terminal following the look-up in the database. Therefore, the number of cryptographic operations required for PB-2 does not depend on the number of e-tickets in the system.

The downside of the two solutions presented in [94] is a very coarse-grained billing supported by the system (any ride costs 2 tokens) which prevents the effective implementation of flexible travel tariffs and loyalty programs. Moreover, it must be ensured that token generation is collision free during each issuance of a travel permit for each e-ticket in a system. Provided that this procedure is performed frequently (one-time tokens for each ride), it may impose additional requirements on the token size and collision checks. The main properties of both PB-1 and PB-2 are summarized in Table 4.7.

Table 4.7: A summary of the main properties of PB-1 and PB-2 [94]. Advantages are marked with a ✓sign.

	PB-1	**PB-2**
tight coupling between terminals and the back-end	yes	yes
anonymity and untraceability against terminals	yes ✓	yes ✓
mutual authentication in the front-end	no	no
fine-grained billing is feasible	no	no
terminals are trusted	no ✓	no ✓
vending machines are trusted	yes	no ✓
the back-end is trusted	no ✓	no ✓
tokens must be collision free (affects scalability)	yes	yes
constant time complexity (in terms of crypto operations)	yes ✓	yes ✓

ESPT Based On E-cash by Heydt-Benjamin et al. (HCDF) [8]

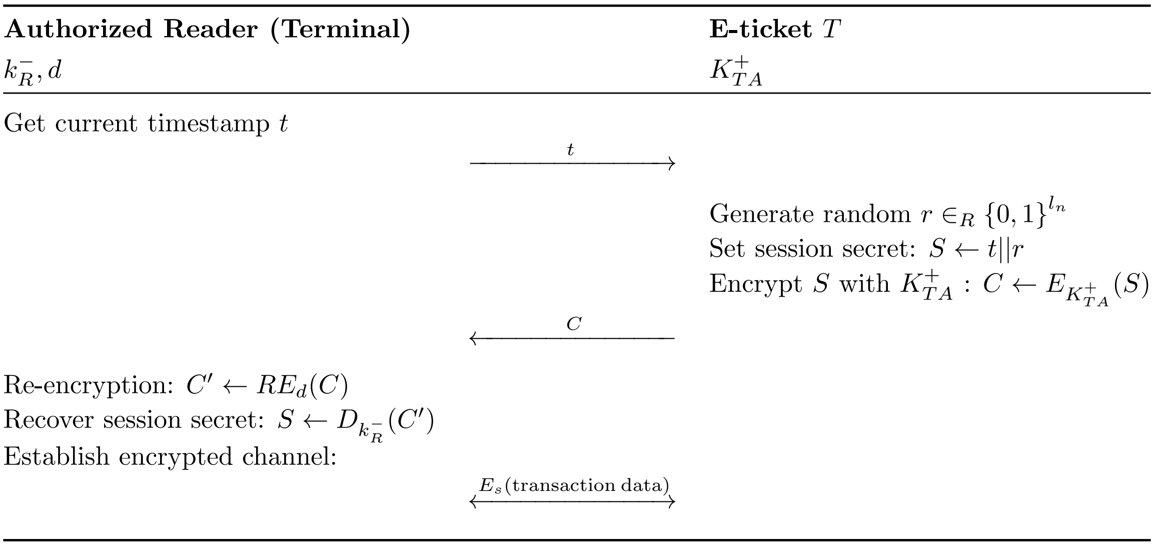
In [8], a privacy-preserving framework for ESPT based on e-cash, anonymous credentials, and proxy re-encryption was presented. The authors considered two types of tickets: *(1)* temporally-bounded (TB) and *(2)* stored-value (SV). Similarly to Peng and Bao [94], Heydt-Benjamin *et al.* holistically consider the generic processes happening in an e-ticketing system. HCDF framework, therefore, essentially includes the following core aspects:

1. *E-ticket acquisition.* It implies a transaction between a vending machine and a user (alternatively between a vending machine and an e-ticket, which has to be topped up).

2. *Entering the ESPT.* A transaction between a faregate (terminal) and an e-ticket is involved.

3. *Leaving the ESPT (exit).* On exit, a terminal proves the validity of the travel permission stored in the e-ticket and calculates the price for a ride (in case of SV ticket).

Before each transaction between an e-ticket and a terminal, a secure channel is established through negotiation of a secret key. It is achieved through a special kind of authentication which is called re-encryption based authentication in [8]. This method is based on delegation keys d which are valid only one day and are issued by a TA to each reader (terminal) on a daily basis. Possession of a such non-expired delegation key allows a reader to re-encrypt the message C previously encrypted with the TA's public key K^+_{TA}, so that the result C' can be further decrypted with the reader's private key k^-_R (see [8] for details). Revocation of a reader in this scenario is achieved by not issuing a new delegation key d to it. The aforementioned re-encryption based authentication can be used to negotiate a session secret S between an authorized reader (having a valid d) and an e-ticket parameterized with a public key of a TA K^+_{TA}, see Figure 4.6. An e-ticket, therefore, has to be capable of performing one encryption under the public key of a TA.



Authorized Reader (Terminal)
k^-_R, d

E-ticket T
K^+_{TA}

Get current timestamp t

$\xrightarrow{t}$

Generate random $r \in_R \{0,1\}^{l_n}$
Set session secret: $S \leftarrow t||r$
Encrypt S with K^+_{TA} : $C \leftarrow E_{K^+_{TA}}(S)$

$\xleftarrow{C}$

Re-encryption: $C' \leftarrow RE_d(C)$
Recover session secret: $S \leftarrow D_{k^-_R}(C')$
Establish encrypted channel:

$\xleftrightarrow{E_s(\text{transaction data})}$

Figure 4.6: HCDF framework: an authentication session between an authorized reader and an e-ticket [8]. k^-_R and d are respectively reader private key and non-expired delegation key. K^+_{TA} denotes public key of a TA.

As it can be seen from Figure [8], each transaction in which an e-ticket is involved (see the aforementioned generic processes) is secured against an observing attacker. In what follows, the main constituents of HCDF framework are going to be concisely presented and analyzed.

1. E-ticket acquisition. Actors involved: *(a) E-ticket*, *(b) Vending machine*, *(c) TA*.
For a *temporally-bounded ticket (TB)*, the following transactions are performed. *(1)* Firstly, an e-ticket pseudonym has to be negotiated between a TA and an e-ticket (it is performed through a vending machine). Each party stores its portion of the respective information pertaining to the pseudonym. At the e-ticket side, the corresponding secret key only known to the current e-ticket is stored. The TA maintains the other portion of information which corresponds to the negotiated pseudonym together with some additional information which later allows it to verify

the validity of the credential issued for this pseudonym (in this case a TB travel permission). The idea strongly resembles e-cash in that a blinded e-ticket-specific pseudonym is used to prevent double spending. If a user does not abuse the protocol, the transactions bound to the pseudonym are unlinkable. *(2)* After the pseudonym has been agreed upon, an attribute-based credential (bound to this pseudonym) is granted to a user proving the validity of the travel permission (the attribute in this case is the time period for which this e-ticket is valid). Note that different validation sessions using the granted credential (when a user checks in/out) are unlinkable provided that a user adheres to the protocol (see [8] for details).

In case of a *stored-value ticket (SV)*, a TA issues several e-cash based tokens to an e-ticket (via a vending machine). The amount of tokens corresponds to the acquired balance thus forming an analog of a virtual purse at the e-ticket side. Needless to say that the tokens are unlinkable during the spend phase (see further) unless double spending has taken place.

2. Entering the ESPT. Actors involved: *(a) E-ticket*, *(b) Terminal (validator)*, *(c) TA*. In case of a *temporally-bounded ticket (TB)*, an e-ticket has to prove to a terminal (in zero-knowledge) that *(1)* it possesses a valid credential to enter the system and *(2)* that this credential is still valid for the current time period. This is performed without the direct involvement of the TA. The terminal does not learn any additional information concerning the e-ticket (and its former transactions) beyond the validity status of the presented (unlinkable) credential.

For a *stored value ticket (SV)*, the so-called entrance cookie C_E parameterized by the station ID is sent to the e-ticket. According to [8], C_E is essentially a one-show credential formed similarly to the token creation for TB tickets (see e-ticket acquisition above). The TA, therefore, has to be online in this case for pseudonym negotiation and issuance of C_E. It is not clear, though, why the authors did not consider the option of performing pseudonym negotiation during e-ticket acquisition as in the case of TB tickets. An entrance cookie could then be directly issued by the terminal during check-in and bound to the e-ticket pseudonym in a blind way (to prevent tracking). In this way, the terminal would not have to additionally relay communication for pseudonym negotiation between the TA and an e-ticket. Moreover, the terminal would learn no auxiliary information transmitted during pseudonym negotiation, since any authorized terminal can decrypt messages encrypted under the public key of a TA using the valid delegation key.

3. Leaving the ESPT (exit). Actors involved: *(a) E-ticket*, *(b) Terminal (validator)*, and *(c) TA*.
For a *TB ticket*, the check-out procedure is carried out in the same way as for check-in (see entering the ESPT above). In case of a *SV ticket*, the entrance cookie C_E has to be revealed to the terminal in order to enable price calculation. The latter is performed by the TA on receiving C_E from the terminal side. The calculated cost (essentially the amount of tokens to be withdrawn) is then sent to the terminal which relays it the e-ticket. The latter spends the specified amount of tokens (which are relayed back to the TA), deletes them from its memory (to prevent double spending) and deletes the current entry cookie (against tracking).

Three main processes described above (e-ticket acquisition, entering/exiting ESPT) form the core of HCDF framework. In [8], the authors additionally define *user transfer* to a different segment of a system (performed similarly to enter/exit), *add value* (similar to e-ticket acquisition), and finally *cancel e-ticket* (technically performed similarly to tokens spend operation and double spending detection). The aforementioned additional processes were omitted in the concise description above for brevity due to their (technical) similarity to the main ones.

HCDF Assessment. Similarly to Peng and Bao protocols (PB-1, PB-2) discussed above, HCDF framework presented by Heydt-Benjamin *et al.* in [8] explicitly considers the public transport scenario. The major difference between the two approaches lies in how the travel permission (tokens) is created. In case of PB protocols, the tokens are either random values (PB-1) or are the elements of a hash chain (PB-2). In HCDF, in contrast, the tokens are e-cash based. As a result, the framework essentially inherits its privacy-preserving properties from the e-cash concept. Namely, the transactions between an e-ticket and a terminal are unlinkable. E-tickets, therefore, cannot be tracked unless double spending has taken place. Each session is protected against an observing attacker through the establishment of a secure channel between an e-ticket and a legitimate terminal. Moreover, unlike the majority of reviewed approaches (including PB protocols), e-tickets remain anonymous and untraceable even towards the TA due to the inherent properties of e-cash (unless double spending occurs). Therefore, the framework provides protection against all three types of attacker discussed in Section 3.6 (outsider, terminals, TA). This is an essential advantage provided by HCDF. At the same time, one the most significant disadvantages of this framework is the inability to support fine-grained billing and hence different loyalty programs attractive to customers. Moreover, the efficiency of the proposed approach is questionable. Firstly, in order to prove the validity of possessed credentials, the e-ticket has to able to perform an interactive zero-knowledge proof which may be computationally prohibitive for many constrained devices. Unfortunately, the authors did not back-up their concept with a proof-of-concept implementation demonstrating the applicability of their solution to the e-ticketing scenario (this was rather left for the future work). Secondly, in case of a stored-value ticket, the price for a ride is calculated on the fly in the back-end which is likely to introduce additional delays during check-out. Thirdly, a full-fledged mutual authentication in its classic sense is not performed in the front-end, since an e-ticket is not authenticated to the authorized reader[1]. However, it should be noted that by proving the possession of a valid travel credential later in the protocol, an e-ticket indirectly authenticates itself. Lastly, it was not clearly stated how the delegation key d is actually get expired after 24 hours, since the corresponding public key of a TA (stored at the e-ticket side) remains the same. HCDF framework is summarized in Table 4.8.

4.2.2.4 Solutions Based on Asymmetric Cryptography: A Short Summary

Leveraging the properties of public key cryptography provides qualitatively new ways of enhancing privacy in ESPT as well as greatly expands the choice of available security measures. In this section, two privacy-preserving approaches based on asymmetric cryptography which are specifically targeted at ESPT were shortly presented and analyzed. Namely, Peng and Bao (**PB**) protocols [94] and **HCDF** framework due to Heydt-Benjamin *et al.* [8] were considered. In both cases, it was assumed that back-end is always online (tightly-coupled systems) and can process in real-time the front-end requests originated from interaction between terminals and e-tickets (check-in/out, e-ticket acquisition/top-up). Unlike PB-2 where tokens form the

[1] Indeed, the mere knowledge of the public key of a TA is not sufficient for e-ticket authentication

Table 4.8: Main properties of HCDF [8] summarized. Advantages are marked with a ✓sign.

Criterion	Assessment	
tight coupling between terminals and the back-end	yes	
anonymity and untraceability against terminals	yes	✓
mutual authentication in the front-end	yes (implicit)	✓
fine-grained billing is feasible	no	
terminals are trusted	no	✓
the back-end is trusted	no	✓
efficiency considerations due to ZKP during check-in/out	yes	

elements of a hash chain (and are further encrypted element-wise in a non-deterministic way), a travel permission in HCDF framework is mainly based on the e-cash concept. As a consequence, even the TA is not able to track e-tickets and therefore to profile users (unless double spending of a travel token occurs). This comes at a cost, however. Proving the validity of an e-cash based travel permission in zero-knowledge (each time on check-in/out) is likely to impose a severe burden on the minimal computational capabilities of end user devices carrying an e-ticket. PB-2, in contrast, is more lightweight and does not involve any cryptographic operations from a tag during check-in/out. Security properties of this protocol, however, are notably weaker than those of HCDF (no mutual authentication, any terminal can query e-tickets). Moreover, the back-end must be completely trusted in PB-2, whereas HCDF enables to relax trust assumptions for the back-end due to the inherent properties of e-cash (unlinkability of transactions, etc). This, however, completely prevents the implementation of flexible travel policies and hence the adoption of fine-grained billing in HCDF, since it is by design not possible to correlate check-in/out events by any entity in the system.

4.2.3 Tightly-coupled Systems: Review Summary

In this section, the privacy-preserving solutions were discussed where it is assumed that the back-end is always online and can process the front-end requests originating from check-in/out events in real time (see tightly-coupled systems in the taxonomy depicted in Figure 4.1). The reviewed approaches can be further divided into the ones leveraging the properties of asymmetric cryptography[1] and the solutions based on symmetric cryptography (see the respective taxonomy in Figure 4.2). Each of the solutions has its own benefits and disadvantages. Taking into account the requirements discussed in Section 3.5 as well as the derived evaluation criteria (see Section 4.1), the following approaches stand out: OSK protocol enhanced by *Avoine et al.* **(OSK-A)** [5], revised Song and Mitchell's protocol **(RSMP)** [91] (symmetric-key based) and a privacy-preserving framework for ESPT based on e-cash due to Heydt-Benjamin *et al.* **(HCDF)** [8]. The first two solutions satisfy the majority of the core criteria but fail to provide privacy-protection against the back-end (therefore, only considering the first two types of attacker discussed in Section 3.6). HCDF framework, to the contrary, covers all three attacker types but fails to provide the support for fine-grained billing which

[1] The distinction is made with respect to cryptographic operations performed on the e-ticket side.

is essential in modern ESPT (see the respective discussion in Section 3.5.2). Moreover, all approaches reviewed so fare are inherently based on the assumption that the back-end can process requests originating from check-in/out events in real time. Taking into account the size of a widely-distributed e-ticketing system and the tight timing requirements posed to the processing of check-in/out events, such system architecture is very likely to result in the back-end becoming a bottleneck (see also the introductory discussion in Section 4.2.1). Moreover, scaling issues together with the additional requirements posed to the connectivity of widely distributed (mobile) terminals arise.

Summarizing the analytic review performed in this section, it can be concluded that none of the reviewed solutions fully satisfies the set of core requirements (see Section 3.5) posed to the target ESPT. Therefore, in the next section, the solutions which consider the back-end decoupled from real-time interaction in the front-end (see Figure 4.1) are reviewed and analyzed.

4.3 Related Work Analysis: Decoupled Systems

4.3.1 An Introductory Discussion

This section considers the most generic privacy-preserving solutions which are based on the system architecture with the back-end decoupled from the front-end processes. Depending on the degree (and frequency) of back-end involvement into the front-end performance, decoupled systems can be further divided into *(1)* loosely-coupled and *(2)* semi-coupled systems (see also the general taxonomy of the reviewed solutions in Figure 4.1). In case of the *loosely-coupled* systems, relatively infrequent updates from the back-end side are necessary to enable proper functionality in the system front-end. That is, check-in/out events can be served and processed locally at the terminal side. The updates are received from the back-end on a monthly or quarterly basis (for example, certificate updates, system patches, etc.). A *semi-coupled* system, however, requires frequent updates regularly performed on a daily or even hourly basis (blacklist updates, for example).

Decoupled systems (both loosely-coupled and semi-coupled) have the following advantages over the close-coupled ones:

1. *Better scaling (due to loose-coupling).* Since back-end and front-end are loosely coupled, the former does not have to process the requests originating from check-in/out events in real time. This results in much better scaling compared to a tightly-coupled system.

2. *Terminal-side e-ticket validation (efficiency).* In a decoupled system, terminals can locally validate an e-ticket without depending on the back-end's answer. As a result, the front-end processing of check-in/out events is more efficient (both in terms of communication costs and processing time).

3. *Relaxed requirements to terminals interconnection.* In contrast to tightly-coupled systems, back-end does not have to maintain constant (24/7) reliable connection with the terminals. Therefore, link costs are considerably lower in case of decoupled systems (especially if mobile on-site terminals are considered).

The aforementioned advantages come at a cost, however. Compared to tight-coupling, the decoupled architecture implies the following disadvantages:

1. *Higher requirements to the terminal processing power.* Since the whole validation cycle is performed in the system front-end, terminals have to possess enough resources to carry out all operations required to validate an e-ticket (including the ones involving cryptography which may be rather computationally expensive).

2. *Substantially increased effort for managing the decentralized infrastructure.* Due to loose-coupling, additional effort is required to *(a)* keep the system up-to-date, *(b)* collect travel records from terminals, and *(c)* resolve possible conflicts.

3. *Requirements conflict: privacy versus local validation.* In order to validate an e-ticket locally (possibly including blacklist check), terminals are likely to operate on (partially) identifiable data. This, however, is in direct conflict with the privacy requirement (see Requirement *1b* in Table 3.4), since terminals must be prevented from deriving any PII during validation.

The generic representatives of both loosely-coupled and semi-coupled systems are going to be briefly presented and analyzed within the following review. Similarly to the taxonomy of tightly-coupled systems (see Figure 4.2), the one for the decoupled systems is presented in Figure 4.7. The trivial case of each terminal locally managing the entire tag database (under the category of symmetric cryptography in Figure 4.7) is mentioned solely for completeness reasons and is not going to be discussed due to its straightforwardness.

4.3.2 Decoupled Systems: A Detailed Review

4.3.2.1 Solutions Based on Symmetric Cryptography

TanSL Protocols by Tan et al. [96]

Tan *et al.* addressed the problem of local, terminal-side e-ticket validation by presenting two protocols in their paper [96] collectively referred to as TanSL in this dissertation. The primary goal was to achieve what the authors called "serverless search and authentication" in the area of RFID. More specifically, the solution provides for privacy-respecting terminal-side tag identification together with mutual authentication. The solution falls into the category of loosely-coupled systems (see Figure 4.7), since after the initialization phase no further interaction between terminals and back-end is required to validate tags. Three main parties are considered in TanSL: an RFID tag (e-ticket), a terminal (reader) as well as a trusted certificate authority (CA). In order to remove the need for real-time persistent database in the back-end, the authors of [96] suggest that the terminal-specific tag access lists (AL) are computed and uploaded to each terminal by a trusted CA (via a secure connection on successful terminal-to-CA authentication). ALs are *bound* to each terminal. Namely, for each terminal (reader) r_i CA creates a *terminal-specific* access list AL_i with the elements L_{ij} corresponding to each tag t_j registered in the system. Namely, $AL_i = \{L_{ij} : L_{ij} = h(r_i||s_j)\}_n$ where $j = [1, n]$ with n being the number of all tags registered in the system, h denotes a cryptographic hash function, r_i is the terminal ID and s_j represents the tag's secret. The latter is known only to CA and to each

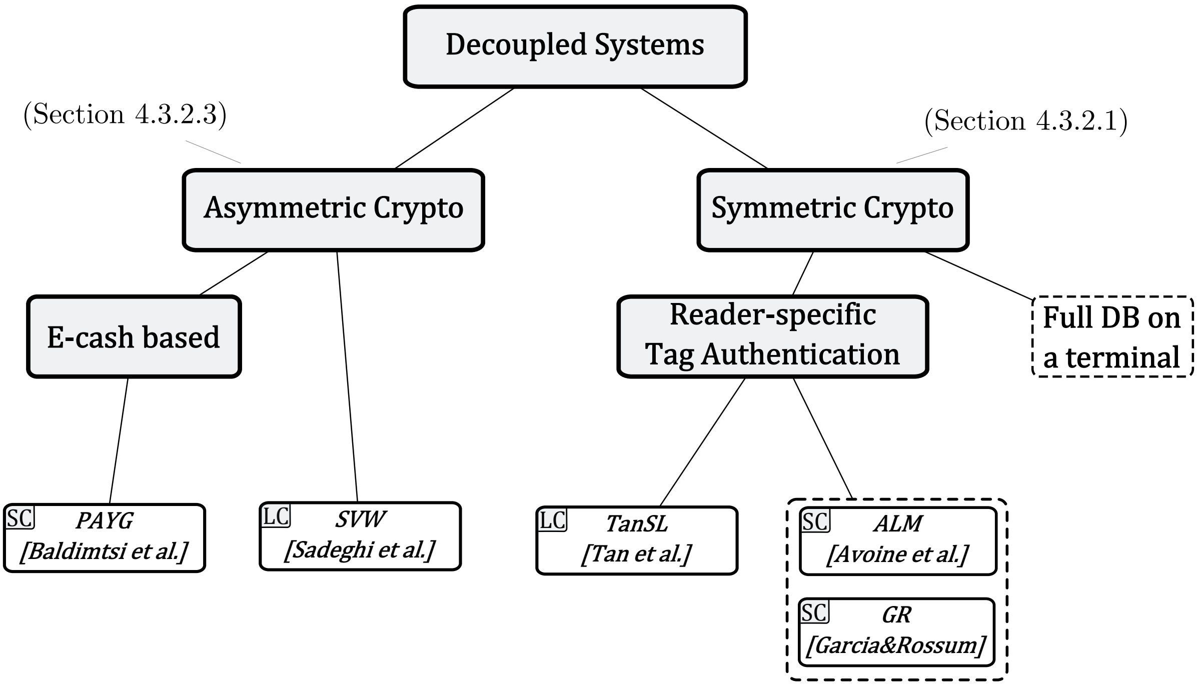


Figure 4.7: Solutions taxonomy: decoupled systems. LC stays for loosely-coupled architecture, SC for the semi-coupled one.

particular tag itself. Every terminal R_i, therefore, maintains its specific AL_i where each record L_{ij} points to the ID of a particular tag t_j (tag's ID is not secret, in contrast to its secret s_j). Notations used within the current discussion are summarized in Table 4.9 for clarity. During check-in/out, each authorized terminal *fully identifies* a tag if the respective record was found in AL. At the same time each terminal stays unaware of the underlying tag's secret s_j, which was used by CA to create the respective AL, and operates only on the corresponding hash value (see L_{ij}). This is essential for the security of the system in case a terminal becomes compromised. An "authorized terminal" in this case means it possesses the respective access list AL_j. Moreover, once authenticated to CA (e.g. by means of well-established signature-based authentication), all terminals are considered to be *trusted*.

The aforementioned terminal-specific access lists are the basis for the two privacy-preserving protocols with mutual authentication developed by Tan *et al.* [96], namely TanSL-1 and TanSL-2. The first protocol is based on challenge-response for mutual authentication and subsequent tag identification. In TanSL-2, authentication is implicit, since only an authorized terminal is able to *(1)* correctly interpret the tag's message and *(2)* extract the tag's ID contained therein. TanSL-2, therefore, does not provide mutual authentication in its classic sense.

TanSL-1. Prior to engagement into the challenge-response phase, a terminal and a tag exchange nonces, n_r and n_t respectively. The terminal additionally sends its ID r_i to the tag. Thus, both parties possess the necessary basis to create and reproduce a challenge. Firstly, a tag calculates the so-called challenge material (a basis for challenge) $C = h(h(r_i||s_j)||n_r||n_t)$ of

Table 4.9: Notations used during the discussion of TanSL protocols.

Notation	**Meaning**
$h()$	a cryptographic hash function;
CA	a trusted certificate authority;
AL_i	access list for the i^{th} terminal (reader) r_i;
r_i, t_j	ID of the i^{th} reader (terminal) and the j^{th} tag respectively;
s_j	the secret of the j^{th} tag (known only to the tag itself and to CA);
L_{ij}	an element in AL_i, equals to $h(r_i \| s_j)$;
n	the number of all registered e-tickets in a system;
n_r, n_t	nonces generated by reader and tag respectively;
C	challenge material, equals to $h(h(r_i \| s_j) \| n_r \| n_t)$;
U	the first q elements of C, namely $C_0, C_1, \ldots, C_{q-1}$;
C'	challenge material recovered at the terminal side;
l	length of the hash function output (and consequently of C);
k	the number of randomly chosen positions among the last $(l - b)$ elements in C;
$ques_r, ques_t$	strings of k random positions pointing to the elements of C (a question).

length l. Note that $h(r_i||s_j)$ in C corresponds to the respective entry L_{ij} which an authorized reader has in its access list AL_i. Then the tag randomly chooses k positions ($k \leq (l-b)/2$) among the last $(l-b)$ elements of C by this obtaining a random string also called a tag's question: $ques_t = \{\, p \,:\, p \in_R [l-b, l]\,\}_k$. This string together with the first q elements of C is sent to the reader as a challenge. The challenge semantics, therefore, is in a way similar to the cut-and-choose method known from other application areas.

On receiving the challenge, the terminal *(1)* tries to reproduce it locally using its access list (element-wise) and *(2)* each time compares the first q bits of the result with U received from the tag. More specifically, for each registered tag t_j (including the one trying to authenticate), CA has already precomputed $h(r_i||s_j)$ for each terminal r_i and uploaded these values in form of an AL . Therefore, for each entry in AL, the terminal can recover the challenge by performing the same computations as the tag did to compute C. For each recovered challenge C', the terminal compares its first q elements with U received from the tag. The procedure is sequentially repeated until a match is found. Having found the match, the terminal computes an answer ans_r to the tag's challenge by sending the actual elements with the position's numbers defined in $ques_t$. If the answer is correct, the terminal has successfully authenticated itself to the tag. A tag is authenticated to a reader in a similar way. If several entries corresponding to U are found, the previous steps are repeated till the tag is singulated. TanSL-1 is presented in Figure 4.8.

Should a certain terminal become compromised, the possible damage to the system (including negative consequences for privacy) is substantially reduced. Terminal compromisation implies that all information stored at the terminal side is exposed to the attacker. However, even being in the possession of a terminal-specific access list and therefore having the knowledge of the respective mapping to tag IDs, such an attacker would have access only to the check-in/out events served by this particular terminal which has been compromised. Tag sessions with other terminals (at other locations in the system) will remain intact, since each terminal uses its specific access list which is bound to its ID r_i. Therefore, according to the authors,

Terminal (Reader) r_i		**Tag** t_j
$r_i,\ AL_i = \{L_{im} : L_{im} = h(r_i \| \| s_m), m = (0,n)\}_n$		s_j
generate nonce n_r	$\xrightarrow{n_r,\ r_i}$	
		generate nonce n_t; *Compute a challenge for the reader:* $C = h(h(r_i \| \| s_j) \| \| n_r \| \| n_t),\ \|C\| = l$; put $U = C_0, C_1, \ldots, C_{q-1}$ (first q bits of C) prepare $ques_t = \{\, p\ :\ p \in_R [l-b, l]\,\}_k$; $C_{t\to r} = (ques_t,\ U)$
	$\xleftarrow{C_{t\to r},\ n_t}$	
Recover the challenge: $\forall L_{im} \in AL_i, m = (0,n)$ compute: $C' = h(L_{im} \| \| n_r \| \| n_t)$; put $U' = C'_0, C'_1, \ldots, C'_{q-1}$; compare $U' \overset{?}{=} U$; `if` a single match is found (e.g. for L_{ij}) compute the answer: $ans_r = \{\, C'_m\ :\ m \in ques_t\,\}_k$ `else if` several entries are found start the protocol again; `else` fill the answer with random values: $ans_r = \{\, p\ :\ p \in_R \,\}_k$ *Compute a challenge for the tag:* generate question: $ques_r$ (similarly to $ques_t$);		
	$\xrightarrow{ans_r,\ ques_r}$	
		Reader authentication: check if ans_r is correct; check that $ques_r \neq ques_t$; compute ans_t similarly to ans_r;
	$\xleftarrow{ans_t}$	
Tag authentication and identification: `if` ans_t is correct tag t_j is successfully authenticated and identified `else` reject the tag.		

Figure 4.8: TanSL-1. C_x denotes the x-th bit of a bit string C .

the information leaked from a single terminal does not help an attacker to defeat privacy in the whole system (or in the other parts of a system).

TanSL-2. The second version of the protocol is more efficient, simpler but at the same time less privacy-preserving (due to tracking implications). Moreover, no explicit reader-to-tag authentication is considered in this case. In contrast to TanSL-1, no challenge-response mechanism is used. After the nonces have been exchanged, the tag t_j: *(1)* computes $V = h(h(r_i||t_j))$ corresponding to $h(L_{ij})$ at the terminal side; *(2)* computes C as in TanSL-1 and performs XOR operation of the result with its ID: $W = C \oplus t_j$; *(3)* sends the first n bits of V together with W to the terminal. At the terminal side, the entries L_{ij} matching the first d bits of V are determined at first. Since V is static for each (terminal, tag) pair, terminals can additionally apply pre-computation by hashing the elements of AL_i in advance. Then for each entry found, C' is recovered as in TanSL-1 and subsequently XORed with W received from the tag. If the correct[1] tag's ID t_j is obtained as a result of this operation, the tag is considered to be authenticated and identified. Else the aforementioned steps are performed for the next entries found until the right ID is obtained. TanSL-2 is presented in Figure 4.9.

Terminal (Reader) r_i r_i, $AL_i = \{L_{im} : L_{im} = h(r_i\|\|s_m), m = (0,n)\}_n$		**Tag** t_j s_j
generate nonce n_r	$\xrightarrow{n_r,\ r_i}$	
		generate nonce n_t; compute $V = h(h(r_i\|\|s_j))$ put $F = V_0, V_1, \ldots, V_{d-1}$ (first d bits); compute $W = h(h(r_i\|\|s_j)\|\|n_r\|\|n_t) \oplus t_j$
	$\xleftarrow{n_t, F, W}$	
$\forall L_{ij} \in AL_i$ check if the first n elements of $h(L_{ij})$ are equal to F. For each match: 1. Recover the challenge C'; 2. Compute $ID' = C' \oplus W$; 3. Compare ID' with corresponding t_j.		

Figure 4.9: TanSL-2.

Protocol Assessment. Two protocols developed by Tan *et al.* allow for terminal-side e-ticket validation without the need to consult the central database in the back-end. Following the classification introduced in Section 4.3.1, the solution presented in [96] can be further classified as the one falling into the category of loosely-coupled systems. The authorized terminals are initialized with terminal-specific access lists AL which enable local validation. After the initialization, terminals essentially function independently from the back-end and are therefore completely decoupled from the rest of the system. Authorized terminals can *fully identify* tags for which there is the corresponding record in the terminal access list. For this reason, only the

[1] The "correct" tag's ID obtained by XOR operation means that it corresponds to the tag's ID being pointed to by the respective entry L_{ij} at the terminal side.

first attacker type (the outsider, see Section 3.6) is addressed in TanSL protocols. In the first protocol, TanSL-1, the complexity of tag authentication (and identification) is linear in the number of tags which the terminal is allowed to validate. That introduces a serious bottleneck and imposes strict requirements on terminal's processing power. Moreover, if several entries are found which correspond to the tag's message, the challenge-response phase must be repeated until a single match is found. In order to improve efficiency, the second protocol was introduced in [96]. TanSL-2 is not based on classic challenge-response. The search complexity can be kept constant by partitioning the terminal-side access lists according to the first d bits of the corresponding hash of each L_{ij} value. Then on receiving the answer from the tag, a terminal can efficiently determine the group which the current tag belongs to and subsequently perform linear search within this group. This is possible, since the tag additionally sends the first d bits of $h(h(r_i||s_j))$ to the terminal along with the challenge. The side-effect is, however, that tag tracking could be possible if the size of the group is small. Moreover, in TanSL-2, no explicit terminal-to-tag authentication is considered.

Regular billing is not directly considered by the authors but it can be implemented on top of the protocol, since authorized terminals fully identify each tag being validated. TanSL protocols are summarized in Table 4.10.

Table 4.10: A summary of the main properties of TanSL-1 and TanSL-2 [96]. Advantages are marked with a ✓sign.

	TanSL-1	**TanSL-2**
tight coupling between terminals and the back-end	no ✓	no ✓
anonymity and untraceability against terminals	no	no
mutual authentication in the front-end	yes ✓	no
fine-grained billing is feasible (as add-on)	yes ✓	yes ✓
terminals are trusted	yes	yes
the back-end is trusted	yes	yes
scalability issues: linear complexity	yes	no ✓

ALM Protocol by Avoine et al. [97]

The solution presented by Avoine et al. in [97] referred to as ALM in this dissertation is based on TanSL protocols discussed above. In contrast to TanSL, the issue of terminal compromisation is explicitly considered in ALM by introducing the so-called reader counter c_i^r to determine "stale" terminals which have failed to synchronize themselves with the back-end. The latter fact points to terminals compromisation, since in [97] it is assumed that the compromised terminals are out of sync with the back-end. The reason for this is that after a compromised terminal has been detected in the system, all other legitimate terminals receive an update including the new reader counter.

Similarly to TanSL, three main parties can be distinguished in ALM: *(1)* tags, *(2)* terminals, and *(3)* back-end. During the system set up, a tag j is initialized with a unique ID t_j (known to terminals and the back-end), secret (long-term) key s_j, and tag counter c_j^t. The secret key is known only to the respective tag itself and to the back-end. Terminals are unaware of s_j.

Each terminal (reader) i is assigned with a reader ID r_i, reader counter c_i^r, and maintains its access list AL_i. The latter is a set of shared keys between this reader and each tag which is precomputed by the trusted back-end: $\left\{k_{ij} : k_{ij} = E_{s_j}(r_i, c_i^r),\ j = [1, n]\right\}_n$. Here, $E_{s_j}()$ denotes symmetric encryption under the tag's secret key s_j. Therefore, the elements of AL are bound to each particular terminal as in the case of TanSL. Moreover, as it was mentioned above, it is possible to prevent the corrupted terminals from validating the tags by updating the counters of other terminals. Having interacted with the legitimate terminal, each tag updates its counter to the new value and hence will no longer accept the corrupted terminal with an "old" counter value (see Figure 4.10 for details). The notations used during the discussion of ALM are summarized in Table 4.11.

Table 4.11: Notations used during the discussion of ALM.

Notation	Meaning
r_i, t_j	the IDs of the i-th terminal (reader) and the j-th tag respectively;
s_j	long-term secret key of the j-th tag (known only to the tag and to the back-end);
c_i^r, c_j^t	counters of the i-th terminal (reader) and of the j-th tag respectively;
AL_i	access list of the i-th terminal (similarly to TanSL);
k_{ij}	a key shared between the i-th reader and the j-th tag;
n_r, n_t	nonces generated by a reader and a tag respectively;
n	the number of all tags registered in the system.

Summarizing, the following steps can be distinguished in ALM:

1. On-the-fly computation of the shared key k_{ij} on the tag side after receiving r_j, c_i^r, n_r from the terminal.
2. Mutual authentication using challenge-response and the computed shared key.
3. Tag's counter update, if necessary.

The protocol is depicted in Figure 4.10.

If terminal compromisation is determined in the system, the back-end updates the counters of the other legitimate terminals and recomputes the corresponding access lists accordingly. The compromised reader, therefore, would be able to validate only those tags which have not yet interacted with the updated terminals. In the public transit setting, this is a rather rare case[1] (consider check-in/check-out events) provided that *(1)* the number of compromised terminals is much less than the number of the legitimate ones and *(2)* the compromised terminals are distributed in the system (as opposed to the case when, for example, all terminals along a certain transit line in a system have been compromised).

Protocol Assessment. Similarly to TanSL, ALM provides for terminal-based tag validation without the need for a real-time connection to the back-end. In ALM, however, the issue

[1] A tag will eventually interact with a legitimate terminal and update its counter.

Terminal (Reader) r_i		**Tag** t_j
r_i, c_i^r, $AL_i = \big\{k_{ij} : k_{ij} = E_{s_j}(r_i, c_i^r),\, j = [1, n]\big\}_n$		s_j, c_j^t
generate nonce n_r	$\xrightarrow{n_r,\, r_i,\, c_i^r}$	
		`if` $c_i^r \geq c_j^t$ compute $k_{ij} = E_{s_j}(r_i, c_i^r)$; generate nonce n_t; compute $V = E_{k_{ij}}(n_r, n_t)$; `else` reject terminal.
	$\xleftarrow{V}$	
Tag authentication: $\forall k_{ij} \in AL_i$ decrypt V until n_r is extracted; `if` n_r has been successfully extracted; extract n_t; `else` reject the tag.		
	$\xrightarrow{n_t}$	
		Reader authentication: check if the received n_t is correct. *Update tag's counter if needed:* `if` $c_i^r > c_j^t$ update $c_j^t \leftarrow c_i^r$

Figure 4.10: ALM protocol.

of terminal compromisation is more profoundly addressed by introducing counters which are updated each time when the compromisation event is detected. The other properties are similar to TanSL with protocol complexity being linear in the number of all tags in the system, as in TanSL-1. Therefore, compared to certain tightly-coupled protocols (e.g., OSK, RSM), both in TanSL-1 and ALM the complexity issue is *shifted* from the back-end to the terminal side. The main properties of ALM are summarized in Table 4.12.

GR Protocols due to Garcia and Rossum [6]

Garcia and Rossum further elaborated on the issues of tag privacy preservation in case of terminal compromisation. Namely, in [6] the issues of the so-called "forward privacy" (FP) and "backward privacy" (BP) together with the ability of the system to re-synchronise itself after malicious actions of the compromised terminal were considered. FP implies that if a certain terminal gets compromised, it cannot infringe on "privacy" of the past transactions. BP, to the contrary, considers "privacy" of future transactions after the event of terminal compromisation has occurred. "Privacy" in this case refers to the inability of a malicious terminal to trace tags. Moreover, the protocols introduced by the authors (collectively referred to as GR in this dissertation) are stateful by design. Therefore, additional issues of state re-synchronisation after the attack mounted by a compromised terminal were addressed as well. The main application area of GR are e-ticketing systems for public transport where it

Table 4.12: ALM: main properties summarized. Advantages are marked with a ✓sign.

Criterion	Assessment	
tight coupling between terminals and the back-end	no	✓
anonymity and untraceability against terminals	no	
mutual authentication in the front-end	yes	✓
fine-grained billing is feasible	yes (as add-on)	✓
terminals are trusted	yes	
the back-end is trusted	yes	
scalability issues: linear complexity	yes	

is assumed that during a single shift mobile terminals stay offline (with respect to the back-end) and validate tickets locally. Terminal updates are performed when vehicles are in depot (during the out-of-shift time, e.g. at night). In contrast to TanSL and ALM, the access lists (called reader tables T in [6]) are not terminal-bound. However, as in TanSL and ALM, the synchronised terminals can *fully identify* tags. There are also three main parties in the system: tags, terminals, and the (trusted) back-end. On system initialization, each tag initially shares a key k with terminals. At the tag side, an additional parameter k' is stored which state is used to (indirectly) maintain synchronisation with the back-end (through terminals). At the terminal side, the following parameters are maintained: $\tilde{k}$, $h(k', C_0)$, and u. The first parameter is referred to as "the first key of the day" and is updated by the back-end in the following way: $\tilde{k} \leftarrow h(k'+1)$ (it is explained in more detail below). C_0 is a known system-wide constant and u is a bit flag indicating if the tag has been validated during the current epoch (that is, between two synchronisation periods, within the current shift). Note that terminals are unaware of k' the initial value of which is shared between tags and the back-end. The notations used during the discussion of GR are summarized in Table 4.13. The major protocol due to Garcia and Rossum is depicted in Figure 4.11 and functions as follows. Having received the nonce n_r from the terminal, the tag computes an answer c and updates its key k. On receiving c, the terminal checks the following three cases.

Case 1 (accept the tag). For each entry k in its table (access list), the terminal tries to recover the corresponding hash chain and to find a match with c in a similar way as it is done in OSK (see Section 4.2.2.1). The difference to OSK is that a nonce n_r is taken into account (see Table 4.11). If a match is found (rendering case 1 true), the conclusion can be drawn that the tag with the respective ID has been validated for the first time within the current epoch. The terminal then computes an answer m as shown in Figure 4.11. The tag checks if m is correct and if so, further updates k and k'. Then it computes an answer c' and updates k again. At the terminal side, the condition $h(\tilde{k}, n_r) = c'$ is checked. If it is true, u is set to true which signalizes that k' has to be updated in the back-end (namely, $k' \leftarrow h(k')$) which happens when the tables are gathered and analyzed during the off-the-shift time. Note that $\tilde{k}$ at the terminal side is pre-computed be the back-end and equals to $h(k'+1)$. This corresponds to the new value of k updated by the tag after checking the correctness of m (see Figure 4.11).

Case 2 (accept the tag). If the tag has been already validated during the current epoch, the second case evaluates to true. In order to check the respective condition, operations similar to the ones in case 1 are performed but using $\tilde{k}$ instead of k. The reason is that the value of terminal-side $\tilde{k}$ pre-computed by the back-end corresponds to the one of $k \leftarrow h(k'+1)$ updated by the tag after receiving m (see case 1 above and Figure 4.11). If case 2 evaluates to true, m is set to a random value and $\tilde{k}$ is updated as follows to maintain consistency: $\tilde{k} \leftarrow h^{i+1}(\tilde{k})$. On receiving m, the tag computes an answer c' and updates its k. Having received c', the terminal performs no further actions (that is, the value of u remains intact) since case 1 has been evaluated to false.

Case 3 (reject the tag). If no match has been found, neither using k nor $\tilde{k}$, the tag is rejected.

At the end of each epoch, the tables from all terminals are gathered by the back-end and analyzed. Namely, for each tag the latest value of key k is determined. If a table is found for which u is true, then the corresponding $\tilde{k}$ is updated as follows: $\tilde{k} \leftarrow h(k'+1)$ and u is set to false. Moreover, k' is updated in the back-end database: $k' \leftarrow h(k')$. After the aforementioned operations, the updated tables are distributed to terminals.

Table 4.13: Notations used during the discussion of GR.

Notation	**Meaning**
k	the key initially shared between a tag and terminals
k'	the secret key of a tag, shared only with the trusted back-end
C_0	a well known system wide constant
$\tilde{k}$	the "first key of the day", initially $\tilde{k} \leftarrow h(k'+1)$

Protocol Assessment In [6], the terminal-side tag validation was addressed together with the issues of privacy preservation in case of terminal compromisation. The former is ensured through the so-called terminal tables which correspond to access lists in TanSL and ALM. The important difference is, however, that the tables are not terminal-bound as in the two previous protocols. The advantage of it is that the back-end does not have to compute tables individually for each terminal as it is done in TanSL and ALM. Should a certain terminal become compromised, however, additional measures have to be undertaken to protect privacy (see forward privacy and backward privacy discussed above). For this, each tag maintains its secret key k' shared with the trusted back-end which is used to maintain consistency between legitimate terminals and tags and to prevent compromised terminals from tracing tags (across epochs). This is done through regular updates of k' for each new epoch. Moreover, in [6], the issues of self-stabilization after a de-synchronisation attack mounted by a compromised terminal were addressed as well.

Depending on the fact if a tag has already been validated during the current epoch, two cases are foreseen for terminal-side tag validation. Both of them are essentially based on OSK in terms of a match search (see Section 4.2.2.1) additionally taking nonce n_r into account. Therefore, the worst case complexity for tag validation equals to the one of OSK protocol, namely $O(Vn)$ where V is the maximum length of the hash chain and n is the number of all tags registered in the system. Moreover, within a single epoch a tag can be validated a limited number of times (corresponding to V).

Therefore, GR is similar to TanSL and ALM in terms of the following properties: no

Terminal (Reader)		**Tag**
$T : [id, k, \tilde{k}, h(k', C_0), u]$		k, k'
generate nonce n_r	$\xrightarrow{n_r}$	
		compute: $c \leftarrow h(k, n_r)$ update: $k \leftarrow h(k)$
	$\xleftarrow{c}$	
case 1: $\forall k \in T, i \leq V$ search for a match $h(h^i(k), n_r) = c$ if found, compute: $m \leftarrow h(h^{i+1}(k), h(k', C_0))$ **case 2:** $\forall k \in T, i \leq V$ search for a match $h(h^i(\tilde{k}), n_r) = c$ if found, set: $m \leftarrow random$ $\tilde{k} \leftarrow h^{i+1}(\tilde{k})$ **case 3:** if no matches have been found, reject the tag, set $m \leftarrow random$		
	$\xrightarrow{m}$	
		`if` $h(k, h(k', C_0)) = m$ update $k \leftarrow h(k' + 1)$ $k' \leftarrow h(k')$ compute: $c' \leftarrow h(k, n_r)$ update: $k \leftarrow h(k)$
	$\xleftarrow{c'}$	
`if` **case 1** is true and $h(\tilde{k}, n_r) = c'$ set $u \leftarrow 1$ (signal to update k' in the back-end)		

Figure 4.11: GR protocol [6]. V represents the maximum length of a hash chain $h^i()$.

anonymity and untraceability against terminals, trust assumptions, billing as well as coupling nature between the back-end and terminals. However, no terminal authentication towards the tag is considered. Moreover, the worst-case complexity of tag validation is similar to OSK, namely $O(Vn)$. The properties of GR are summarized in Table 4.14.

4.3.2.2 Solutions Based on Symmetric Cryptography: A Short Summary

In this section, the most relevant solutions which can be applied to develop a privacy-preserving decoupled ESPT system based on symmetric cryptography were reviewed and analyzed. Since the decoupled systems can be further divided into loosely-coupled and semi-coupled (see Figure 4.7), the representatives of both sub-categories were reviewed. Thus, the two protocols due to Tan *et al.* [96] referred to as TanSL-1 and TanSL-2 fall into the category of loosely-coupled systems. Whereas TanSL-1 provides for mutual authentication (unlike TanSL-2), it is fairly inefficient in terms of performance (tag validation complexity is linear in the number of all tags

Table 4.14: GR: main properties summarized. Advantages are marked with a ✓sign.

Criterion	Assessment	
tight coupling between terminals and the back-end	no	✓
anonymity and untraceability against terminals	no	
terminals are trusted	yes	
fine-grained billing is feasible	yes (as add-on)	✓
mutual authentication in the front-end	no	
the back-end is trusted	yes	
the number of tag validations is limited within an epoch	yes	
scalability issues: linear complexity, $O(Vn)$	yes	

in the system). Similar issues has ALM due to Avoine *et al.* [97] which is based on TanSL. In ALM, however, terminal updates are foreseen (unlike in TanSL) that are targeted at tackling the problem of terminal compromisation. Moreover, ALM is a semi-coupled protocol. Garcia and Rossum (GR) further addressed the issue of terminal compromisation in their solution [6] (also a semi-coupled approach). Unlike TanSL and ALM, GR is a stateful protocol. Therefore, mechanisms for state re-synchronization after a possible attack from a compromised reader is presented in GR as well. One of the important constituents of GR is tag search in a terminal-side table. It is based on OSK protocol (see Section 4.2.2.1) with minor modifications (that is, inclusion of the reader nonce). The negative consequence of it is the impact on system scalability and terminal efficiency, since the worst case complexity for tag validation is $O(Vn)$ where n is the overall number of tags registered in the system and V is the maximum length of the hash chain (used for OSK tag search). All protocols reviewed in the current section are based on the fact that (authenticated and legitimate) terminals as well as the back-end are fully trusted. Moreover, each terminal *fully identifies* the tag on validation. Therefore, in each solution, only the attacker of type 1 is addressed (see Section 3.6). Mutual authentication between a tag and a terminal is explicitly considered only in TanSL 1 and ALM.

4.3.2.3 Solutions Based on Asymmetric Cryptography

As it was previously mentioned in Section 4.2.2.3, among the solutions applying the mechanisms based on symmetric cryptography which is well-established in the area of RFID authentication, there are approaches leveraging the potential of asymmetric cryptography. The latter require RFID devices with relatively profound performance capabilities. Their price, however, is no longer prohibitive for deployment in large-scale systems like ESPT which motivates to include the approaches making use of asymmetric cryptography into our review.

Pay as You Go (PAYG) by Baldimtsi et al. [9]

Similarly to HCDF [8] discussed in Section 4.2.2.3, Baldimtsi *et al.* [9] developed a solution essentially based on e-cash which is referred to ”Pay as You Go” (PAYG) in this dissertation. As HCDF, PAYG is specifically targeted at ESPT. PAYG, however, does not require terminals to maintain a constant connection to the back-end for e-ticket validation.

There are four main parties in PAYG: *(1)* e-ticket, *(2)* terminals, *(3)* vending machines, and finally *(4)* back-end. In PAYG, only single trip tickets are considered. The ESPT adheres to check-in/out model (CICO, see Figure 2.3) and the actual fare for the trip is calculated by the check-out terminal based on the location where an e-ticket was checked in. Similarly to HCDF, PAYG can be regarded as a privacy-preserving framework for ESPT that covers the following core processes:

1. E-ticket acquisition (namely, acquisition of e-cash based tokens);
2. Entering the ESPT (check-in).
3. Leaving the ESPT (check-out).
4. Obtaining a refund (reimbursement).

The first three processes are generic for a CICO-based ESPT. The last one (obtaining a refund) is specific to PAYG since it is a refund-based system. That is, on each check-in, a fixed amount of e-cash based tokens is spent (independently of the route) which corresponds to the most expensive (e.g., the longest) trip. On spending the tokens, the so-called refund calculation token[1] (RCT) is obtained from the check-in terminal which is used by the check-out terminal to issue a refund. The aforementioned core processes including the refund mechanism are explained in more detail below. The underlying cryptographic mechanisms can be found in [98].

1. E-ticket acquisition. Parties involved: *(a) e-ticket, (b) vending machine, (c) back-end.* E-ticket acquisition essentially implies a user acquiring e-cash based tokens from a vending machine. The prerequisite is that a user has the respective account opened either in some external bank or directly in TA (following the concept of e-cash). More specifically, the vending machine issues a travel credential referred to as Trip Authorization Token (TAT) to a user. TAT is based on e-cash and therefore an identifying piece of information pertaining to the user (e.g., the number of a credit card used for a transaction) is encoded into TAT (to prevent double-spending). TAT is therefore bound to a particular ID (that is, the ID is encoded into TAT) but in such a way that the ID cannot be directly obtained from a token. The cost of a single TAT corresponds to the most expensive single trip in the system. The actual price together with the corresponding refund will be determined later during check-out (see further). Therefore, PAYG is an e-ticketing system with upfront payment. Following the e-cash principle, the events of TAT withdrawal and its future use cannot be linked. The acquired TATs are stored at the e-ticketing medium of a user which is referred to as an e-ticket (in this case, acting like an electronic purse).

2. Entering the ESPT (check-in). Parties involved: *(1) e-ticket, (2) terminal.* To check in, the user spends one TAT maintained by the e-ticket at the corresponding terminal. On spending, an e-ticket has to prove (in zero-knowledge) that it knows the ID encoded into the TAT. If that is the case, the e-ticket is issued the so-called Refund Calculation Token (RCT) which corresponds to the entrance cookie in HCDF. RCT is bound to the TAT having been spent by the e-ticket and additionally contains a time stamp together with the terminal ID.

[1] The concept of RCT is similar to entrance cookie in HCDF.

RCT is signed by the corresponding terminal. After RCT has been issued, the user is allowed to enter the system. The aforementioned operations are performed locally by the terminal without the need to consult the back-end (that is, in an offline fashion).

3. Leaving the ESPT (check-out). Parties involved: *(1) e-ticket, (2) terminal.*
On check-out, the RCT together with the corresponding zero-knowledge proof of the encoded ID is presented to the terminal which calculates the actual fare (based on RCT) and the respective refund. The latter is aggregated into the so-called Refund Token (RT) maintained at the e-ticket side (refund top-up). These operations are also performed in an off-line fashion without the involvement of the back-end.

4. Obtaining a refund (reimbursement). Parties involved: *(a) e-ticket, (b) vending machine, (c) back-end.*
Refund is obtained from a vending machine by presenting the corresponding RT to it. The vending machine consults the central database in the back-end in order to check if the same RT has been already reimbursed and then issues a refund. The latter can be either cashed directly or used to obtain new TATs.

In Table 4.15, the main notations used during the discussion of PAYG are summarized for clarity.

Table 4.15: Notations used during the discussion of PAYG.

Notation	Meaning
TAT	Ticket Authorization Token (obtained from a vending machine);
ID	a piece of information uniquely identifying the user (e.g., a credit card number);
RCT	Refund Calculation Token (acts as a stamped ticket);
RT	Refund Token, aggregates multiple RCTs.

PAYG Assessment. PAYG is a framework targeted specifically at privacy-preserving ESPT. As in HCDF, its privacy-preserving properties are essentially inherited from e-cash. Namely, the unlinkability of tokens withdrawal and spending is guaranteed provided that no double spending occurs. E-tickets, therefore, are untraceable and unidentifiable not only towards terminals but also towards the back-end (as long as no double spending occurs). PAYG thus considers all three attacker types (see Section 3.6) which is an essential advantage over TanSL, ALM, and GR. Moreover, in contrast to HCDF, variable pricing schemes are supported through the encoding of the respective attributes (for example, a student, elderly, or disabled) into TAT. However, the system supports only single trip tickets. In order to obtain such a ticket, a (personalized) account must be opened beforehand to enable TAT issuance. Alternatively, the tokens could be bought without preliminary registration if for that a credit card is used, for example (which ID is then encoded into TAT for double spending prevention). This means that there is no possibility to obtain a single trip ticket in a fully anonymous way without the need for preliminary identification of a user. Whereas for a monthly or a year ticket it would be a viable option (which is not possible with PAYG), the sporadic travelers willing to use the system only once may be reluctant to perform an identifying transaction to obtain a single trip ticket. Similarly to HCDF, regular billing is by design not feasible in PAYG. Another issue is that mutual authentication between terminals and e-tickets is not directly considered in PAYG. This fact does not pose a serious privacy threat due to unlinkability properties (inherited from the e-cash concept). However, without mutual authentication an e-ticket can be covertly queried by an attacker (e.g. using a pocket reader) in order to illegitimately obtain the tokens. The adversary would not be able to use the received TAT further since it does not possess the knowledge of the ID encoded into the token (required for the corresponding zero-knowledge proof). However, as a DoS attack such a process of getting tokens from e-ticket is possible, since an e-ticket does not check the correctness of the terminal's answer (that is, the correctness of RCT). Moreover, all reimbursed refund tokens (RT) must be stored in the back-end to prevent double spending (merely for an infinite amount of time). Taking into account that refunds can be claimed after each ride (without accumulating individual refunds into RT), the amount of the redeemed tokens which must be stored and processed in the back-end on each reimbursement request may introduce a bottleneck. Furthermore, the refund-based approach itself may be seen as additional nuisance for users as it is necessary to spend extra time on reimbursement procedure. An important issue is the price calculation which happens at the terminal side during check-out. Terminals must be able to calculate the price on-the-fly based on the presented RCT. This renders flexible pricing schemes which require more complex processing time-prohibitive and imposes extra requirements on terminal performance. The authors did not include the time required for price calculation during PAYG evaluation in [9]. The main properties of PAYG are summarized in Table 4.16.

Table 4.16: Main properties of PAYG [9] summarized. Advantages are marked with a ✓sign.

Criterion	**Assessment**	
tight coupling between terminals and the back-end	no	✓
anonymity and untraceability against terminals	yes	✓
mutual authentication in the front-end	no	
fine-grained billing is feasible	no	
terminals are trusted	no	✓
the back-end is trusted	no	✓
efficiency considerations due to ZKP during check-in/out	yes	

Sadeghi, Visconti, and Wachsmann (SVW) [10]

A privacy-preserving framework for ESPT developed by Sadeghi, Visconti, and Wachsmann [10] referred to as SVW within this dissertation, is essentially based on secure key storage with physically unclonable functions (PUFs), symmetric key authentication, and re-randomizable encryption. In order to capture the core processes of an e-ticketing system for public transport, the following parties are considered in the framework: *(1)* token issuer, *(2)* verifiers, and *(3)* user devices managing tokens. Verifiers are further divided into *(a)* entrance verifiers, *(b)* inspectors, and *(c)* exit verifiers essentially depicting the process of a user beginning a ride by entering ESPT, presenting a ticket to a conductor if requested, and finally exiting the system. E-tickets in this case are represented by digital tokens the authenticity and validity of which must be proven to verifiers in order to use the system. Even though such system architecture *per se* would allow to implement CICO-based fare calculation on top of it (see Figure 2.3), in SVW framework, the support for flexible fare schemes and fine-grained billing was not explicitly considered.

SVW framework relies in large on token-based authentication between user devices (in this case, RFID tags) and verifiers. A token T is constructed by using an authentication secret K_T. In order to securely store the latter, the utilization of physically unclonable functions[1] (PUFs) was suggested in [10]. In order to bootstrap symmetric-key authentication between verifiers (terminals) and an e-ticket, the so-called re-randomizable public key encryption is employed (see [101, 102] for details). In essence, the idea is to additionally encrypt the authentication key of a token K_T under some public key pk_V, namely $c_T \leftarrow \text{Enc}_{pk_V}(K_T)$. The corresponding private key must be known to *all* verifiers (terminals) in the system. The respective ciphertext c_T is stored in the memory of an e-ticket and presented to a verifier on each authentication. Thus, the symmetric authentication key of a token K_T can be securely transferred to legitimate verifiers for authentication. The external entities without the knowledge of the respective private key to decrypt c_T (e.g., an observing attacker) are not able to extract K_T and subsequently misuse is. However, if the value of c_T is static and persists across multiple sessions, it can serve as a tracking parameter for external entities which perform sniffing. In order to mitigate that, the re-randomizable encryption comes into play. Namely, c_T can be re-randomized by an additional process called anonymizer. In SVW framework, the latter can be further classified into four categories:

1. *Integrated anonymizers.* The re-randomization process is performed directly by e-tickets if the respective computational capabilities are present.
2. *User-controlled anonymizers.* This kind of anonymizers is represented by an external personal user device capable of performing re-randomizable encryption and communication with an e-ticket carrier medium to transfer the computation result.
3. *Transport authority anonymizers.* The devices for re-randomization are deployed by a transport authority and are installed at stations or possibly been included into trusted verifiers.
4. *Public anonymizers.* Re-randomization can be performed by any device. That implies that every (possibly malicious) third party may host such an anonymizer which renders this type of anonymizers impractical due to security concerns.

The following core processes can be distinguished in SVW framework: *(1)* token acquisition, *(2)* token verification, and *(3)* token anonymization.

1. Token acquisition. Parties involved: *(a) issuer, (b) user device.*
On user request, the token issuer creates a token incorporating e-ticket parameters ρ_T (ticket type, validity period, etc.), token symmetric authentication key K_T as well as the corresponding issuer's signature σ_T on (ρ_T, K_T). Then the issuer creates the ciphertext c_T which is going

[1] PUFs are functions which are closely bound to physical properties of each concrete device and essentially originate from the unique random physical irregularities introduced during the manufacturing process. Therefore, PUFs are believed to be unclonable. For further information, see [99], [100].

to be sent to verifiers (terminals) for authentication: $c_T \leftarrow \mathrm{Enc}_{pk_V}(K_T, \rho_T, \sigma_T)$. In general, a public key of the anonymizer pk_A chosen by a user is delivered to the issuer as well. The latter in turn creates the anonymizer authentication key K_A and encrypts it under pk_A obtaining $c_A \leftarrow \mathrm{Enc}_{pk_A}(K_A)$. Finally, (K_T, K_A, c_T, c_A) is written to the user device by the issuer. The notations used are summarized in Table 4.17 for clarity.

Table 4.17: Notations used during the discussion of SVW.

Notation	Meaning
ρ_T	e-ticket parameters (ticket type, validity period, etc.);
K_T	token symmetric authentication key;
σ_T	issuer's signature on (ρ_T, K_T);
pk_A	public key of an anonymizer;
pk_V	public key of verifiers (terminals);
c_T	ciphertext for authentication between an e-ticket an a terminal;
c_A	ciphertext for authentication between an e-ticket and its anonymizer;

2. Token verification. Parties involved: *(a) user device, (b) verifiers.*
In order to verify the token, a terminal sends a verification request to a user device. The latter answers with c_T. The terminal in turn decrypts it using its secret key and verifies the extracted token signature σ_T and the validity of the corresponding travel attributes ρ_T. If both checks pass, the terminal engages into symmetric key authentication with the user device using the extracted K_T. If the authentication is successfully passed, the e-ticket is accepted.

3. Token anonymization. Parties involved: *(a) user device, (b) anonymizer.*
The anonymization protocol is performed between a user device maintaining the token (e-ticket) and the corresponding anonymizer (see different anonymizer types discussed above). First, the anonymizer authenticates itself to be allowed to perform re-randomization of credentials. For this, a user device generates a challenge N_T and sends it together with c_T and c_A to the anonymizer. The latter decryptes c_A using its secret key and obtains the anonymizer authentication key K_A created during the issuance phase (see token acquisition). Then the anonymizer re-randomizes (c_T, c_A) obtaining (c'_T, c'_A) respectively. Finally, with the extracted symmetric key K_A, the anonymizer authenticates newly re-randomized credentials together with the challenge N_T and sends everything to the user device. The latter checks the authenticity of received data and in case the check has passed, updates its credentials with newly received values (c'_T, c'_A). Depending on the type of anonymizer, re-randomization may be performed before each verification or as often as it is currently possible/convenient for the user (for example, in case of transport authority anonymizers).

SVW Assessment. Similarly to PAYG [9], SVW is explicitly targeted at privacy preservation in ESPT, likewise suggesting a framework capturing the core processes taking place. The main goal of SVW is to provide privacy protection (in the first place concerning user whereabouts) against the entities which are exogenous (i.e. external) to the e-ticketing system. That is, the verifiers (terminals) are considered to be trusted as long as they can successfully authenticate themselves to a user device. The authors of SVW mention that in order to reduce trust in terminals, the latter could maintain a real-time connection to the trusted issuer (the issuer is always trusted within the scope of SVW) or verifiers should be additionally equipped with a security module of the issuer (e.g., a TPM). Both of these options are, however, assessed as rather impractical by Sadeghi *et al.* Moreover, the necessity of maintaining a costly real-time time connection to the issuer would imply a considerable load increase at the issuer side which usually only issues credentials and is not supposed to constantly verify them in the running system. Furthermore, the system architecture would not be adhering to the decoupled principle any longer and consequently would lose all the respective advantages (see the introductory discussion in Section 4.3.1).

In SVW framework, the focus is made on ticket-to-terminal authentication. Terminals, however, are not explicitly authenticated to the e-tickets (whereas the anonymizers are). Therefore, mutual authentication is not explicitly considered. Moreover, should no anonymizer service be available for a certain time period, the e-ticket is traceable by *all* entities sniffing communication including the external ones. The reason for this is that c_T released from an e-ticket on each verification query remains static between re-randomization sessions and therefore can be used as a tracking parameter. Therefore, only the first type of attacker (the outsider, see Section 3.6) is considered by SVW and only in case re-randomizations take place regularly enough (that is, for each verification request c_T is freshly re-randomized). Moreover, no support for fine-grained billing is explicitly considered within SVW framework. Implementing this functionality on top is, however, in principle possible. The main properties of SVW are summarized in Table 4.18.

Table 4.18: A summary of SVW framwork [10]. Advantages are marked with a ✓sign.

Criterion	**Assessment**	
tight coupling between terminals and the back-end	no	✓
anonymity and untraceability against terminals	no	
mutual authentication in the front-end	no	
fine-grained billing is feasible	yes (as add-on)	✓
terminals are trusted	yes	
the back-end is trusted	yes	

4.3.2.4 Solutions Based on Asymmetric Cryptography: A Short Summary

Within this section, two approaches were reviewed which leverage asymmetric cryptography and are based on the decoupled system architecture. Both frameworks, PAYG due to Baldimtsi *et al.* [9] and SVW by Sadeghi *et al.* [10] are explicitly targeted at public transport scenario. PAYG achieve relatively profound privacy properties inherited from the e-cash concept. More specifically, event the token issuer can not link token transactions in the front-end (provided that no double spending occurs). Neither can terminals and system back-end trace e-tickets. Unlike PAYG, in SVW framework, the authorized terminals as well as the back-end are considered to be fully trusted. Moreover, the e-tickets are traceable between two consecutive re-randomizations involving the anonymizer process. At the same time, the processing power requirements with respect to user devices maintaining e-tickets are much lower in SVW than in PAYG (in case the anonymizer service is running on an extra device). Moreover, SVW framework does not prohibit the implementation of fine-grained billing policies whereas PAYG allows only relatively limited options for that (in PAYG, the price is determined on the fly during check-out).

4.3.3 Decoupled Systems: Summary

The approaches based on the decoupled architecture have an important practical advantage following from the fact that terminals can locally validate e-tickets without consulting the back-end in real-time. In case of TanSL [96] and ALM [97] this is achieved by uploading terminal-specific access lists pre-calculated in the back-end to the terminals. In GR [6], an alternative solution was presented which is based on regular updates of terminal secrets and does not require the calculation of terminal-specific access lists in the back-end. Similarly to TanSL and ALM, GR does not provide anonymity and untraceability of e-tickets against authorized terminals. Moreover, it is based on similar trust assumptions and does not prohibit fine-grained billing. However, unlike TanSL and ALM, no mutual authentication is provided in GR. The SVW framework presented in [10] specifically address the public transport domain. As in case of the aforementioned approaches, SWV is based on the assumption that authenticated terminals as well as the back-end are trusted. Therefore, all of the approaches above only consider the first type of attacker discussed in Section 3.6 (namely, external observers). The PAYG framework [9], however, covers this issue and holistically addresses all attacker types due to its decent privacy-properties essentially inherited from the e-cash concept. However, the main limitation of PAYG is that it by design prohibits fine-grained billing. Summarizing, it can be concluded that none of the solutions reviewed in this section fully satisfied all core requirements discussed in Section 3.5.

4.3.4 Approaches Leveraging Decentralized Billing with Client Involvement

As it was discussed in Section 3.5.2, privacy-preserving approaches involving fine-grained billing can be roughly divided into two categories: *(1)* centralized (performed in the back-end) and *(2)* decentralized (involving client-side processing). Due to the reasons presented in Section 3.5.2, the main focus of the dissertation is made on the approaches falling into the first category. In case of the decentralized billing, it is commonly assumed that the necessary statistics (e.g., time, check-in/out location, etc.) including a portion of protocol transcripts from each check-in/out session is maintained at the user side. Based on it, the trusted user equipment performs billing itself by carrying out all necessary calculations and preparing (cryptographic) proofs of their correctness and accordance to the agreed fare policies. At the end of each billing period (e.g., once a month) the calculated bill together with the respective proof (possibly including signed commitments to the calculated values and pieces of detailed statistics) is sent to the transport authority. The latter checks the corresponding signatures and may challenge the user side with specific requests to prove the correctness of the calculated bill for a certain time period. This may entail requests to open certain commitments and to provide respective proofs that the corresponding bill parts were calculated correctly for that specific time period (for which the transport authority now knows the detailed pieces of statistics). If all aforementioned checks successfully pass, the transport authority can be reasonably sure that the bill calculated at the client side is correct. At the same time, the transport authority does not gain enough information to invade customer privacy (since it knows only an overall bill and a few pieces of detailed statistics).

The decentralized billing approach is often used for privacy preservation in the domain of toll collection, see e.g., [71, 103, 104], as well as in the area of smart metering, see [105, 106]. Even though currently there exist no similar approaches known to us which explicitly target public transport based on e-ticketing, it should be possible to apply the aforementioned concept for this domain as well. However, as already discussed in Section 3.5.2, the decentralized billing generally entails a substantial additional overhead for a user compared to more traditional and well established centralized billing.

4.4 Related Work: Summary

Each of the existing approaches reviewed above was assessed in detail in the corresponding section. The most important ones are summarized in Table 4.19 for clarity and comparison purpose. The core requirements which played the decisive role during the evaluation are presented first and marked with gray in Table 4.19. As in the case of individual summaries for each approach reviewed, check marks (✓) indicate advantages (where applicable). Further evaluation criteria are provided as well in Table 4.19 for completeness.

As it immediately follows from Table 4.19, none of the current solutions fully satisfies the set of core requirements presented in Section 3.5. Therefore, our own approach discussed in the next chapter addresses this issue and closes the gap.

Table 4.19: Related work: a general summary. The core criteria are highlighted with gray.

Criteria		**The most relevant approaches reviewed**						
		PAYG [9]	HCDF [8]	SVW [10]	GR [6]	ALM [97]	OSK [80]	RSMP [91]
Tight coupling		no ✓	yes	no ✓	no ✓	no ✓	yes	yes
Anonymity against term.		yes ✓	yes ✓	no	no	no	yes ✓	yes ✓
Untraceab. against term.		yes ✓	yes ✓	no	no	no	yes ✓	yes ✓
Mutual authentication		no	no	no	no	yes ✓	no	yes ✓
Fine-grained billing possible		no	no	yes ✓	yes ✓	yes ✓	yes ✓	yes ✓
Terminals are trusted		no ✓	no ✓	yes	yes	yes	no ✓	no ✓
The back-end is trusted		no ✓	no ✓	yes	yes	yes	yes	yes
Explicitly considers ESPT		yes ✓	yes ✓	yes ✓	yes ✓	no	no	no
Revocation is possible		yes ✓	yes ✓	yes ✓	yes ✓	yes ✓	yes ✓	yes ✓
Dynamic extensibility		yes ✓	yes ✓	yes ✓	no	no	yes ✓	no
Tamper resist. required		not consid.	not consid.	partly	not consid.	not consid.	no ✓	no ✓
Involves external device		no ✓	partly	yes	no ✓	no ✓	no ✓	no ✓
Crypto primitives	Symmetric	no	yes	yes	yes	yes	no	yes
	Hash	yes	yes	no	yes	no	yes	yes
	Asymmetric	yes	yes	yes	no	no	no	no

Chapter Summary

In this chapter, a detailed analysis of the related work was performed. In order to provide for a systematic review, the evaluation criteria were discussed in Section 4.1. Based on the elaborated set of criteria, the approaches under review were generally classified into the ones based on tight coupling between the front-end and the back-end (discussed in Section 4.2) and the ones based on the decoupled architecture (reviewed in Section 4.3). The advantages and shortcomings of each architecture type were discussed at the beginning of the respective sections. Finally, after the detailed review and assessment of each approach, the most important ones were summarized in a tabular form in Section 4.4. Having reviewed the most relevant solutions, it can be concluded that none of them fully satisfies the whole set of core evaluation criteria and consequently the core requirements discussed in Chapter 3 (see Section 3.5). Therefore, there is a need for new solutions covering this issue. This is addressed in the next chapter where our approach is presented and discussed.

5 Suggested Solution

Having performed an extensive state-of-the-art analysis in the previous chapter, the conclusion can be made that no solution developed so far satisfies all requirements for a privacy-preserving e-ticketing system for public transport (ESPT) and addresses all core evaluation criteria at the same time (see Section 4.1). In this chapter, a solution to this problem is presented and discussed in detail. Firstly, the main building blocks of our approach are outlined in Section 5.1. An overview of the developed privacy-preserving framework is provided in Section 5.2. The proposed solution is further elaborated in Section 5.3. The chapter is concluded by the discussion of approach limitations and estimation of integration costs into real-world systems in Section 5.4 and Section 5.5 respectively.

5.1 Solution Building Blocks

Our solution is essentially comprised of three core building blocks:

1. *Mutual authentication* between an e-ticket and a terminal (system front-end). In this case, the front-end of ESPT is in focus, since the backbone communication can be secured using fully-fledged, well-established standard mechanisms (see the discussion in Section 3.5.1).

2. *Local revocation* of invalid e-tickets at the terminal side without the need to consult the back-end database in a timely fashion (system front-end), supports loose-coupling.

3. *Path reconstruction.* Correlating different travel rides to a user pseudonym for billing purposes (system back-end).

All of the three aforementioned building blocks (depicted in Figure 5.1) must be implemented in a privacy-preserving way corresponding to the attacker model defined in Section 3.6 and being conform to the requirements discussed in Section 3.5. As indicated in Figure 5.1, there are important interdependencies between the building blocks. Privacy-preserving authentication should not prohibit local revocation at the terminal side as well as path reconstruction in the back-end. Similarly, local revocation (with the desired privacy properties) should not prevent path reconstruction. The aforementioned interdependencies introduce a serious challenge for system design, since such properties as non-identifiability and untraceability on the one hand and user revocation together with path reconstruction on the other hand are inherently contradicting. Our approach outlined in Section 5.2 and elaborated in Section 5.3 demonstrates how the aforementioned challenges can be solved. At first, however, each building block is discussed in more detail together with the respective challenges.

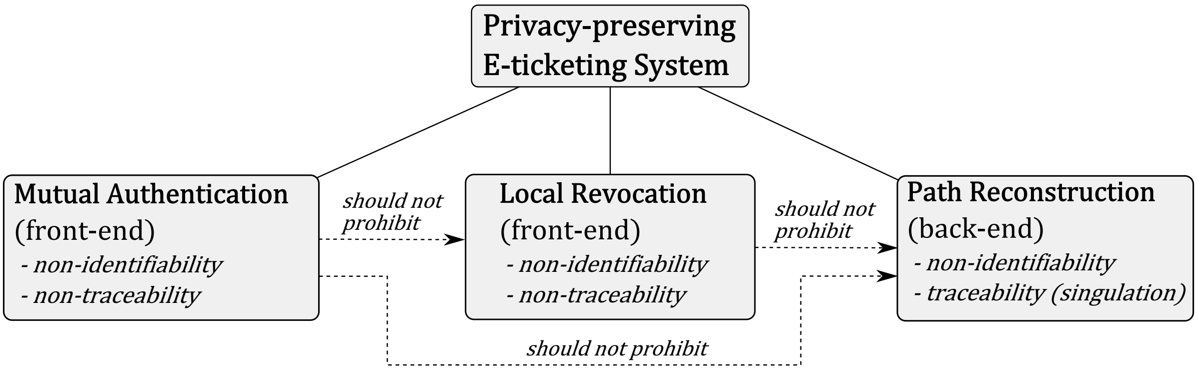

Privacy-preserving E-ticketing System
Mutual Authentication (front-end)
- non-identifiability
- non-traceability
should not prohibit
Local Revocation (front-end)
- non-identifiability
- non-traceability
should not prohibit
Path Reconstruction (back-end)
- non-identifiability
- traceability (singulation)
should not prohibit

Figure 5.1: Main building blocks of our solution. Privacy properties such as non-identifiability and untraceability refer to an e-ticket (either against a terminal in the front-end or against the back-end system).

5.1.1 Mutual Authentication

Mutual authentication between an e-ticket and a terminal (during check-in/out events) is required for a number of reasons:

1) to ensure that an e-ticket can be queried only by a legitimate terminal (privacy);
2) a terminal should process the events triggered only by the legitimate e-tickets (correctness and integrity);
3) man-in-the-middle attacks must be prevented (even though such kind of attack is considered to be inapplicable to NFC/RFID in practice, see [107], for example).

On each check-in/out, mutual authentication should not leak any auxiliary information about the e-ticket which can be misused by terminals or exogenous entities for its *(a)* identification (the worst case), *(b)* tracking, or *(c)* linking different communication sessions with the terminal(s) together. Hereinafter, the term "identification" with respect to an e-ticket refers to the process carried out by some entity (being a part of a system as a terminal or an exogenous one) outputting a parameter uniquely identifying an e-ticket within the system (such as a serial number).

For the current e-ticketing scenario, the fundamental challenge is essentially bootstrapping the authentication process without a terminal being able to *(a)* distinguish between e-tickets, let alone *(b)* to identify them. Moreover, mutual authentication should not prohibit path reconstruction in the back-end, see Section 5.1.3.

There are several advanced cryptographic tools available which provide for privacy-preserving mutual authentication, such as the utilization of group signatures (for example, [108, 109]), explicitly leveraging various zero-knowledge techniques [110, 111] for proving the possession of a valid authentication parameter, etc. However, the aforementioned approaches introduce certain limitations[1] for the target use case scenario of ESPT and re-

[1] For example, in case of anonymous credentials due to Chaum [108], they are one-show credentials. If the same credential is presented to the same terminal more than once, the anonymity property can be defeated (by the terminal solving a linear system of equations and extracting the secret parameter).

quire additional assumptions for their successful deployment. With this respect, the following challenges should be mentioned:

1. *Dynamic extensibility.* The deployed mutual authentication scheme must support dynamic accommodation of new e-tickets joining the system.

2. *Bootstrapping authentication.* Mutual authentication should prevent tracking of e-tickets by terminals. At the same time, many existing approaches require some kind of a bootstrapping parameter for authentication which is often static (and therefore can serve as a tracking attribute).

3. *Efficiency.* Applying advanced mutual authentication methods with decent privacy properties often has negative efficiency implications and may even be prohibitive in case of constrained devices (used as a carrier medium for e-tickets, e.g., smartphones or smart cards).

Under the current circumstances, we suggest that such kind of mutual authentication is implemented using a specialized certificate-based approach. That is, a terminal provides its unique certificate, which is in turn signed by a transport authority (TA). An e-ticket possesses another type of certificate signed by TA and therefore can perform signing operations as well. Unlike the terminal's signature, the one of an e-ticket solely proves that the latter belongs to a valid *ticket group* (e.g., a monthly or a yearly ticket, possibly with certain attributes like a student or an elderly) and does not reveal any identifiable information on each particular e-ticket. The aforementioned challenge of bootstrapping the authentication process, therefore, can be solved by essentially checking the respective certificate chain. Unlike the conventional certificate-based authentication, the e-ticket cannot be traced and uniquely identified due to its group signature.

5.1.2 Local Revocation

Whereas the users possessing a valid travel permission must be granted access to public transport system, the holders of the invalid e-tickets (which have been blacklisted) must be rejected by a terminal. Therefore, terminals must be able to check if the e-ticket communicating with them has been blacklisted or not. An important condition here is similar to the one for mutual authentication (see Section 5.1.1): valid e-tickets must remain anonymous (to the terminal) and untraceable. This is, however, a fairly challenging task. One of the possible solutions would be to utilize the concept of cryptographic accumulators, see [112]. In this case, instead of checking if the current e-ticket has been blacklisted, it is proved (in zero-knowledge) that a certain committed value securely stored at the e-ticket side, has been included into the terminal-side whitelist, or an accumulator. The process of revocation then would essentially imply removing the specific value from the accumulator and recalculating it anew. The new accumulator must be redistributed to all terminals in the system, similarly to the blacklist. An important caveat is, however, that along with the terminal-side accumulators, the e-ticket credentials must be updated as well on each revocation (for the proof of membership to function properly). This requirement renders it highly impractical to apply the cryptographic accumulator concept to the public transport scenario where e-tickets are for the most time offline. Even in case the updated accumulator value were dynamically delivered to a user device during check-in/check-out, recalculating the credentials (which have to be performed on the fly) would introduce an additional computational overhead and hence add a costly time delay to the handling of time critical check-in/check-out events. Another solution to the problem of privacy-preserving revocation could be the concept of ”anonymous blacklists”. An example of this approach is an anonymous blacklisting scheme called ”Nymble” [113]. In this system, the multi-party concept is used to protect user privacy, namely along with the service provider and users there are two additional trusted parties: the so-called pseudonym manager and the nymble manager. In order to obtain an anonymous credential (consisting of ”nymbles”) for using the service, a user has to *(1)* contact the pseudonym manager at first to request a pseudonym which is *(2)* subsequently presented to the nymble manager who issues the so-called ”nymbles” to the user. Nymbles are essentially one-time credentials empowering the user to get the service. This approach is, however, hardly applicable to the target scenario of public transport. Firstly, additional interaction must be carried out by each user device (an e-ticket) on a regular basis to obtain credentials (which are the basis for blacklist check) from the respective trusted parties. In [113], it is recommended that new credentials are obtained every 24 hours. This may introduce additional nuisance for the users who would have to carry out the update procedure on such a regular basis (especially if an additional piece of equipment like RFID/NFC reader is required for this). Secondly, using one-time credentials requires some kind of a global register to keep track of the already used credentials in order to prevent double spending, etc. This is prohibitive considering a highly decentralized system architecture employing offline terminal-side validation. Thirdly, in order to get a credential, an anonymous connection must be established from a user device to the nymble manager. The implementation of such anonymous channel is far from straightforward especially in case of a smart card or a smartphone as a user device. Lastly, the user device must possess the clock to chose the right credential which is associated with the current time epoch. In case of a passive smart cards as a user device, this could be especially difficult.

Therefore, in this thesis another solution for privacy-preserving local revocation was developed, namely based on terminal-side blacklists which are regularly updated by the back-end. The detailed discussion on this follows in Section 5.3.3.

5.1.3 Path reconstruction

Lastly, in order to issue a bill, different rides must be correlated to each other in such a way that the underlying identity of each user remains unknown to the transport authority (referred to as path reconstruction in Figure 5.1). The following challenges should be considered in this case. On the one hand, the supported fare schemes which are applied during the billing phase need to be *flexible* and *extensible*. That is, they should not be hard-tailored to a specific fare collection approach let alone to the privacy-preserving mechanisms in use (as it was done, for example, in [103]). Moreover, it should be possible to combine the rides to issue discounts (consider the example of a ”short ride” ticket up to 4 stations). On the other hand, the process of fare calculation and subsequent billing must be carried out in a privacy-preserving way, that is without directly identifying the customer and leaking information about his/her travel habits. The aforementioned issues introduce a severe contradiction between fare scheme flexibility and user privacy (in terms of individual traceability). In case of a relatively simple fare scheme where the price for a ride between each two stations (that is, between two successive check-in/check-out events) can be represented in form of a fare matrix, it is already possible to implement billing with decent privacy properties, see [16], for example. However, using such matrix-based approach entails considerable disadvantages, namely *(1)* billing inefficiency (in case of [16], cubic complexity in the number of travel records processed), and *(2)* billing inflexibility, e.g., inability to combine (i.e., to link) several rides to issue a discount, etc.

Therefore, in our framework an approach based on pseudonymisation is chosen. More specifically, a reasonable trade-off is considered, namely whereas check-in/out events are completely unlinkable and untraceable in the front-end, the back-end is able to correlate different rides to a single pseudonym and to subsequently apply the deployed fare scheme for billing. The back-end still does not learn the underlying user identity, which is managed by the trusted third party in our framework (see Section 5.2). Such approach goes in line with the adopted attacker model (see Section 3.6). Our pseudonymisation scheme is elaborated in Section 5.3.2.

In what follows, our solution addressing all challenges outlined above is going to be outlined (Section 5.2) and presented in more detail (Section 5.3).

5.2 Solution Outline

Our privacy-preserving framework follows the principles of information minimization and task division between non-colluding entities. The solution essentially considers two distinct entities: *(1)* a transport authority (TA) and *(2)* an external trusted third party (TTP), see Figure 5.2. TA represents a public transport company which provides transport services. TTP acts as a trusted mediator between a transport authority (which maintains the public transport system) and its end users. The main idea behind this concept is the following. In order to issue a bill, the back-end of a TA does not necessarily have to possess identifying information about a particular user of an e-ticket. It merely needs to be able to correlate different rides performed by a customer to a certain pseudonym which has been negotiated with a TTP in advance and based on this information to apply the deployed pricing schemes with the subsequent billing procedure. The resultant bill together with the corresponding pseudonym is periodically sent to the TTP. The latter has no knowledge of travel history of a customer but is solely aware of the overall bill and the user behind the pseudonym. Individual payments are then forwarded to the TA by the TTP in an aggregated form. In addition, the rides history could be stored at the user device (e.g. a smartphone) so that it can be locally viewed by a customer later. Thus, the TA trusts the TTP that users are correctly billed (together with payment enforcement) while customers rely on the TTP to protect their privacy and forward payments to the TA. In essence, the approach outlined above creates a privacy overlay with respect to the sensitive data pertaining to users. Thus, the back-end of a TA (alternatively, of several TAs for interoperability) is operating on this abstract layer without directly processing the identifiable information of customers. An important condition here is that the TTP does not collude with the TA to defeat privacy protection provided by the solution.

The concept of task division between several entities has already been leveraged in practice. As an example, a service-based architecture for fare management and interoperability specified in the respective German e-ticketing standard (VDV-KA[1]) [36] can be considered. In this case,

[1] VDV stands for Verband Deutscher Verkehrsunternehmen, KA – for Kernapplikation (core application).

however, the main goal was in the first place to achieve better efficiency and loose-coupling between system components without explicitly considering the mechanisms for privacy protection. Moreover, each e-ticket is uniquely traceable in the front-end of VDV-KA system [114]. The concept of a trusted third party has been also leveraged in the Octopus e-ticketing system rolled out in Hong Kong. Namely, the TTP in this case performs key management for the providers of public transport services as well as for other market players which deliver various value added services and have met the respective agreements [115]. Thus, the aforementioned examples demonstrate that the additional effort for integrating our privacy-preserving solution into a real-world system is considerably lower than it can be expected at the first sight. The reason for this is that the concept of task division has been already leveraged in real systems meaning that the respective interfaces required for integrating a TTP are already present. What should additionally be done is rerouting the system flow via the established TTP. Further discussion with respect to the integration of our solution into a real system (with VDV-KA as an example) is provided in Section 5.5.

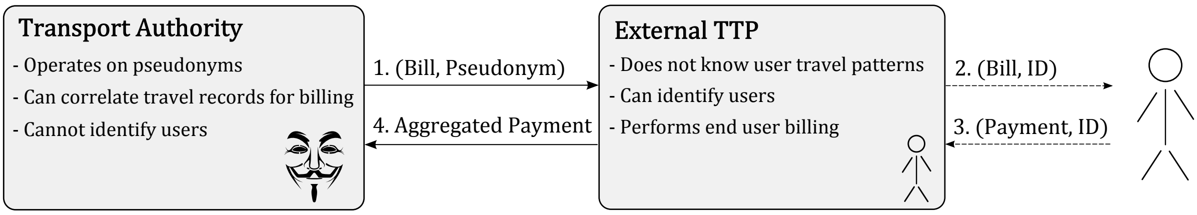


Figure 5.2: A privacy-preserving framework: an overview.

A Remark On Possible System Variations. In case the assumptions with respect to the attacker model allow, a TTP could be represented by a trusted part of ESPT operating in the secured area dedicated for such kind of processing with the respective enforcement of strict access control mechanisms, etc. In this scenario, the back-end of a transport authority would be able to perform user identification in its trusted part. In the front-end, however, all privacy properties would preserve their nature as in the case of a system architecture with an external TTP.

5.3 Suggested Solution: A Detailed Description

5.3.1 Basic Information Flow

5.3.1.1 Initialization

Before being able to actively use the transport system, each customer has to engage into an initialization phase with the TTP. This can be carried out in one of the following ways: *(1)* via a special issuing machine, *(2)* directly at the customer office of a transport authority, or *(3)* via the Internet. On registering the customer, the TTP creates the respective pseudonym P_i^T and forwards it to the TA. The latter further transforms the received pseudonym into its encrypted form P_i^A and operates on it (pseudonymization is explained in more detail in Section 5.3.2). The customer in turn gets the necessary credentials from the TTP to start using the system, namely: a TA public key k_{ta}^{+}, a key pair corresponding to the subscription group (e.g. a monthly or yearly pass) (k_{gr}^{+}, k_{gr}^{-}), as well as the specifically created customer pseudonym P_i^T (together with its TA-form P_i^A). Note that a public key is denoted as k^+ and a private one as k^-. A subscription group key can be constructed in accordance with the concept of group signatures, see [108], for example. Another more straightforward and sometimes more efficient approach is to use a conventional key pair (e.g. RSA) for each ticket group. The received key pair is signed by the TA and can be viewed as a kind of a digital certificate.

5.3.1.2 System in Action

Front-end (time-critical). On entering the public transport system, a user performs check-in at the entrance terminal. This involves three stages: *(1)* a secure session establishment, *(2)* mutual authentication, and *(3)* blacklist check (see Figure 5.3). The aforementioned processes must be handled in a timely fashion, since a check-in/check-out operation implies a customer holding his/her device near the terminal waiting to begin/end or to continue the travel. The secure session can be established either using an algorithm defined by the e-ticketing application itself (e.g., through the application-defined Diffie-Hellman key agreement) or alternatively by leveraging the standard techniques defined in ISO 7816-4 [60] or in NFC-SEC-01 [61]. Note that depending on the way the secure session has been established during the first stage, additional binding of the exchanged keying material (e.g., DH ephemeral keys) to the corresponding certificates may be required to prevent man-in-the-middle attacks. It has to be mentioned, however, that due to the physical properties of communication between terminals and e-tickets, man-in-the-middle attacks (in contrast to relay attacks) mounted on such wireless interface are extremely unlikely in practice [107]. The subsequent communication between an e-ticket and a terminal is, therefore, secured against an observing attacker (*attacker 1*, see Section 3.6). Afterwards, mutual authentication between an e-ticket and a terminal is performed as follows. The terminal has its unique public key k_T^+ signed by the back-end. The e-ticket uses its group key pair (k_{gr}^{+}, k_{gr}^{-}) which is signed by the TA back-end as well. Mutual authentication is then essentially performed according to the certificate-based challenge-response. Lastly, the terminal (locally) checks if the credentials of the current e-ticket have not been revoked by consulting the blacklist (see *BL Check* in Figure 5.3). This is performed in a privacy-preserving way (in contrast to the majority of conventional systems). That is, each e-ticket stays anonymous and untraceable against the terminal as long as it has not been included into a terminal-side black list (similarly to the notion of conditional anonymity defined in [116]). On successful check, the terminal creates the so-called travel record (TR) corresponding to the current check-in/out event. It usually contains a timestamp, location, and other pieces of information pertaining to the e-ticket (including its session pseudonym, see Section 5.3.2).

Back-end (non-real time). A set of travel records maintained by each terminal is regularly sent to the back-end system via the backbone network where they are processed for billing purposes (Figure 5.3, *Billing*). Terminal-side blacklists are regularly updated as well. The frequency of such updates is mainly determined by the connection type between terminals and the back-end (e.g., nightly updates as considered in [97] or more frequent updates if the connection allows). In the back-end, different travel records are sorted with respect to the e-ticket they pertain to. Hereinafter, this is referred to as singulation (see Figure 5.3). After the singulation phase, the respective billing policies are applied resulting in the overall bill for a certain customer pseudonym (pseudonymisation is explained in detail in the next section).

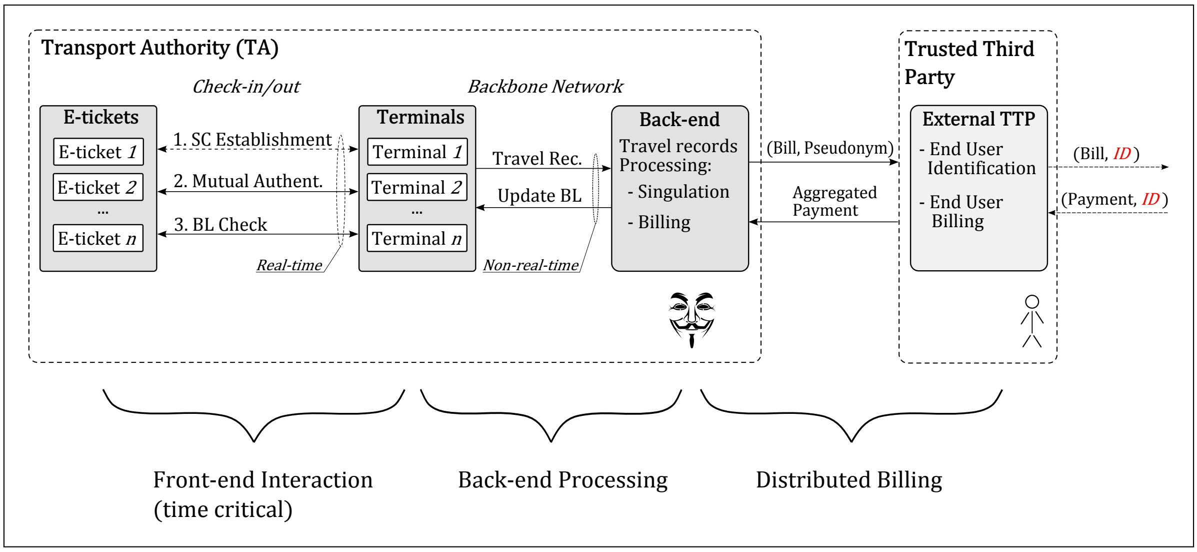


Figure 5.3: The suggested privacy-preserving framework. *BL* stands for blacklist, *SC* – for secure channel.

Having described the core information flow in the system, an elaboration with respect to the specific system constituents is provided below. As already mentioned in Section 5.1, there are strong dependencies between certain building blocks of our solution (see Figure 5.1). Since both, mutual authentication and local revocation must not prohibit path reconstruction in the back-end, the latter is going to be considered in the first place within the detailed discussion hereinafter.

5.3.2 Path Reconstruction Through Pseudonymisation

In what follows, our pseudonymisation scheme allowing for privacy-preserving path reconstruction (see Section 5.1.3) is discussed in detail.

During the initialization phase, a static pseudonym P_i^T is created by a TTP for each e-ticket ID. The mapping[1] between P_i^T and the respective e-ticket ID is kept secret at the TTP side. P_i^T is then sent to a transport authority (TA) to be included into the TA's pseudonym set P^T. TA, therefore, is only operating on pseudonyms and stays unaware of the underlying e-ticket ID. In order to further separate the processes of end user billing (performed by TTP) and TA-internal processes including bill calculation, P_i^T is transformed into a TA-specific pseudonym: $P_i^A \xleftarrow{trans} P_i^T$ (the notations are summarized in Table 5.1). This transformation is performed in such a way, that a TTP even having gained access to several records containing

[1] One of the ways to implement such mapping is to probabilistically encrypt the e-ticket ID (for semantic security) with the private key of TTP and to keep the latter secret.

TA-specific pseudonyms, would neither be able to *(1)* restore the underlying P_i^T *(2)* nor to distinguish between records pertaining to different e-tickets. Such properties are required to further enforce the "separation of concerns" between TTP and TA, namely to make sure that the TTP does not gain additional information (it does not require to operate) concerning the history of rides. The transformation can be carried out as follows: $P_i^A = E_{k_{ta}^+}(P_i^T,\ s_i)$, where $E_{k_{ta}^+}$ denotes encryption under the TA public key and s_i is a random value (salt). In order to be able to restore P_i^T, the encrypted salt value is stored together with P_i^A in the back-end:

$$P_i^T \mapsto \left(P_i^A,\ E_{k_{ta}^+}(s_i)\right) \tag{5.1}$$

In order to prevent terminals from tracking e-tickets (covering attacker type *2*, see Section 3.6), a session pseudonym SP_j is created at the e-ticket side on each interaction with a terminal:

$$SP_j = E_{k_{ta}^+}\left(P_i^A \cdot r_j\right), \tag{5.2}$$

where r_j is nonce number generated by the e-ticket. Since a terminal is not in the possession of a TA's decryption key (k_{ta}^-), it is infeasible for it to tell if two session pseudonyms obtained from different check-in/out events pertain to the same e-ticket or not. Neither can the terminal gain any knowledge from interaction with an e-ticket about the static pseudonym (P_i^A) of the latter. Thus, for each particular e-ticket, travel records created by terminals on check-it/out contain different session pseudonyms SP_j (see Figure 5.4). In order to enable bill calculation in the back-end part of the system, the pseudonym singulation step is required to correlate different session pseudonyms $\{SP_j\}$ with the respective static one P_i^A using the private key of the TA k_{ta}^-. Afterwards the billing process is carried out on static pseudonyms $\left\{P_i^A\right\}$ which are finally decrypted to the initial TTP pseudonyms $\left\{P_i^T\right\}$. The result of the billing step is a set of tuples $\left(bill, P_i^T\right)$ which is regularly (e.g., monthly) sent to the TTP for end user billing. The pseudonymisation scheme described above is depicted in Figure 5.4 for clarity.

Table 5.1: Pseudonymisation: notation used.

Notation	**Meaning**
P_i^T	a static pseudonym created by TTP;
P_i^A	a static pseudonym created by TTP from P_i^T;
SP_j	a session pseudonym (randomized P_i^A).

5.3.3 Privacy-preserving Local Revocation Based on Blacklists

5.3.3.1 Basic Scheme

In order to provide the transport service only to the legitimate customers, black list check is performed during check-in/check-out (see Section 5.3.1.2). More specifically, we resort to a custom and relatively simple yet privacy-preserving blacklisting scheme. It is based on (inherently) homomorphic properties of the underlying encryption scheme in use. Namely, the following property is exploited:

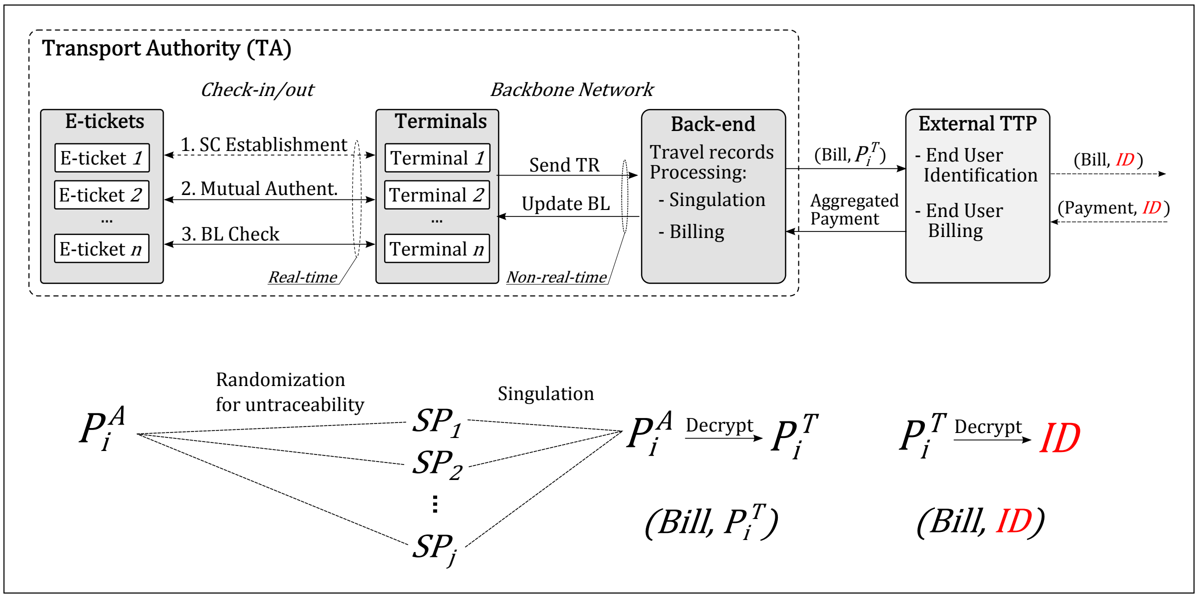


Figure 5.4: The employed pseudonymisation scheme and privacy-preserving framework.

$$E(x \cdot r) = E(x)^r, \tag{5.3}$$

where for clarity and conciseness x represents the TA-side pseudonym P_i^A (see Table 5.1), r is a nonce value as given in Equation (5.2). Therefore, $E(x \cdot r)$ corresponds to the session pseudonym SP_j. The notations used in this section together with the respective associations are summarized in Table 5.2.

Table 5.2: Blacklist check: notations used.

Notation	Meaning/Association
x	an e-ticket static pseudonym, P_i^A;
y	a blacklisted x;
$BL : \{y\}$	a blacklist (a set of y);
r	a random nonce;
$E_{k_{ta}^+}(x \cdot r)$	a session pseudonym, SP_j;
$\left(E_{k_{ta}^+}(x \cdot r), E_{k_{ta}^+}(r)\right)$	a session pseudonym tuple (SPT).

The terminal-side blacklist (BL) contains a set of blacklisted static pseudonyms[1],

$$\{y : y \in BL\} \tag{5.4}$$

[1] That is, the subset of pseudonyms created by the TA from the TTP ones, see Equation (5.1).

which are checked against during the e-ticket verification procedure. After mutual authentication (see Section 5.3.5), an e-ticket presents its Session Pseudonym $E_{k_{ta}^{+}}(x \cdot r)$ to a terminal along with the encrypted nonce value $E_{k_{ta}^{+}}(r)$ used for masking. Session pseudonym and the encrypted nonce form the so-called *Session Pseudonym Tuple (SPT)*:

$$SPT \leftarrow \big(E_{k_{ta}^{+}}(x \cdot r),\ E_{k_{ta}^{+}}(r)\big). \tag{5.5}$$

Having obtained this tuple, the terminal can use the homomorphic property (5.3) to perform blacklist check:

$$x \overset{?}{\in} BL : \{y\}\,.$$

For this, it creates an auxiliary temporary check set C and computes its elements as follows:

$$\forall y \in BL, E_{k_{ta}^{+}}(r) \in SPT\ :\ c \leftarrow E_{k_{ta}^{+}}(r)^{y}. \tag{5.6}$$

Then a terminal pairwise compares the computed c elements with the delivered Session Pseudonym:

$$c \overset{?}{=} E_{k_{ta}^{+}}(x \cdot r)\ \ \forall c \in C. \tag{5.7}$$

Note that for performance reasons, the comparison operation expressed in Equation (5.7) can be performed element-wise immediately after each new element of C is calculated within the Equation (5.6). If a match is found, the e-ticket is in the blacklist set BL and must be rejected. The developed local revocation scheme is presented in summarized in Algorithm 1 for clarity.

Therefore, due to the randomized nature of session pseudonyms, terminals are prevented from tracking valid e-tickets (see the terminal's view on check-in/check-out transaction in Figure 5.4). Should an e-ticket be on the blacklist, however, the algorithm would find a match (see Equation (5.7)) and the user would be prohibited from entering the public transport network. The visual representation of black list check procedure is provided in Figure 5.5.

Choosing an Appropriate Encryption Scheme

As an example of an encryption function possessing the homomorphic property (5.3), the scheme based on the intractability of the Discrete Logarithm Problem[1] (DLP) can be used. Thus, $\forall x \in \mathbb{G}_q : \mathbb{G}_q \subseteq \mathbb{Z}_n^*$ (with n, q classically being large primes, $q|n-1$) the encryption can be written as:

$$E(x) = g^{x} \qquad (mod\ \ n). \tag{5.8}$$

A session pseudonym (see Equation (5.2)), therefore, can be expressed as:

$$SP_j \leftarrow g^{x \cdot r_j} \qquad (mod\ \ n), \tag{5.9}$$

where x is an e-ticket pseudonym, r_j is a session nonce generated to mask x, and $r_j, x \in \mathbb{G}_q$.

[1] DLP follows from the hardness to extract x out of g^x in $\mathbb{Z}_n^*$. See, for example, [117] for further reference.

Algorithm 1: Terminal-side local revocation based on blacklists.

```
Input:
        $BL : \{y\}$ ;                                     // a terminal-side blacklist set
        $SPT_i \leftarrow \big(SP_i, : \ E_{k_{ta}^+}(r_i)\big)$;   // an SP tuple for session $i$
Output: accept/reject the e-ticket
begin
    extract $E_{k_{ta}^+}(r_i)$ from the $SPT_i$;
    forall the $y \in BL$ do
        compute $C_j \leftarrow E_{k_{ta}^+}\big(r_i\big)^y$;     // the elements of a check set $C$
        if $SP_i \stackrel{?}{=} C_j$;                      // test if $SP_i$ is in the blacklist
        then
            reject;                                        // reject the e-ticket
            map $C_j \mapsto y_i$;                          // map $C_j$ to the respective $y_i \in BL$
            mark $y_i$;                                    // mark $y_i$ that caused the hit
    accept;
```

The homomorphic property (5.3) can then be expressed as follows:

$$\begin{aligned} E(x \cdot r) &= g^{(x \cdot r)} \\ &= \big(g^x\big)^r \qquad (mod \ \ n) \\ &= E(x)^r. \end{aligned}$$

In order to enable efficient singulation (correlating different session pseudonyms SP_j to a single static one P_i^A, see Section 5.3.2), a trapdoor due to Okamoto-Uchiyama [118] can be used. That is, if n is a large composite number which is computed as specified in [118], then knowing the corresponding factorization of n, the discrete logarithm can be efficiently computed. For other parties without the knowledge of the factors of n, the factorization stays extremely difficult (see details in [118]).

Note, that the choice of the underlying encryption scheme is not limited to the one presented above. In principle, any other (deterministic[1]) encryption function with homomorphic properties as expressed in Equation (5.3) can be used.

5.3.3.2 Encrypted Blacklists

In the basic setting described above, the blacklisted pseudonyms $\{y\}$ are stored in clear in the blacklist BL, see Equation 5.4. However, in certain cases it may be desirable to operate on encrypted blacklists. An example would be to introduce a problem of terminal compromisa-

[1] In case of probabilistic encryption, the randomization factor must be additionally delivered to the terminal for blacklist check.

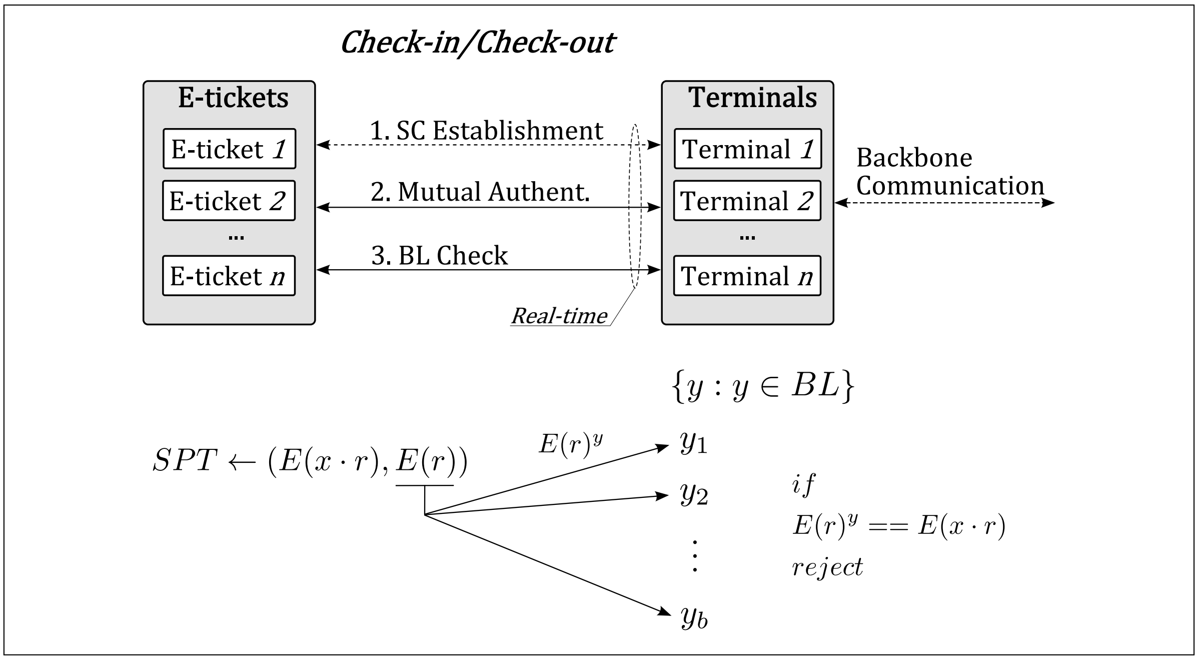


Figure 5.5: Local revocation check: visual representation.

tion resulting in leakage of terminal-side information to third parties. In order to provide for encrypted blacklists, a few modifications to the original version are necessary, more specifically with respect to *(1)* blacklist structure and *(2)* session pseudonym, SP. Moreover, two encryption functions with different exponentiation bases g and h respectively are going to be required (see the original encryption function expressed in Equation 5.8 for comparison). The underlying encryption functions, therefore, can be expressed as follows:

$$\begin{aligned} E_g(x) &= g^x; \\ E_h(x) &= h^x. \end{aligned} \quad (mod\ \ n) \tag{5.10}$$

Note that in Equation (5.10) above, the indices g or h represent the respective base used for exponentiation. Hereinafter, it is already implicitly assumed that encryption is performed under the public key of a TA.

New Blacklist Structure

Let $\{y\}$ be a set of blacklisted e-ticket pseudonyms, as in the basic scheme. Then considering the new encryption functions (Equation (5.10)) an encrypted blacklist is represented by a set of tuples (a, b) corresponding to each blacklisted element y and can be constructed by a TA as follows:

$$BL \leftarrow \{(a, b)\} \tag{5.11a}$$

where $a \leftarrow E_g(y)$, $b \leftarrow E_h\left(y^{-1}\right)$. Therefore,

$$BL \leftarrow \left\{\left(E_g(y),\, E_h\left(y^{-1}\right)\right)\right\}. \tag{5.11b}$$

Taking into account Equation (5.10), we obtain:

$$BL \leftarrow \left\{\left(g^y,\, h^{y^{-1}}\right)\right\} \tag{5.11c}$$

A New Structure of the Session Pseudonym

A session pseudonym SP as initially expressed in Equation (5.2), has to be calculated in a slightly different way, namely:

$$SP \leftarrow E_g(x) \cdot E_h(r), \tag{5.12a}$$

where x represents a static pseudonym of an e-ticket that has to be checked, see Table 5.2. Taking into account Equation (5.10), we get:

$$SP \leftarrow g^x \cdot h^r. \tag{5.12b}$$

A new form of the session pseudonym, therefore, essentially resembles the basic construct of Pedersen commitment scheme [119].

The initial form of a Session Pseudonym Tuple (SPT), see Equation (5.5), has to be altered accordingly:

$$SPT \leftarrow \left(E_g(x) \cdot E_h(r),\, t\right), \text{ where } t = x \cdot r. \tag{5.13a}$$

Taking the expression for the encryption functions into account (Equation 5.10), we get:

$$SPT \leftarrow \left(g^x \cdot h^r,\, t\right) \tag{5.13b}$$

and substituting t with $x \cdot r$:

$$SPT \leftarrow \left(g^x \cdot h^r,\, x \cdot r\right). \tag{5.13c}$$

Revocation Check with Encrypted Blacklists

In the setting with encrypted blacklists (see Equations (5.11)) and the updated method for calculation of session pseudonyms tuples (see Equations (5.12)), the procedure for blacklist check is performed as follows. Firstly, a terminal extracts t from SPT (recall Equation (5.13b)). Then operating on a blacklist (see Equations (5.11)), the elements $\{c\}$ of an auxiliary check set C are computed as follows:

$$c \leftarrow b^t \cdot a; \tag{5.14a}$$

Taking into account Equations (5.11b) and (5.11c), we obtain respectively:

$$c \leftarrow \left(E_h\left(y^{-1}\right)\right)^t \cdot E_g(y) \tag{5.14b}$$

$$c \leftarrow \left(h^{\frac{1}{y}}\right)^{x \cdot r} \cdot g^y, \quad \forall y \in BL,\ t \in SPT. \tag{5.14c}$$

Having performed the aforementioned operations, a terminal acts as in the basic version (see Section 5.3.3.1). Namely, it pairwise compares the received SP with the elements of C set:

$$c \overset{?}{=} SP \ \ \forall c \in C.$$

If a match is found, the e-ticket is blacklisted.

5.3.4 Boosting Performance

The presented approach for privacy-preserving local revocation based on blacklists (both variants) has linear complexity with respect to the number of elements in the blacklist. In order to boost the performance, the anonymity set of each session pseudonym SP can be reduced in a controllable way by leveraging the concept of k-anonymity initially presented in [120]. Namely, an e-ticket can additionally deliver its k-anonymous identifier (signed by a TA) to a terminal during check-in/check-out. This would substantially lower the search time over the respectively partitioned black list and render $\mathcal{O}(1)$ complexity with respect to the number of elements in the blacklist.

5.3.5 Privacy-preserving Mutual Authentication

5.3.5.1 Bootstrapping Well-established Techniques

A core building block of our solution is mutual authentication between an e-ticket and a terminal. It is of paramount importance to check the authenticity of communication partners before engaging into a check-in/check-out session. Note that the majority of the existing privacy-preserving approaches reviewed in Chapter 4 (see Table 4.19) does not provide for mutual authentication. As discussed in Section 5.1.1, due to the specific challenges of ESPT systems which are in focus of this dissertation, many advanced cryptographic constructions providing for mutual authentication do not suit our purposes. For that reason, a variation of the widely adopted certificate-base authentication (e.g., with RSA or ECC as underlying cryptographic primitives) is suggested for the mutual authentication block of our solution. The details follow in the next section.

5.3.5.2 Key Types

Depending on the subscription type (e.g., weekly, monthly, semester, or yearly pass), an e-ticket is initialized with the respective group signature key pair, that is $K_e \leftarrow (k_{gr}^+, k_{gr}^-)$. An e-ticket, therefore, is not distinguishable within that particular group. To the contrary, each terminal possesses a unique key pair $K_t \leftarrow (k_t^+, k_t^-)$. Both, K_e and K_t are signed by the transport authority (TA) and therefore can be easily checked for authenticity (the public key of a TA k_{ta}^+ is known to all parties). Therefore, the mutual authentication step (see Figure 5.3) can be efficiently performed by employing simple but powerful and efficient certificate-based authentication scheme. The employed key types are listed in Table 5.3 for clarity.

Table 5.3: The employed key types for mutual authentication

Key	**Type**
$K_e \leftarrow (k_{gr}^+, k_{gr}^-)$	group key pair of an e-ticket;
$K_t \leftarrow (k_t^+, k_t^-)$	unique key pair of a terminal;
$K_{ta} \leftarrow (k_{ta}^+, k_{ta}^-)$	unique key pair of a TA;

5.3.5.3 Integration Into Real-world Systems

An additional benefit of leveraging well-established techniques for mutual authentication lies in compatibility with the ESPT systems already deployed in the real-world scenario. For example, many public transport companies in Germany comply with the so-called core application standard (VDV-KA), see [36]. Its security architecture relies *inter alia* on RSA-based certificates [121]. Therefore, the front-end devices in ESPT conform to VDV-KA are inherently capable of handling operations required for mutual authentication considered in our solution. That in turn substantially minimizes additional effort of integration our privacy-preserving approach into real-world systems.

5.4 Limitations of the Proposed Solution

The proposed solution covers all requirements discussed in Section 3.5 and goes in line with the adopted attacker model presented in Section 3.6. It has, however, certain limitations as well. Firstly, secure key management must be supported by a user device. For example, the corresponding private key k_{gr}^- (see Table 5.3) must be secured accordingly (never leave the dedicated secure area of the device, ect.). Smart cards provide such a mechanism by default. In case of NFC enabled smartphones, however, additional techniques have to be considered to secure the private key. A possible way in the latter case would be to leverage the mechanisms provided by secure elements (see the respective discussion in Section 2.5.2) and additional software protection.

Secondly, the computations with respect to the creation of a session pseudonym (SP) on check-in/check-out (see Section 5.3.2) must be carried out in the secure area of a user device. Otherwise, an arbitrary string instead of the correct SP could be delivered to the terminal. Relying on secure computations enforced by hardware (and additionally software) is, however, rather typical for the majority of the deployed e-ticketing systems for which using advanced and hence extremely resource demanding cryptographic constructs is prohibitive. Moreover, in the architecture of New German Electronic Identity Card (nPA) [122] rolled out in 2010, similar assumptions were made with respect to the calculation of a provider-specific revocation token by a smart card. Namely, during revocation check, service providers have to trust that the so-called revocation token has been calculated correctly by the smart card (for details, see [123, 124]). Well-formedness and correctness of the revocation token itself are not explicitly checked. It should be noted, however, that it is in theory possible to further enhance revocation check mechanism of our solution by additionally employing special signature schemes based on advanced zero-knowledge techniques as for example, CL-signatures [125]. The latter allow to prove the possession of a valid signature for a concealed parameter (such as a static pseudonym P_i^A in our case) without revealing the parameter itself. Moreover, they provide for unlinkability of proofs across multiple sessions and allow to cryptographically bind the encrypted nonce value r to the corresponding randomization parameter used for the creation of the session pseudonym, see Equation (5.5) in Section 5.3.3. Unfortunately, the application of such advanced cryptographic methods would greatly increase the complexity of our scheme for revocation check. That would in turn negatively affect the overall front-end performance, which is time-critical in our case. Similarly, such complexity issues and the resultant performance considerations were discussed in the context of the New German Identity card, see [126].

Thirdly, certain cryptographic operations required for local revocation check (e.g., discrete exponentiation) are not directly supported by smart cards out of the box. Therefore, additional effort may be required (depending on the smart card type) for implementing the developed approach in the smart card area. However, in case of NFC smartphones, there are cryptographic libraries and hence respective API available which allow for efficient implementation of our solution.

Furthermore, the price for the privacy-preserving property achieved during the local revocation check (see Section 5.3.3) is efficiency. The proposed solution in its basic form has linear complexity in the number of blacklisted e-tickets. However, as discussed in Section 5.3.4, the performance can be substantially enhanced by segmenting the set of all e-tickets and reducing the anonymity in a controllable way by using the notion of k-anonymity originally presented in [120]. The challenge in this case is, however, ensuring that e-tickets are equally distributed across the k-anonymity sets and that specific travel patterns of each particular user (e.g., traveling at night, etc) do not leak additional information to terminals which could be misused to abuse user privacy.

Lastly, the implications of data mining techniques with respect to customer travel patterns processed in the back-end have not been directly addressed within this dissertation. Further research is required to explicitly address this issue which is out of scope of the current work. It is assumed that in a large city, travel patterns consisting of check-in/out locations corresponding to stops/stations (which have a rather quantized[1] nature) will not leak enough information to enable customer identification in the back-end. Moreover, the developed privacy-preserving framework relies on the non-collusion assumption between the TA and TTP. Should that be violated, privacy against the back-end (see *Requirement 1c* in Section 3.5) could not be guaranteed any more within the adopted attacker model (see Section 3.6).

5.5 Integration Effort: Estimation

The proposed solution can be applied to the majority of real-world systems at a relatively low cost due to the following reasons:

[1] Note, however, that personal travel patterns originating from much more precise GPS or cellular data are far more privacy invasive (see [74] for instance).

1. *Loose-coupling.* Unlike the majority of other privacy-preserving solutions (see Chapter 4), ours is based on a semi-coupled architecture which is adopted almost in every large-scale e-ticketing system for public transport (ESPT) in the world.

2. *Several non-colluding entities for billing and payment transactions.* As already discussed in Section 5.2, in many real e-ticketing systems like the ones conform to the German standard VDV-KA [36], several independent entities cooperate in order to handle payment transactions and provide for interoperability. For example, in case of VDV-KA, multiple transport authoritys (TAs) cooperate to ensure seamless travel for customers and share the revenue. Actual payment transactions with the end user are performed by other dedicated entities participating in a system – the so-called KVP (Kundenvertragspartner). Therefore, the proposed solution with an external trusted third party (TTP) would solely require rerouting the system flow via another dedicated privacy-protecting party (namely, TTP) by leveraging the existing interfaces. Note that the deployed VDV-KA system is already using external parties primarily for payments handling. For this reason, an additional effort of integrating a dedicated TTP into the existing infrastructure can be substantially minimized.

3. *Capabilities required from end user devices.* Most of the building blocks of our solution leverage the cryptographic primitives which can be efficiently implemented on the majority of the user devices currently available on the market. For example, mutual authentication is performed using the slightly tuned version of widely used certificate-based authentication efficiently supported by both, smart card and smartphone industry.

Chapter Summary

In this chapter, our solution for constructing privacy-preserving e-ticketing systems for public transport (ESPT) was presented. Firstly, the main building blocks comprising the suggested approach were presented in Section 5.1. In order to provide for basic intuition behind our solution, a respective outline was made in Section 5.2. A detailed description was presented in Section 5.3. Lastly, the limitations of the approach together with the estimated effort for its integration into real-world systems were discussed in Section 5.4 and Section 5.5 respectively.

6 Evaluation

In the previous chapter, our solution allowing for the development of privacy-preserving e-ticketing systems in the public transport environment has been presented. In what follows, the developed approach is going to be assessed in detail. First, the conceptual evaluation is presented in Section 6.1. It is primarily targeted at formal assessment of requirements satisfaction provided by the proposed framework as well as at the degree of achieved privacy and security properties of the system. Second, the feasibility of the proposed solution is demonstrated by two developed prototypes which description and performance analysis are provided in Section 6.2. More specifically, the NFC prototype targeted at NFC-enabled smartphones as a user device is presented in Section 6.2.1. The prototype developed for the smart card platform is discussed in Section 6.2.2. The chapter is concluded with Section 6.3 where the research questions introduced in Section 1.3 are discussed.

6.1 Conceptual Evaluation: Satisfying the Core Requirements

In Chapter 3, the core requirements of a privacy-preserving e-ticketing system under design were discussed together with the adopted attacker model (see Section 3.5 and Section 3.6 respectively). In what follows, it is demonstrated that these requirements are satisfied under the adopted attacker model and the respective discussion is performed.

First, let us recall the set of core requirements discussed in Section 3.5. For clarity, they are presented again within this section in Table 6.1. Next, each requirement is separately analyzed against within each dedicated section below.

6.1.1 Privacy

Protecting customer privacy in the e-ticketing scenario is the main goal of this dissertation. The subsequent evaluation of the achieved privacy properties is performed with respect to the capabilities of possible adversaries formally defined as an attacker model and discussed in Section 3.6.

6.1.1.1 Privacy against external observers

There are two main aspects providing evidence that our solution is secure against external observers.

First, due to mutual authentication between an e-ticket and a terminal (see Sections 5.1.1 and 5.3.5), the former will not engage into the full protocol without authenticating the terminal at first. That renders querying attacks from unauthorized terminals/rogue devices essentially ineffective as no useful information would be released from an e-ticket.

Table 6.1: A summary of core requirements

1. *Privacy*
   a) *Privacy against external observers.* An external attacker must be prevented from deriving any PII from interaction between e-tickets and terminals in the front-end.
   b) *Privacy against terminals.* Terminals must be prohibited from tracking and distinguishing between valid e-tickets as well as from identifying the users associated with them.
   c) *Privacy against the back-end.* The back-end is allowed to correlate travel records related to a single e-ticket while being prohibited from identifying the users associated with e-tickets.
2. *Fine-grained billing support*
3. *Loose-coupling*
4. *Efficiency.* Check-in/out events handling must comply with the timing requirements.
5. *Multilateral security*

Second, the communication between a customer device managing an e-ticket and a terminal is secured through the respective secure session mechanisms discussed in Section 5.3 (see *SC Establishment* in Figure 5.3). Therefore, an external observer tapping the communication in the front-end of the system is essentially prevented from learning any useful information about the e-ticket (and its owner) beyond the mere fact that the communication takes place. As a result, privacy against external observers is guaranteed by the security of the underlying scheme for secure session establishment. The latter in turn relies on the cryptographic strength of the primitives used for key exchange (e.g., DH/ECDH or RSA/ECC) and subsequent channel encryption (e.g., AES, 3DES, etc.).

As a result, under the assumption that the aforementioned cryptographic primitives (with respective key lengths) are secure, our solution provides protection against external observers.

6.1.1.2 Privacy against terminals

According to the privacy requirement 1 *b*) (see Table 6.1), terminals must not be able to distinguish between different e-tickets, track them or identify a customer associated with them. From the terminal's perspective, several sessions with the same e-ticket appear to be computationally indistinguishable from the ones with different e-tickets (see the terminal's view on the front-end session in Figure 5.4). Moreover, neither mutual authentication nor revocation check leak any additional information allowing a terminal to perform tracking or to find out the concealed static pseudonym. More specifically, mutual authentication only reveals that an e-ticket is authentic and belongs to a valid group (see Section 5.3.5). For local revocation check, a terminal gets a session pseudonym tuple from an e-ticket consisting of a session pseudonym (SP) and an encrypted nonce used to create SP, recall Equation (5.5):

$$SPT \leftarrow \big(E_{k_{ta}^{+}}(x \cdot r),\ E_{k_{ta}^{+}}(r)\big).$$

Since encryption $E_{k_{ta}^{+}}(\cdot)$ is performed under the public key of a transport authority k_{ta}^{+}, a terminal is essentially prevented from learning the static pseudonym of an e-ticket (represented by x in the equation above) let alone the identity of the customer associated with it. In this case it is assumed that the underlying encryption function is secure (see Section 5.3.3). Therefore, a rogue terminal would have to solve a cryptographic problem which is commonly regarded to be extremely difficult (e.g., factorizing a large prime or solving a discrete logarithm) in order to compromise the desired privacy properties of the system. Consequently, our solution provides protection against possible privacy invasion from the terminal's side.

6.1.1.3 Privacy against the back-end

Achieving decent privacy properties against the back-end is fairly challenging, since at the back-end side, certain pieces of information pertaining to user rides are required to enable fine-grained billing. In our solution, the building block referred to as *path reconstruction* (see Section 5.1) covers this issue by essentially providing a privacy overlay which on the one hand supports fine-grained billing and on the other hand protects user identities from the back-end. More specifically, the developed pseudonymisation scheme allows the back-end to restore correlation between the rides pertaining to a user (see *Singulation* in Figure 5.4) and extract the static user pseudonym P_i^A. However, due to the fact that the mapping between the corresponding user ID and the TTP-side P_i^T is kept secret by the TTP, the user ID remains unknown to the back-end of a transport authority (TA). Since customer payments are forwarded to the TA from a TTP in an aggregated form, the TA is essentially prevented from performing any additional inference from payment information which may be used to infringe on user privacy. As a limitation of our solution, it should be mentioned that the mere possession of customer travel patterns (even without the knowledge of corresponding user identities) may incur additional privacy concerns in certain scenarios. This has been already discussed together with other limitations of the approach in Section 5.4. Note, however, that the complex issue of patterns mining is out of scope of this dissertation and is not considered in the attacker model (see additional assumptions in Section 3.6.1).

Summarizing, it can be concluded that the developed solution provides adequate privacy protection against the back-end within the adopted attacker model.

6.1.2 Fine-grained Billing Support

As discussed in Section 3.5.2, the support for fine-grained billing is an essential functional requirement for a modern e-ticketing system. In this dissertation, it is enabled through the *path reconstruction* building block realized as a custom pseudonymisation scheme (see Section 5.3.2 for details). The back-end maintains a set of travel records which were received from all terminals as a result of check-in/out events in the front-end. Each travel record contains session pseudonym tuples $\{SPT_j\}$ and context information such as time, station, etc. The travel records are sorted according to each billing period (e.g. monthly billing). Then the static customer pseudonym P_i^A is extracted from each SPT_j using the TA's secret key k_{ta}^{-} and the travel records are respectively correlated to each extracted P_i^A, $i \in T$ with T representing the subset of all e-tickets having been used within the current billing period. This process is collectively referred to as singulation (see Figure 5.4). After that, the travel history for each pseudonym has been obtained and the back-end can apply any type of billing and discount policy which was previously negotiated. There are essentially no restrictions on the supported policy types. End user billing is subsequently performed via the TTP as described in Section 5.3.2. It should be mentioned that a positive side effect of fine-grained billing support is the availability of statistics information. That is of high importance for any TA since it enables to greatly optimize public transport network and to respond to customer demands more effectively.

Therefore, it can be concluded that the suggested solution provides support for fine-grained billing at the same time guaranteeing the protection of customer privacy.

6.1.3 Loose-coupling

The developed privacy-preserving framework is inherently based on the loosely-coupled architecture. All time critical interaction in the front-end is handled locally without relying on real-time connection to the back-end (see Figure 5.3). More specifically, the time critical part of the framework encompasses *mutual authentication* and *local revocation*. Both of these building blocks were designed to be locally realized in the front-end part. Regular terminal updates from the back-end (black list updates, maintenance, etc.) are performed in an asynchronous fashion and do not directly affect the handling of check-in/out sessions. Therefore, the developed solution fully satisfies the loose-coupling requirement.

6.1.4 Efficiency

Handling of check-in/out sessions between e-tickets and terminals is the most time critical part of the system since it directly affects customer experience. In general, timing requirements for a single session between an e-ticket and a terminal range from 0.2 sec (London Oyster Card) to 2.0 sec (early versions of Singapore EZ-Link) [72]. It is desirable, however, that the maximum runtime does not exceed 1 sec.

The developed framework is based on loosely-coupled architecture thus eliminating network delays which otherwise would be an inherent part of the front-end session runtime. Moreover, the back-end does not introduce a single point of failure for front-end sessions handling.

As previously discussed, a check-in/out session consists of three parts: *(1)* secure session establishment, *(2)* mutual authentication, and *(3)* local revocation check. The latter part (local revocation) is the most time consuming since in its default version, the complexity is linear in the number of blacklisted e-tickets (see Section 5.3.3.1). However, the performance can be significantly enhanced if the revocation is performed on partitioned blacklists and each e-ticket additionally delivers its k-anonymous identifier to the terminal (see Section 5.3.4).

Mutual authentication is for the most part designed to bootstrap well-established techniques relying on certificate-based authentication (see Section 5.3.5). Due to the wide support of existing cryptographic libraries and hardware-based acceleration, the time required for mutual authentication is essentially negligible with respect to local revocation check.

Depending on the exact implementation, the secure session establishment can be either combined with mutual authentication or carried out in its own right. In the former case, a secure session *per se* can be established on virtually no cost. Otherwise, the underlying off-the-shelf mechanisms such as the ones compliant to NFC-SEC-01 [61] may be efficiently bootstrapped.

As a result, the efficiency of the check-in/out sessions is to a large extent defined by the most costly operation – local revocation check. As it is going to be shown further in Section 6.2, our practical tests have demonstrated that it is feasible to achieve acceptable performance using blacklist partitioning and k-anonymous approach.

6.1.5 Multilateral Security

Multilateral security (MLS) is a complex notion capturing the specific security goals of each party (which may be partly conflicting) as well as the process of negotiation to achieve common ground for all parties in a system [73]. In the suggested framework, there are two core parties with distinct security goals: *(1)* transport authority (TA) and *(2)* users of the system. The trusted third party (TTP) essentially acts as an external trusted mediator and in this case is not explicitly considered within the MLS context. From the TA's perspective, it has to be guaranteed that e-tickets cannot be forged (without the substantial effort rendering such forgery worthless) and that the used transport service is adequately paid for. The users, however, require that they are billed correctly and cannot be framed for making use of transport service in case that had not happened in the reality.

The *unforgeability of e-tickets* is ensured in the first place by the utilization of digital signatures during the issuance process. As discussed in Section 5.3.1, the necessary credentials for using a system cannot be created by the user herself but must be rather obtained from the TTP in a proper way. *The adequate payment* for the used transport service is ensured at the TA side by regular comparison of the overall calculated bill for all customers with the respective aggregated payment received from the TTP. Should the TA find out any inconsistencies, it forwards the claim for investigation along with the respective evidence to the TTP. The latter can then easily find out the culprit since it maintains the end user payment statistics (together with the respective mappings between all $\left\{P_i^t\right\}$ and user IDs). Moreover, before aggregating the payments, the TTP can run internal checks to make sure that every user has indeed paid accordingly. In case of any (repetitive) fraud/payment failures, the TTP forwards the respective user pseudonym P_k^T to the TA so that it can be included into the black list. At the same time, depending on the exact agreements and regulation, the TTP may forward the charge claim together with the fraud evidence to the respective law enforcement agency to enable a fine claim and costs compensation for the unpaid transport service.

The *billing correctness* can be ensured by regular external audits of TA (out of scope of this work). With this respect, the suggested framework does not differ from the majority of deployed e-ticketing systems for public transport (ESPT) which calculate bills internally and undergo audits. For the reasons discussed in Section 3.5.2.2, the user involvement into billing process (beyond interaction with terminals during check-in/out sessions) is considered to be rather impractical. Despite the fact that a user is not actively involved into billing process, he/she is in principle able to locally compare the correctness of the bill by analyzing the history of travels stored at his/her device. Note, however, that it requires additional software as well as the necessary capabilities of a user device (that is, it may be prohibitive for a smart card to store monthly statistics as a whole without, for example, regularly storing it in parts on external persistent storage). Should a user have evidence of incorrect billing, a respective investigation inquiry can be issued to the TTP.

It should be mentioned that the detailed analysis of each party's security goals together with careful consideration of possible conflicts and negotiation processes is out of scope of this work (which primarily focuses on privacy protection). Rather, it has been demonstrated that the core security goals of both parties the TA and users are indeed respected by the suggested framework and are not violated *inter alia* by the achieved privacy properties.

6.2 Practical Validation

The proposed solution presented in Chapter 5 describes a novel framework for developing privacy-preserving e-ticketing systems for public transport (ESPT). Whereas during the conceptual evaluation (previous section) the suggested system architecture has been assessed in its entirety, practical validation of all system processes involved would have been barely feasible within a single thesis. Recall that the developed framework, in large, consists of three core processes (see also Figure 5.3): *(1)* front-end interaction (time critical), *(2)* back-end processing, and lastly *(3)* distributed billing. The most critical set of processes from a user point of view is the first one (front-end interaction) since it directly affects user experience. Poor performance in the front-end is likely to be reflected in long lines being formed near terminals, nuisance, and eventually user dissatisfaction. Moreover, front-end interaction represents the part of a system most specific to the e-ticketing scenario in terms of hardware (user devices and terminals) and the respective pieces of software. Therefore, to demonstrate practical relevance and feasibility of our solution, its performance in the front-end has to be assessed in the first place (see Figure 6.1).

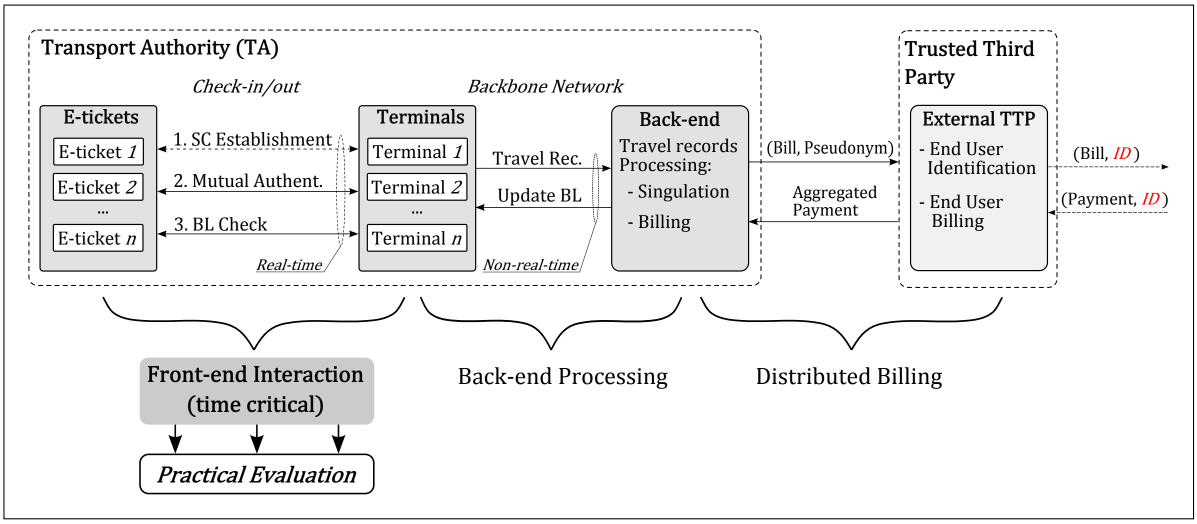


Figure 6.1: Practically evaluated part of the developed framework

6.2.1 The Developed NFC-based Prototype

6.2.1.1 Introductory Discussion

As discussed in Section 2.5, the front-end communication between user devices managing e-tickets and terminals is performed via a wireless interface compliant to RFID or NFC standard. In ESPT environment, NFC can be viewed as a unification of different RFID-based e-ticketing standards such as MIFARE, Calypso or FeliCa. Therefore, an NFC-enabled smartphone as a user device is in principle backwards-compatible with the legacy systems based on RFID. Providing the full support of all NFC modes as defined by the NFC forum (see Section 2.5.2) is, however, left to each smartphone vendor. Currently, the popularity of headsets equipped with NFC is rapidly growing. According to the report from analytics company IHS [127], the worldwide shipments of smartphones supporting NFC have been predicted to rise by 325% from 2013 to the end of 2018. Therefore, using a smartphone as a single user device for managing several e-tickets originating from different TAs (possibly on an international level) is no longer a pure vision (see Figure 6.2).

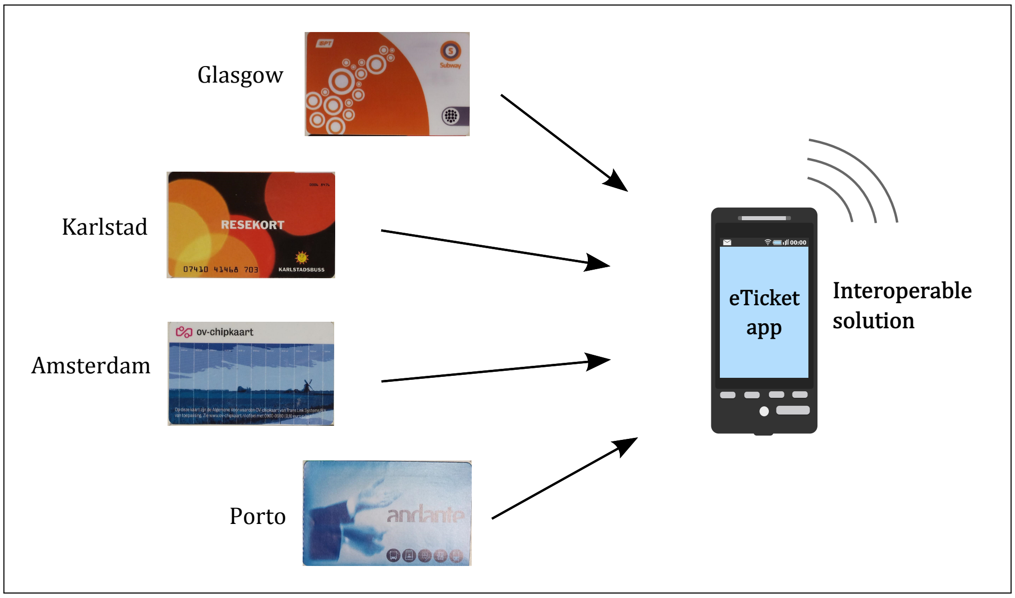


Figure 6.2: NFC-enabled smartphone as a single user device hosting multiple e-tickets

Moreover, NFC-enabled smartphones based on Android OS are currently dominating the market. It is reported that in 2013 around 93% of all smartphones supporting NFC were running Android and such upward trend is likely to be further maintained [127]. Taking the aforementioned issues into account, it was decided to validate the developed framework in the first place on NFC platform using commodity user devices with Android OS.

6.2.1.2 Prototype Description

The developed front-end prototype of ESPT is essentially comprised of *(1)* a user device (in the form of an NFC-capable smartphone) and *(2)* a terminal. The latter can be further divided into *(a)* NFC reader part and *(b)* main computing unit (see Figure 6.3 and Table 6.2). The decision to modularize the NFC terminal into two parts is aimed at decoupling the process of NFC communication (covered by the NFC reader part) and the protocol logic itself such as control of session parameters, black list check, etc. (performed by the main computing unit). Moreover, due to technical limitations of common USB NFC readers, the NFC reader part had to be implemented using custom hardware, see further.

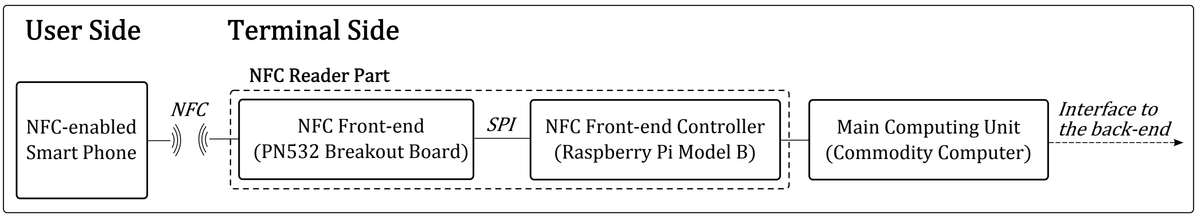

User Side
Terminal Side
NFC Reader Part
NFC-enabled Smart Phone
NFC
NFC Front-end (PN532 Breakout Board)
SPI
NFC Front-end Controller (Raspberry Pi Model B)
Main Computing Unit (Commodity Computer)
Interface to the back-end

Figure 6.3: The developed front-end prototype based on NFC. SPI stands for serial peripheral interface.

Hardware part. The pieces of hardware comprising the prototype are summarized in Table 6.2 for clarity. A user device was represented by a commodity NFC smartphone Samsung Galaxy Nexus GT-I9250. At the terminal side, the following pieces of hardware were used. First, the NFC reader part consisted of *(i)* the NFC front-end represented by PN532 Breakout Board (essentially acting as a controllable NFC antenna) and *(ii)* NFC controller in the form of Raspberry Pi Model B with 256MB RAM. The control logic pertinent to the front-end framework processes as described in Chapter 5 was handled by the main computing unit implemented by a commodity computer Dell Vostro 3700, Intel Core i5 M 460 (2,53GHz). Initially, in order to enable NFC communication, an attempt was made to leverage several existing USB-based NFC readers and by this to eliminate the necessity of building a custom NFC reader part. However, as the first practical tests have shown, the NFC library libNFC [128] installed at the main computing unit to control terminal-side communication, turned out to be intolerant of specific delays introduced by the interactive protocol implemented on top (see the discussion on the encountered issues further in Section 6.2.1.5). Therefore, a decision was made to resort to a custom built NFC Reader part which is based on PN532 Breakout board (essentially an NFC antenna) connected via a serial peripheral interface (SPI) bus to the Raspberry Pi Model B (depicted in Figure 6.3). Due to the SPI connection, the issues with delay intolerance and resultant messages drop could be overcome[1].

Software part. *At the user side,* the native Android OS was substituted by the open source CyanogenMod version 11 (based on Android 4.4) [130] due to the lack of support for firmware updates for the used NFC smartphone. Moreover, the utilization of CyanogenMod OS enabled more flexibility during prototype testing and debugging. At the user device, two distinct apps were implemented: *(1)* one to enable communication via the NFC interface *(NFC app)*

[1] The detailed analysis of different NFC reader types and their suitability for interactive NFC communication as well as the eventual implementation of the underlying interface via NFC were performed at our department by Manuel Weißbach within the dedicated student project [129]

Table 6.2: The front-end prototype based on NFC: hardware

Part	**Hardware piece used**
1. User side	Samsung Galaxy Nexus GT-I9250
2. Terminal side:	
a) NFC reader part:	
(i) NFC front-end	PN532 Breakout Board
(ii) NFC controller	Raspberry Pi Model B, 256MB RAM
b) Main computing unit	Dell Vostro 3700, Intel Core i5 M 460 (2,53GHz)

and *(2)* another one acting as an application-specific e-ticketing app *(E-ticketing app)*, see Figure 6.4.

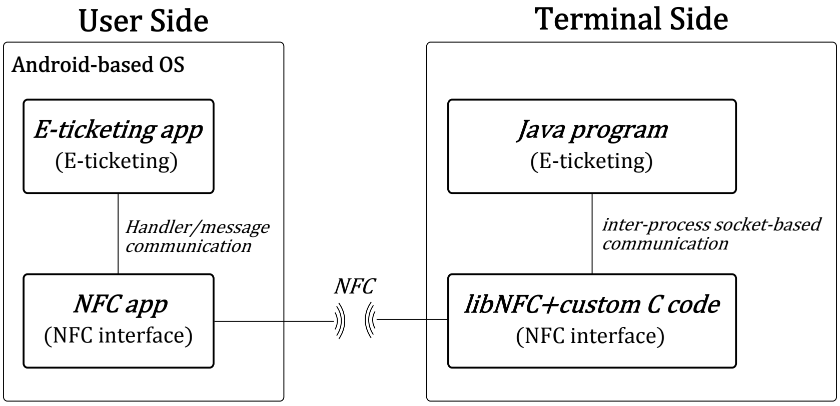


Figure 6.4: Communication stack in the developed NFC prototype.

The reason for this modularization is the following. Early tests have shown that the P2P mode (see Section 2.5.2) provided by Android OS in fact does not allow for repetitive interactive communication. It turned out that the only operation for which this mode was explicitly considered was a one-time stateless message exchange between two NFC devices after which the NFC connection gets automatically terminated. In order to overcome this, a decision has been made to implement the underlying interactive NFC communication using the so-called ”inverse reader mode” first introduced in [131] and further extended at our department in [129]. This mode emulates a fully interactive bidirectional communication by having a smartphone acting in a standard reader/writer mode and the NFC Reader part of a terminal in card emulation mode. As a result, in order to send a message to the terminal, a smartphone writes it to the tag emulated at the terminal side (write tag operation in reader/writer mode). The message gets read out from the emulated tag by the terminal and the new message to be sent to the smartphone is written to the tag (overwriting the old one). The smartphone then subsequently reads the new message out (read tag operation) and is ready to respond back in the way described above. At the smartphone side, the inverse reader mode is handled by the NFC app (see above) running in the background and abstracting the details of the underlying NFC communication from the e-ticketing app[1]. The latter is responsible for the actual implementation of the user-side protocol steps defined by the developed privacy-preserving framework.

At the terminal side, the inverse reader mode was handled by an open source C-based library for NFC communication libNFC [128] and a piece of custom code adding the necessary functionality to enable interactive NFC communication (see Figure 6.4). The logic responsible for the actual protocol prescribed by the developed privacy-preserving framework was implemented in Java.

6.2.1.3 Setting description

As discussed in Chapter 5, a single check-in/out event in the front-end of the developed privacy-preserving framework consists of three main steps: *(1)* secure session establishment, *(2)* mutual authentication, and *(3)* local revocation check (see Figure 6.5).

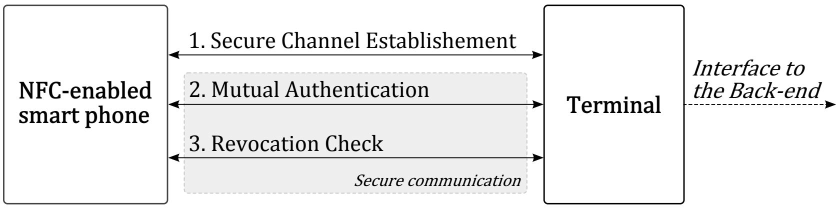


Figure 6.5: Main steps performed during a single check-in/out session

In order to secure the communication session (see the first step in Figure 6.5), an application-specific (i.e. handled by the e-ticketing app and its counterpart at the terminal side) Diffie-Hellman (DH) key exchange was used with modulo bit length being varied from 1024 to 4096. The exchanged DH key is used to derive a 256-bit AES key (operating in CBC[2] mode) to secure further communication. In order to provide for mutual authentication (step 2 in Figure 6.5), RSA-based certificates were used as discussed in Section 5.3.5 with key length varied from 1024 to 4096 bit. Lastly, local revocation check consists of the e-ticketing app calculating the session pseudonym tuple (SPT, see Section 5.3.3), sending it to the terminal side where the extracted session pseudonym (SP) is used to perform blacklist check. The size of blacklists was varied from 100 to 10000 elements. The implemented front-end session is summarized in Figure 6.6 for clarity with the respective notations presented in Table 6.3 for convenience.

6.2.1.4 Performance analysis

The front-end performance of the developed privacy-preserving framework was assessed by measuring the total execution time required to serve a single check-in/check-out event (excluding the delay introduced by the underlying NFC channel). The aforementioned run time was measured depending on the size of the blacklist at the terminal side for three different settings with key lengths for DH secure channel establishment and RSA-based mutual authentication of 1024, 2048, and 4096 bits. As presented in Figures 6.6, the run time consists of *(1)* secure

[1] The e-ticketing app together with the corresponding terminal part was implemented at our department by Pavel Plakhin within the dedicated student project [132] at our department.

[2] CBC stands for cipher block chaining.

E-ticket app		**Terminal**
$\left(k_{gr}^{+}, k_{gr}^{-}\right)$, k_{ta}^{+}, (g, q, p), P_i^A		$\left(k_t^{+}, k_t^{-}\right)$, (g, q, p), $S_t \leftarrow sign_{k_{ta}^{-}}\left(k_t^{+}\right)$
	1. Secure channel establishment	
Generate $a \in_R [1, q-1]$, $A \leftarrow g^a \mod p$	$\xrightarrow{A}$	
		Check that $A \in G = \langle g \rangle$ Generate $b \in_R [1, q-1]$, $B \leftarrow g^b \mod p$
	$\xleftarrow{B}$	
Check that $B \in G = \langle g \rangle$ Compute $k_{sh} \leftarrow B^a \mod p$, derive k_{aes}		Compute $k_{sh} \leftarrow A^b \mod p$, derive k_{aes}
	2. Mutual authentication	
Compute $S_A \leftarrow sign_{k_{gr}^{-}}(A)$	$\xrightarrow{S_A}$	
		Check S_A, compute $S_B \leftarrow sign_{k_t^{-}}(B)$
	$\xleftarrow{S_B,\, S_t}$	
Check $S_B,\ S_t$		
	3. Local revocation check	
Generate random r		
Compute $SPT \leftarrow \left(E_{k_{ta}^{+}}(P_i^A \cdot r),\ E_{k_{ta}^{+}}(r)\right)$	$\xrightarrow{SPT}$	
		Do revocation check (as discussed in Section 5.3.3, see Algorithm 1)

Figure 6.6: A summary of the implemented front-end session.

channel establishment, *(2)* mutual authentication, and *(3)* local revocation check. Protocol complexity is essentially defined by the terminal-side calculations during the last step – local revocation. The latter in turn linearly depends on the number of blacklisted elements currently comprising the blacklist. Therefore, the measurements were targeted at assessing the run time performance with respect to the number of the elements in the blacklist. The dependencies between the blacklist size and run time for key lengths 1024, 2048, as well as for 4096 bits are presented in Figure 6.7, Figure 6.8, and Figure 6.9 respectively.

As it was discussed in Section 3.5, the run time requirements vary from 200 ms to 2000 ms depending on each particular e-ticketing system. In most of the modern ESPT aiming at enhancing user experience, much effort is targeted at keeping the run time of a single check-in/out session below 1000ms. Therefore, in order to assess the performance of the implemented prototype, the following run time sets are considered for analysis:

1. *Acceptable performance* (AP). The set is defined by the area corresponding to the acceptable performance which is limited by point A with the associated run time of 2000 ms. The area is marked with light yellow color in Figures 6.7, 6.8, 6.9.

2. *Optimal performance* (OP). The set is defined by the area corresponding to the optimal performance which is limited by point B associated with the run time of 1000 ms. The

OP area is marked with light green color in Figure 6.7 and Figure 6.8. Note that the values corresponding to the *OP* area form a subset of the ones related to *AP*.

Table 6.3: Notations used for front-end protocol diagram in Figure 6.6

Notation	**Meaning**
$\left(k_{gr}^{+}, k_{gr}^{-}\right)$	a group key pair of an e-ticket;
$\left(k_{t}^{+}, k_{t}^{-}\right)$	a unique key pair of a terminal;
S_t	a public key of the terminal signed by the back-end;
$\left(k_{ta}^{+}, k_{ta}^{-}\right)$	a unique key pair of a TA;
(g, q, p)	parameters forming a group $G = \langle g \rangle$;
P_i^A	an e-ticket static pseudonym;
$E_{k_{ta}^{+}}\left(P_i^A \cdot r\right)$	a session pseudonym, SP_j;
$\left(E_{k_{ta}^{+}}(P_i^A \cdot r), E_{k_{ta}^{+}}(r)\right)$	a session pseudonym tuple (SPT);
k_{sh}	a shared key agreed upon using DH;
k_{aes}	a session AES key derived from k_{sh};
S_A	a public ephemeral DH key signed by the e-ticket;
S_B	a public ephemeral DH key signed by the terminal.

In the cases corresponding to key sizes of 1024 and 2048 bits (Figures 6.7, 6.8), the *optimal point* on the graph can be chosen from the subset of points lying in the area of optimal performance: $P_{opt}\left(K_{opt}, T_{opt}\right) \in OP$ such that $K_{opt} = 1000$. This point on the performance curve represents the run time value corresponding to the number of elements in the blacklist (against which the check is performed) equal to 1000. P_{opt} can be used for boosting the performance of front-end sessions by leveraging the k-anonymity approach and partitioning the blacklist respectively (as discussed in Section 5.3.4). Therefore, in the implemented setting, the k-anonymity parameter would correspond to $K_{opt} = 1000$. For privacy reasons, it would be desired to have this parameter as large as possible. The value of K_{opt}, however, is limited by the optimal performance area (OP) defined by the run time of 1000 ms (see point B on both graphs in Figures 6.7, 6.8).

In case of the last setting with 4096-bit keys (see Figure 6.9), no OP area can be observed, since even with only a few elements in the blacklist, the minimum run time was measured to be beyond 1500ms. The reason for this is the relatively long key size and consequently a much longer processing time required for secure session establishment, mutual authentication, and blacklist check (especially with respect to the user device part). However, even in this case an *acceptable* configuration can be determined which is defined by point P_{acc} in Figure 6.9 with corresponding blacklist size (or alternatively a k-anonymity parameter) of $K_{acc} = 1000$ elements and run time $T_{acc} = 1883\,ms$.

The optimal configuration (or alternatively the acceptable one in the last setting) achieved with our prototype is summarized in Table 6.4.

Table 6.4: NFC Prototype performance with 1000 elements in the blacklist.

Setting	**Key size**	**Run time**
1)	1024 bits	686 ms
2)	2048 bits	907 ms
3)	4096 bits	1833 ms

Therefore, it can be summarized that the developed prototype has demonstrated the viability of the front-end part of the suggested privacy-preserving framework. The observed performance lies within the optimal area (with run time less than 1000ms and blacklist size of 1000 elements) for key lengths of 1024 and 2048 bits currently actively used in practice. According to the recent report from the Federal Office for Information Security (BSI) [133], the key length of 2048 bits is going to remain secure till the end of 2020 and for this reason it is explicitly advised to be used for future applications. Hence, the achieved performance for the second setting with 2048-bit keys does not only demonstrate the real-world pertinence[1] of our prototype at this moment but at the same time indicates its future-proofness (with respect to the key length).

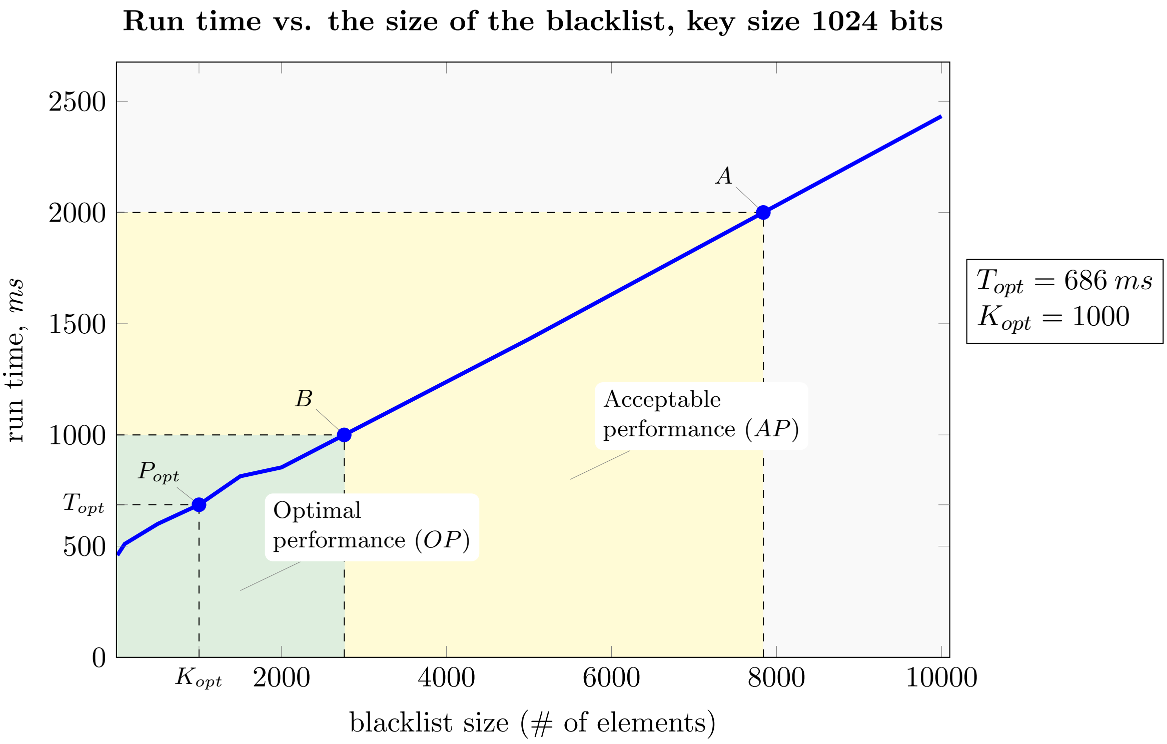


Figure 6.7: NFC prototype. Performance of check-in/out events handling with respect to the size of the blacklist. Key sizes used are 1024 bits.

[1] Moreover, further code optimization is likely to enhance prototype performance for setting 3 with key sizes of 4096 bits.

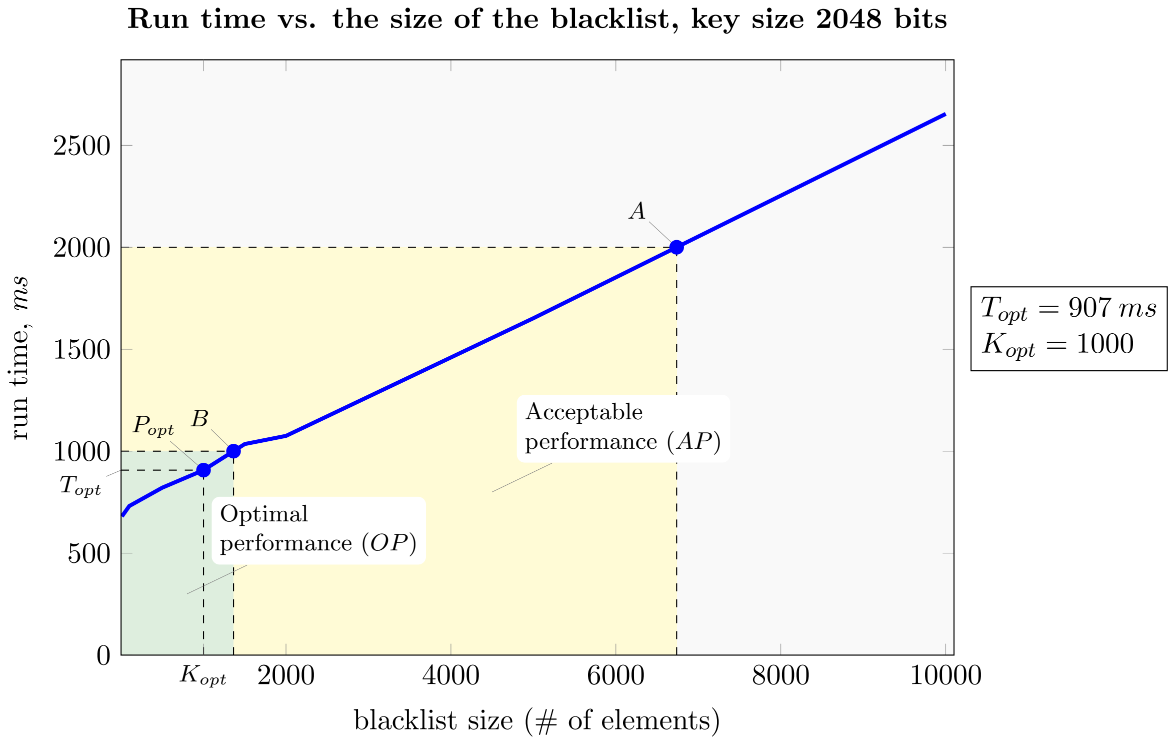


Figure 6.8: NFC prototype. Performance of check-in/out events handling with respect to the size of the blacklist. Key sizes used are 2048 bits.

6.2.1.5 Encountered Issues and Collateral Findings

During the implementation work, several challenges were faced and needed to be overcome. Firstly, the implementation of the underlying fully interactive, stateful communication via the NFC interface turned out not to be directly supported by the available NFC-related API of Android. That is to say, none of the three modes considered by the NFC forum (P2P, reader/writer, and card emulation) could be directly leveraged to implement an interactive protocol on top of NFC communication. Despite of the declared support for a P2P mode which is the first obvious choice to leverage, it turned out that after a single round of bidirectional messages exchange, the communication would be automatically terminated. In order to perform another exchange round, NFC communication had to be initiated again by removing the devices from mutual vicinity area and returning them there again. This has rendered the off-the-shelf P2P mode unsuitable for the prototype. Due to this fact, a decision was made to resort to a custom mode based on the so-called ”inverse reader mode” firstly mentioned in [131] and further enhanced at our department in [129]. The concept is based on having a user device acting in a conventional reader/writer mode (constantly changing between writing and reading operations) and the terminal being run in a card emulation mode (essentially emulating a re-writable NFC tag). Then, a user device performs a write operation to send a message to the terminal side and a read operation to get the new message from the terminal. The latter having read the last message written to the emulated tag, overwrites it with the new

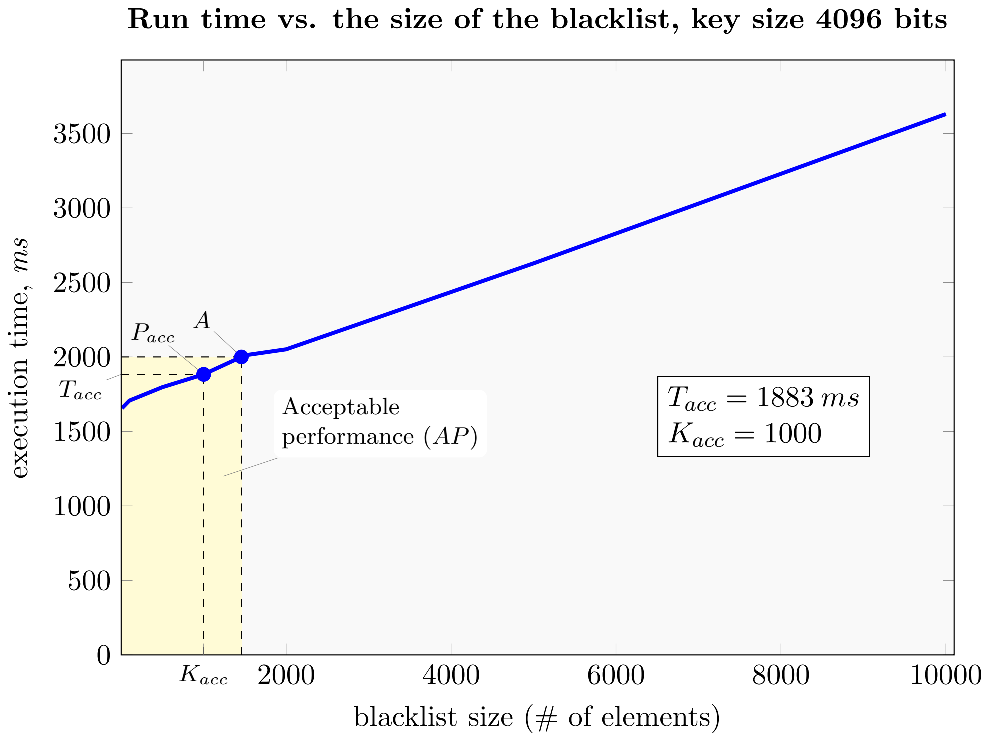


Figure 6.9: NFC prototype. Performance of check-in/out events handling with respect to the size of the blacklist. Key sizes used are 4096 bits.

one to be transmitted to the user device. By this, a fully interactive stateful communication over the NFC interface could be achieved.

Another issue encountered with respect to the implementation of the custom inverse reader mode is related to commercially available NFC readers. It turned out that most of them were unable to efficiently support card emulation mode using the libNFC library. More specifically, card emulation was either not supported at all or it was not functioning stably due to timing-requirements imposed by libNFC (especially in case of longer messages [129]). The reason is likely to be the specific model of NFC chip used (with the respective low-level NFC stack) and the connection type between the NFC chip and the rest of the circuitry. For example, with respect to the latter issue (connection type), the USB-to-UART[1] bridge turned out to be unstable in our case compared to the connection via a serial peripheral interface (SPI) which is widely used in industry. The possible reason may be the SPI interface is faster than the USB-to-UART bridge. Due to these limitations, it was decided to implement an NFC reader using the NFC chip with card emulation support (namely PN532-NFC model) and a controlling single-board computer Raspberry Pi interconnected via a faster serial peripheral interface (SPI), see Figure 6.3. It has to be additionally noted that during this research, it could be concluded that to the best of our knowledge the majority of industrially deployed NFC readers are based on SPI connection.

[1] UART stays for universal asynchronous receiver transmitter.

Moreover, an attempt was made to further enhance front-end performance by leveraging the elliptic curve cryptography (ECC) with smaller key sizes instead of RSA. Contrary to our expectations, the achieved performance (especially at the user side) was nearly one order of magnitude worse (i.e. slower) than in case of RSA. The reason for this is likely to be twofold. First, the cryptographic Java-based library used for the implementation (SpoungyCastle 1.5.0.0) seems to provide a rather inefficient support for ECC. Being confronted with the similar issue, the authors of [134] further elaborate that it can be caused by the fact that the intrinsic operations required by ECC such as point multiplication are internally implemented on top of the underlying Java BigInteger class leading to considerable inefficiency. Second, the user device may not be providing dedicated hardware support (e.g. hardware acceleration in the form of a co-processor) for ECC-related operations.

6.2.2 The Developed Prototype for the Smart Card Platform

A separate prototype was developed to assess the front-end performance in case a smart card with an RFID contactless interface is used as a customer device. A Java card platform [135] was chosen for the implementation. Namely, an NXP J3A080 Java smart card was utilized as a user device. The terminal side was comprised of a smart card reader represented by Reiner SCT cyberJack RFID and a controlling machine Intel Quad Core i5, 2,4 GHz each, 8 GB RAM with Ubuntu 12.04 (64 bit), see Table 6.5.

Table 6.5: The front-end prototype based on RFID and Java Card Platform: hardware

Part	**Hardware piece used**
1. User side	J3A080 NXP Java Card
2. Terminal side:	
a) RFID reader part:	Reiner SCT cyberJack RFID
b) Main computing unit	Intel Quad Core i5 (2,4 GHz)

The Java card model which was available for our tests does not provide direct support for modular multiplication and exponentiation unlike other cards based on MULTOS platform [136] which have been used, for example, during IRMA ("I Reveal My Attributes") project [137]. It is in principle possible to overcome these limitations by leveraging the RSA cryptographic co-processor of a Java card through delegation of modular exponentiation to low-level implementation of RSA encryption function as it was performed in [138]. Modular multiplication can be realized using an external library which was tested at our department within the corresponding student project [139]. The resultant efficiency, however, remains essentially prohibitive for this generation of Java cards. Therefore, it was decided to slightly modify the initial framework by using an XOR operation instead of modular multiplication and RSA encryption instead of custom modular exponentiation for *(1)* session pseudonym creation and respectively for *(2)* blacklist check. More specifically, in the modified version, the session pseudonym is computed as follows:

$$SP_j \leftarrow E^{rsa}_{k^+_{ta}} \left(P^A_i \oplus r \right) \tag{6.1}$$

The session pseudonym tuple is then expressed as follows:

$$SPT_i \leftarrow \Big(E^{rsa}_{k^+_{ta}}\left(P^A_i \oplus r,\right)\ r\Big) \tag{6.2}$$

Note that in this case, the randomization factor r is sent in clear to the terminal, which however does not jeopardize security. At the terminal side, local revocation is performed by extracting r from SPT_i, pairwise applying XOR operation to each element in the blacklist $BL : \{y\}$, and subsequently encrypting the result under the public key of a TA. Lastly, the outcome is compared with the delivered SP_i:

$$\forall y \in BL,\ c \leftarrow E^{rsa}_{k^+_{ta}}\left(r \oplus y\right),\ c \overset{?}{=} SP_i. \tag{6.3}$$

Thus by performing the aforementioned slight alterations to the initial front-end protocol discussed in Chapter 5, the support for this class of Java cards could be provided. Note that for the rest of the developed framework, it is essentially transparent how exactly the defined front-end interaction is implemented. The general logic of a framework retains its properties as well. It has to be mentioned, however, that in the altered version adapted to the Java card platform, the realization of encrypted blacklists (as discussed in Section 5.3.3.2) is not possible without further modifications and computation tricks mentioned at the beginning of this section.

Java card platform provides an internal support for DH key agreement. Therefore, there was not need to create an application-defined key exchange as it was done in case of the NFC prototype (see Figure 6.6 for comparison). With respect to the rest of the protocol, mutual authentication and local revocation steps essentially follow the pattern presented in Figure 6.6 during the discussion of the NFC prototype (with the slight modifications discussed above). Due to performance reasons, the key lengths used during testing were limited to 1024 bits. The evaluation results[1] are presented in Figure 6.10.

6.2.2.1 Performance analysis

As the measurements have demonstrated, the performance of the smart card prototype is in general worse compared to the NFC implementation. The area of optimal performance (OP) as defined in Section 6.2.1.4 could not be observed at all, similarly to the third setting of NFC prototype with 4096-bit keys. Analyzing the achieved acceptable performance (see point A in Figure 6.10), an acceptable configuration can be determined. The latter is represented by point P_{acc} with the number of blacklisted elements (or alternatively a k-anonymity parameter) of $K_{acc} = 1000$ and run time $T_{acc} = 1856\,ms$ (see Figure 6.10).

The achieved performance of the developed smart card prototype can be explained by the resource constrained environment of the Java card used. The latter is reflected in the first place in the processing time pertaining to relatively heavy cryptographic operations, such as RSA signature generation and verification as well as decryption. It has to be mentioned that the results of benchmarking tests performed in [141] for the model of Java card used in our prototype back this claim. However, despite the inherent limitations of the Java card platform,

[1] The implementation work for the Java card prototype was performed at our department within the respective student project [140].

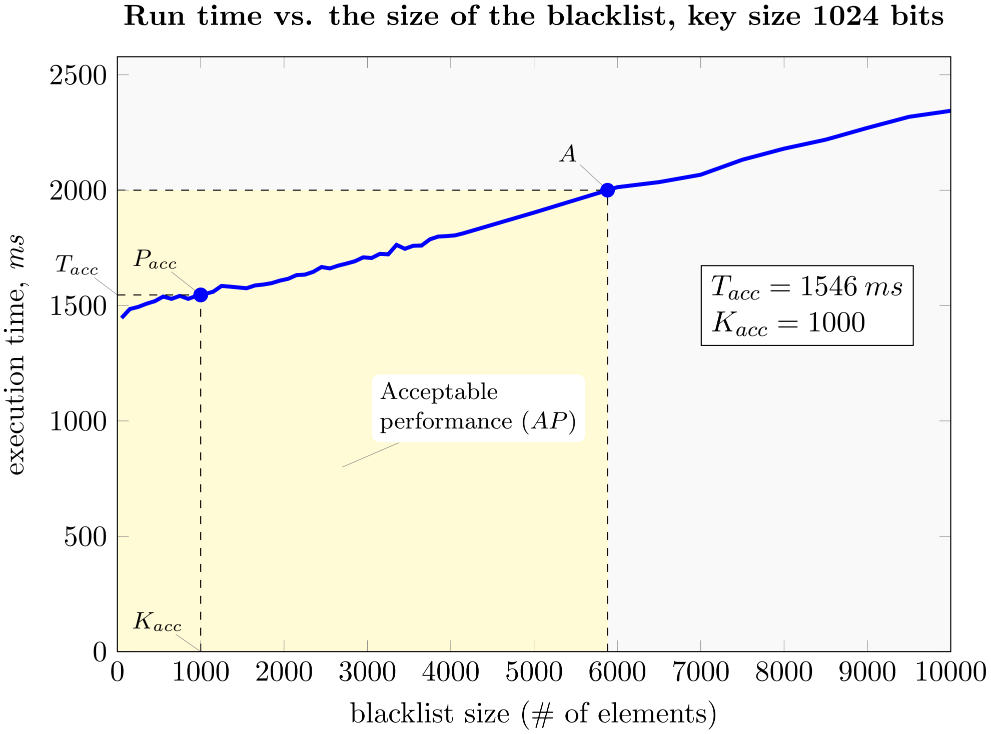


Figure 6.10: RFID prototype. Performance of check-in/out events handling with respect to the size of the blacklist. Key sizes used are 1024 bits.

the developed prototype demonstrated acceptable performance and hence the feasibility of the developed solution in the smart card environment.

6.2.2.2 Potential for Performance Optimization

Due to the specifics of Java card application development, it can be assumed that further code optimization can enhance prototype performance. Moreover, the smart cards based on other operating systems (such as MULTOS [136]) provide more APIs for low-level mathematical operations and memory management and consequently are likely to demonstrate significantly better results including more efficient handling of RSA keys with length greater than 1024 bits. In our case, the equipment already available at the department was used for the evaluation.

6.2.3 Evaluation Results: Summary

Summarizing the evaluation results, it can be concluded that the front-end of the developed privacy-preserving framework was successfully evaluated for two main use cases: NFC-enabled smartphones and smart cards. In general, the NFC prototype performed better than the smart card one taking into account the number of blacklisted elements, run time, and key sizes. For both proof-of-concept implementations, further code optimization is likely to provide for performance gain and better stability. In case of smart cards, the utilization of newer hardware (for the user device) is expected to bring significant enhancements to the overall performance.

6.3 Covering Research Questions

As presented at the beginning of the thesis in Section 1.3, the main goal of this work is the development of a privacy-preserving framework for building loosely-coupled e-ticketing systems for public transport allowing *(1)* local validation of e-tickets and *(2)* supporting fine-grained billing for registered customers. Based on that, the specific research questions were derived in Section 1.3 which now can be answered below.

Research Question 1.

How to provide for a privacy-preserving local validation at the terminal side such that:

a) valid e-tickets remain anonymous to the terminal;

b) invalid e-tickets are rejected.

As presented in Chapter 5, the developed solution essentially consists of three main building blocks: *(1)* mutual authentication, *(2)* local revocation, and *(3)* path reconstruction. The suggested mutual authentication mechanism discussed in Section 5.3.5 ensures that *(a)* user devices (managing corresponding e-tickets) engage into full communication sessions only with authorized terminals, *(b)* terminals can check the validity of travel permission and at the same time learn no further information on the e-ticket itself (no tracking or identifying the associated user is possible). Furthermore, the developed approach for local revocation presented in Section 5.3.3 ensures that the blacklisted e-tickets (for example, the ones associated with customers who failed to pay their bills, or in case of theft, etc.) are rejected without negatively affecting the privacy of those users associated with the valid e-tickets. Therefore, the *research question 1* could be fully answered.

Research Question 2.

How to allow for privacy-preserving travel records processing in the back-end such that:

a) fine-grained billing for the registered tickets is possible;

b) customer identification is prevented.

The third building block of the suggested solution – path reconstruction – realized in the form of a custom pseudonymisation scheme (see Section 5.3.2) ensures that on the one hand, the back-end has enough information to perform fine-grained billing (possibly taking into account different pricing schemes and discounts) and on the other hand is prohibited from identifying customers associated with e-tickets. The employed pseudonymisation scheme essentially creates a privacy overlay enabling travel records processing in the back-end with the desired privacy properties. Consequently, an answer to the second research question is thereby provided.

Summarizing, it can be concluded that both research questions together with the core challenges defined at the beginning of the thesis could be answered by the developed privacy-preserving framework. Practical validation thereof was presented in the previous Section 6.2.

Chapter Summary

In this chapter, the evaluation of the developed privacy-preserving framework presented in Chapter 5 was provided. First, in Section 6.1 the conceptual evaluation was performed with respect to the satisfaction of the core requirements. Practical evaluation of the most critical part of the framework – front-end interaction – was discussed in Section 6.2. More specifically, the developed NFC prototype supporting the NFC-enabled smartphone as a user device was discussed and assessed in Section 6.2.1. The prototype for the smart card platform was presented in Section 6.2.2. After practical validation, the answers to the research questions defined at the beginning of the dissertation were provided in Section 6.3.

7 Summary and Outlook

In this thesis, the focus was made on privacy-preserving e-ticketing systems for public transport which are based on RFID and NFC technologies. Due to dynamic technological advance and primary focus being set on immediate functionality, the issues of privacy protection are often not properly addressed (if at all). Moreover, despite the existing privacy awareness in the European society, it is in the most cases still not sufficient to become a tangible factor providing enough motivation for companies to develop and roll out systems which are privacy-preserving from the outset. This dissertation aims at tackling the technological aspect of this issue in public transport scenario. In order to approach the problem in a holistic way, the necessary background on e-ticketing systems and underlying technologies (RFID and NFC) was provided in Chapter 2 together with the respective analysis and discussion. Chapter 3 then introduced the problem of privacy protection in the e-ticketing scenario by describing privacy from a technical perspective, providing an overview of generic threats and countermeasures as well as by analyzing the extent to which the notion of privacy is considered in the previously discussed RFID and NFC standards. The aforementioned overview and analysis have shown that additional substantial effort is required to provide for privacy protection in the domain of e-ticketing systems for public transport. This analysis paves the way to further discussion of core requirements posed to a privacy-preserving e-ticketing system which was presented in Chapter 3 along with the adopted attacker model. Having introduced the privacy problem in the target domain and discussed the requirements in Chapter 3, a detailed analysis of the related work was performed in Chapter 4. Prior to the related work review, the evaluation criteria were discussed which were essentially derived from the requirements and attacker model. Based on these evaluation criteria, the related work analysis was performed. As a result, it could be concluded that none of the reviewed approaches fully satisfies the requirements which motivates further research in this area. Our solution presented in Chapter 5 covers this issue. During the description of our approach, the general idea together with the main building blocks were discussed at first. Then the elaboration on our solution was presented with each building block being discussed in detail. Chapter 6 evaluated the suggested approach. First, a conceptual evaluation was presented. The practical part demonstrated feasibility of our solution by presenting two prototypes and assessing their performance. Each prototype addressed the use-case with a user device being represented by an RFID-based smart card and an NFC-enabled smartphone, respectively.

Main contribution of this work is, therefore, the development of a framework which allows to design privacy-preserving e-ticketing systems in public transport domain while preserving the core advantages of such systems, such as fine-grained billing support, flexible fare policies and transparent use. Moreover, this thesis provides the classification of privacy-preserving solutions developed so far together with the respective taxonomy which can be used for future research projects in this area. Furthermore, the created overview of different types of e-ticketing systems based on various classification parameters such as e-ticket types, the degree of user involvement, etc., is to the best of our knowledge currently one of the most holistic ones.

As a conclusion, it has to be mentioned that addressing privacy inevitably entails a certain degree of trade-offs between the achieved privacy protection and system functionality as well as between privacy and overall system cost. Moreover, the effort to achieve strong security with optimal efficiency is often prioritized over privacy protection resulting in secure but privacy-invasive systems. Therefore, a careful analysis together with a strong motivation for privacy are required to tackle this issue. This thesis, therefore, presents a solution where privacy is explicitly considered from the outset following the famous but unfortunately often overlooked privacy-by-design paradigm.

In future, it would be desirable to extend our work with usability tests, further optimize the developed prototypes and include more user-friendly GUIs into them. Moreover, our solution would benefit from further research on travel patterns mining and more efficient cryptographic primitives (and their implementations).

Bibliography

[1] Transport for NSW. Opal Card. http://www.transport.nsw.gov.au/content/opal-card, 2014. Accessed on 10.06.2014.

[2] Transport for London. Oyster Online. https://oyster.tfl.gov.uk/oyster/entry.do, 2012. Accessed on 30.10.2012.

[3] Transport for NSW. Touch & Travel. http://www.touchandtravel.de/, 2014. Accessed on 10.06.2014.

[4] The Near Field Communication Forum. http://www.nfc-forum.org/home/.

[5] Gildas Avoine. *Cryptography in Radio Frequency Identification and Fair Exchange Protocols*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2005.

[6] Flavio D. Garcia and Peter Rossum. Modeling Privacy for Off-Line RFID Systems. In Dieter Gollmann, Jean-Louis Lanet, and Julien Iguchi-Cartigny, editors, *Smart Card Research and Advanced Application*, volume 6035 of *Lecture Notes in Computer Science*, pages 194–208. Springer Berlin Heidelberg, 2010.

[7] Ruth Cassidy. NFC Forum Issues Specifications For Four Tag Types. http://www.nfc-forum.org/, 2007. The NFC Forum.

[8] Thomas S. Heydt-Benjamin, Hee-Jin Chae, Benessa Defend, and Kevin Fu. Privacy for Public Transportation. In *Proceedings of the 6th international conference on Privacy Enhancing Technologies*, PET'06, pages 1–19, Berlin, Heidelberg, 2006. Springer-Verlag.

[9] Foteini Baldimtsi, Gesine Hinterwalder, Andy Rupp, Anna Lysyanskaya, Christof Paar, and Wayne P. Burleson. Pay as you go. In *Workshop on hot topics in privacy enhancing technologies, HotPETSs 2012*. http://petsymposium.org/2012/papers/hotpets12-8-pay.pdf, 2012.

[10] Ahmad-Reza Sadeghi, Ivan Visconti, and Christian Wachsmann. User Privacy in Transport Systems Based on RFID E-Tickets. In *Workshop on Privacy in Location-Based Applications (PILBA 2008)*, volume 5283 of *Lecture Notes in Computer Sciences*. Springer-Verlag, October 2008. Malaga, Spain.

[11] EZ-Link Pte Ltd Co. EZ-Link Card System and Technology. http://ezlink.com.sg/about-ez-link/ez-link-card-system-and-technology, 2014. Accessed on 09.12.2014.

[12] São Paulo Transporte SA. SPTrans. http://www.sptrans.com.br/, 2013. Accessed on 09.12.2014.

[13] Ivan Gudymenko. A Privacy-Preserving E-Ticketing System for Public Transportation Supporting Fine-Granular Billing and Local Validation. In *Proceedings of the 7th International Conference on Security of Information and Networks, Glasgow, UK*, SIN '14, New York, NY, USA, 2014. ACM. Best paper award in section "Assuarance and Trust".

[14] Ivan Gudymenko, Felipe Sousa, and Stefan Köpsell. A Simple and Secure E-Ticketing System for Intelligent Public Transportation based on NFC. In *The First International Conference on IoT in Urban Space, Rome, Italy*, Urb-IoT, New York, NY, USA, 2014. ACM.

[15] Ivan Gudymenko. On Protection of the User's Privacy in Ubiquitous E-ticketing Systems Based on RFID and NFC Technologies. In *PECCS 2013 - Proceedings of the 3rd International Conference on Pervasive Embedded Computing and Communication Systems*. SciTePress, February 2013.

[16] Florian Kerschbaum, Hoon Wei Lim, and Ivan Gudymenko. Privacy-Preserving Billing for e-Ticketing Systems in Public Transportation. In *Proceedings of the 12th ACM workshop on privacy in the electronic society, WPES'2013* , WPES '13, pages 143–154, New York, NY, USA, July 2013. ACM.

[17] Xavier Bon, Jürgen Wehnert, Colin Tanner, Jae Lande, Richard Johnson, Ulrike Zeitler, Peter Stodart, Galit Mendelson, Elisabeth Doerner, Phil Sayeg, and Nigel Cullum. Smart Cards Move Onwards, A Survey. *Card Technology Today*, 15(10):12 – 15, 2003.

[18] Public transport – Interoperable Fare Management System – Part 1: Architecture (ISO 24014-1:2007). `http://www.iso.org/iso/catalogue_detail?csnumber=41985`, 2007.

[19] Eoghan McKenna, Ian Richardson, and Murray Thomson. Smart meter data: Balancing consumer privacy concerns with legitimate applications . *Energy Policy*, 41(0):807 – 814, 2012. Modeling Transport (Energy) Demand and Policies.

[20] M. Fahim Ferdous Khan, Yashiro Takeshi, Ito So, Masahiro Bessho, and Ken Sakamura. A Secure and Flexible Electronic-Ticket System. *Computer Software and Applications Conference, Annual International*, 1:421–426, 2009.

[21] Silvester Prakasam and Adeline Wang. Implementing Vehicle Location System for Public Buses in Singapore. *Journal of Institute of Engineers*, 44(vol. 2):103–110, 2004.

[22] G.Valdecasas Vilanova M.Eugenia. Anonymous and Untraceable Electronic Ticketing with Smart Card for Public Transport. Master's thesis, Universität Karlsruhe (TH), Institut für Algorithmen und Kognitive Systeme, 2002.

[23] OstalbMobil. OstalbMobil. Infos zu den Fahrscheinen. `http://www.ostalbmobil.de/fahrscheinarten/informationen.htm`, 2014. Accessed on 10.12.2014.

[24] Octopus Holdings Limited. Octopus Card System. `http://www.octopus.com.hk/home/en/index.html`, 2014. Accessed on 10.12.2014.

[25] Trans Link Systems (TLS) . OV-Chipkaart. `https://www.ov-chipkaart.nl/?taal=en`, 2014. Accessed on 10.12.2014.

[26] T. Gyger and O. Desjeux. EasyRide: Active Transponders for a Fare Collection System. *Micro, IEEE*, 21(6):36–42, dec 2001.

[27] Fraunhofer Institut für Verkehrs- und Infrastruktursysteme (IVI). ALLFA-Ticket: Elektronisches Fahrgeld-management für den öffentlichen Personennahverkehr. `http://www.ivi.fraunhofer.de/content/dam/ivi/de/documents/PB_ALLFA-Ticket_deut.pdf`, 2007. Project description.

[28] Klaus Finkenzeller. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication.* Third Edition. A John Wiley and Sons, Ltd., 2010.

[29] Japan Industrial Standard (JIS). JIS X 6319-4:2005. Specification of Implementation for Integrated Circuit(s) Cards – Part 4: High Speed Proximity Cards, 2005.

[30] ISO. ISO 14443 Standards family. Identification cards – Contactless integrated circuit cards – Proximity cards, 2008-2011.

[31] J.D. Day and H. Zimmermann. The OSI Reference Model. *Proceedings of the IEEE*, 71(12):1334 – 1340, dec. 1983.

[32] Vedat Coskun, Kerem Ok, and Busra Ozdenizci. *Near Field Communication (NFC) : From Theory to Practice.* John Wiley & Sons, 2011.

[33] Calypso Networks Association. Calypso Handbook. `http://www.calypsonet-asso.org/downloads/100324-CalypsoHandbook-11.pdf`, 2010.

[34] Sony Corporation. FeliCa Smartcard Technology. `http://www.sony.net/Products/felica/`. Accessed online on 17.12.2014.

[35] Cord Bartels, Harald Kelter, Rainer Oberweis, and Birger Rosenberg. TR 03126 - Technische Richtlinie für den sicheren RFID-Einsatz. TR 03126-1: Einsatzgebiet "eTicketing im öffentlichen Personenverkehr", 2009. Bundesamt für Sicherheit in der Informationstechnik, Deutschland.

[36] Till Ackermann. Der E-Ticket-Deutschland-Standard, die VDV-Kernapplikation: der Abschluss der Forschungsprojekte ist Startpunkt fuer das interoperable elektronische Fahrgeldmanagement/The e-ticketing standard of Germany: key application of the German VDV. *DER NAHVERKEHR*, 25(4), 2007.

[37] ISO/IEC. ISO/IEC 15693-1,2,3 Identification cards — Contactless integrated circuit(s) cards — Vicinity cards, 2000–2009.

[38] ISO/IEC. ISO/IEC 18092:2004(E): Information technology — Telecommunications and information exchange between systems — Near Field Communication — Interface and Protocol (NFCIP-1), 2004.

[39] J. Langer and M. Roland. *Anwendungen und Technik von Near Field Communication (NFC).* Springer-Verlag Berlin Heidelberg, 2010.

[40] ECMA International. Near Field Communication Interface and Protocol-2 (NFCIP-2). http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-352.pdf.

[41] Jérôme Pelé. NFC Technology from the IC to NFC Middleware. http://www.wima-nfc.com/NXP-Semiconductors/nxp_semiconductorsUK.php, 2008. Presentation at WIMA-2008.

[42] The NFC Forum. NFC Digital Protocol. Technical Specification, 2010.

[43] Gavin Shenker. Putting NFC Forum Specifications to Work. http://www.nfc-forum.org/resources/presentations/, 2012. The NFC Forum Presentations.

[44] The NFC Forum. NFC Activity. Technical Specification, 2010.

[45] The NFC Forum. Logical Link Control Protocol. Technical Specification, 2011.

[46] The NFC Forum. Simple NDEF Exchange Protocol. Technical Specification, 2011.

[47] The NFC Forum. NFC Data Exchange Format. Technical Specification, 2006.

[48] Jonathan Main. NFC Technology Overview. http://www.nfc-forum.org/resources/presentations/, 2009. The NFC Forum Presentations.

[49] The NFC Forum. Type 1-4 Tag Operation Specifications, 2011.

[50] The NFC Forum. Text Record Type Definition. Technical Specification, 2006.

[51] European Telecommunications Standards Institute. ETSI TS 102 613. Smart Cards. UICC - Contactless Front-end (CLF) Interface. Part 1: Physical and data link layer characteristics (Release 8), 2009.

[52] ECMA International. ECMA-373. Near Field Communication Wired Interface (NFC-WI), 2012.

[53] Gilles de Chantérac and Jean-Louis Graindorge. Focus Paper on Privacy in Transport IFM Applications. IFM Project, http://www.ifm-project.eu/fileadmin/WP2/Draft_Deliverable_2.2.pdf, March 2009. Draft Deliverable 2.2.

[54] Davide Zanetti, Pascal Sachs, and Srdjan Capkun. On the Practicality of UHF RFID Fingerprinting: How Real is the RFID Tracking Problem? In Simone Fischer-Hbner and Nicholas Hopper, editors, *Privacy Enhancing Technologies*, volume 6794 of *Lecture Notes in Computer Science*, pages 97–116. Springer Berlin / Heidelberg, 2011.

[55] Ari Juels and Ravikanth Pappu. Squealing Euros: Privacy Protection in RFID-Enabled Banknotes. In *Financial Cryptography '03*, pages 103–121. Springer-Verlag, 2002.

[56] Tong-Lee Lim, Tieyan Li, and Sze-Ling Yeo. Randomized Bit Encoding for Stronger Backward Channel Protection in RFID Systems. In *Proceedings of the 2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications*, PERCOM '08, pages 40–49, Washington, DC, USA, 2008. IEEE Computer Society.

[57] Wonjoon Choi and Byeong-hee Roh. Backward Channel Protection Method for RFID Security Schemes Based on Tree-Walking Algorithms. In Marina Gavrilova, Osvaldo Gervasi, Vipin Kumar, C. Tan, David Taniar, Antonio Laganá, Youngsong Mun, and Hyunseung Choo, editors, *Computational Science and Its Applications - ICCSA 2006*, volume 3983 of *Lecture Notes in Computer Science*, pages 279–287. Springer Berlin / Heidelberg, 2006.

[58] Tong-Lee Lim, Tieyan Li, and Sze-Ling Yeo. A Cross-layer Framework for Privacy Enhancement in RFID systems. *Pervasive and Mobile Computing*, 4(6):889 – 905, 2008.

[59] Ivan Gudymenko. Protection of the Users' Privacy in Ubiquitous RFID Systems. Master's thesis, Technische Universität Dresden, Faculty of Computer Science, December 2011.

[60] ISO. ISO/IEC 7816-4:2005. Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange. `http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=36134`, 2005.

[61] ECMA International. NFC-SEC. NFCIP-1 Security Services and Protocol. Cryptography Standard using ECDH and AES, 2008. White paper.

[62] ECMA International. NFC-SEC: NFCIP-1 Security Services and Protocol, 2010.

[63] ECMA International. NFC-SEC-01: NFC-SEC Cryptography Standard using ECDH and AES, 2010.

[64] Open NFC™ Project. `http://open-nfc.org/`, 2011. Accessed on 09.12.2014.

[65] RSA Laboratories. PKCS#15: Cryptographic Token Information Format Standard. `http://www.emc.com/emc-plus/rsa-labs/standards-initiatives/pkcs-15-cryptographic-token-information-format.htm`. Accessed on 09.12.2014.

[66] Open NFC™ Project. Open NFC™ Security Stack, 2011. Functional Specification.

[67] Sandeep Tamkar, Jan-Erik Ekberg, and N. Asokan. Identity Verification Schemes for Public Transport Ticketing with NFC Phones. In *STC'11*, Chicago, Illinois, USA, 2011. ACM.

[68] Andrea de Panizza et al. RFID: Prospects For Europe. Item-level Tagging And Public Transportation. Report, eur 24416 en, European Commission, JRC, 2010.

[69] Großraum-Verkehr Hannover. HANNOVERmobil. `http://www.gvh.de/hannovermobil.html?&L=1`, 2013. Accessed online on 19.04.2013.

[70] Phoenix Valley Metro. Platinum Pass Program. `http://www.valleymetro.org/employer_programs/platinum_pass`, 2013. Accessed online on 19.04.2013.

[71] Wiebren Jonge and Bart Jacobs. Privacy-Friendly Electronic Traffic Pricing via Commits. In Pierpaolo Degano, Joshua Guttman, and Fabio Martinelli, editors, *Formal Aspects in Security and Trust*, pages 143–161. Springer-Verlag, Berlin, Heidelberg, 2009.

[72] Mohamed Mezghani. Study on electronic ticketing in public transport. Technical report, European Metropolian Transport Authorities (EMTA), May 2008.

[73] Andreas Pfitzmann. Multilateral Security: Enabling Technologies and Their Evaluation. In Günter Müller, editor, *Emerging Trends in Information and Communication Security*, volume 3995 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin Heidelberg, 2006.

[74] Yves-Alexandre de Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. Unique in the Crowd: The privacy bounds of human mobility. *Nature srep.*, 3, 2013.

[75] Robert J. Reilly, Christopher W. Jenks, Gwen Chisholm-Smith, Eileen P. Delaney, and Hilary Freer. TCRP Report 115. Smartcard Interoperability Issues for the Transit Industry. Technical report, Acumen Building Enterprise, INC. Oakland, CA, 2006.

[76] Ari Juels. Minimalist Cryptography for Low-Cost RFID Tags (Extended Abstract). In Carlo Blundo and Stelvio Cimato, editors, *Security in Communication Networks*, volume 3352 of *Lecture Notes in Computer Science*, pages 149–164. Springer Berlin / Heidelberg, 2005.

[77] Tassos Dimitriou. A Lightweight RFID Protocol to protect against Traceability and Cloning attacks. In *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, SECURECOMM '05, pages 59–66, Washington, DC, USA, 2005. IEEE Computer Society.

[78] Jung Hee Cheon, Jeongdae Hong, and G. Tsudik. Reducing RFID reader load with the meet-in-the-middle strategy. *Communications and Networks, Journal of*, 14(1):10–14, 2012.

[79] Gildas Avoine and Philippe Oechslin. A Scalable and Provably Secure Hash-Based RFID Protocol. In *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops*, PERCOMW '05, pages 110–114, Washington, DC, USA, 2005. IEEE Computer Society.

[80] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. Cryptographic Approach to "Privacy-Friendly" Tags. In *In RFID Privacy Workshop*, 2003.

[81] Yasunobu Nohara, Sozo Inoue, and Hiroto Yasuura. A Secure High-Speed Identification Scheme for RFID Using Bloom Filters. In *Proceedings of the 2008 Third International Conference on Availability, Reliability and Security*, ARES '08, pages 717–722, Washington, DC, USA, 2008. IEEE Computer Society.

[82] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, jul 1970.

[83] Basel Alomair, Andrew Clark, Jorge Cuellar, and Radha Poovendran. Scalable RFID Systems: A Privacy-Preserving Protocol with Constant-Time Identification. *IEEE Trans. Parallel Distrib. Syst.*, 23(8):1536–1550, August 2012.

[84] David Molnar and David Wagner. Privacy and security in library RFID: issues, practices, and architectures. In *Proceedings of the 11th ACM conference on Computer and communications security*, CCS '04, pages 210–219, New York, NY, USA, 2004. ACM.

[85] Gildas Avoine, Etienne Dysli, and Philippe Oechslin. Reducing Time Complexity in RFID Systems. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography*, volume 3897 of *Lecture Notes in Computer Science*, pages 291–306. Springer Berlin Heidelberg, 2006.

[86] G. Avoine, M. Bingol, X. Carpent, and S. Yalcin. Privacy-friendly Authentication in RFID Systems: On Sub-linear Protocols based on Symmetric-key Cryptography. *Mobile Computing, IEEE Transactions on*, PP(99):1–1, 2012.

[87] Michael Hutter, Martin Feldhofer, and Johannes Wolkerstorfer. A Cryptographic Processor for Low-Resource Devices: Canning ECDSA and AES like Sardines. In Springer, editor, *Information Security Theory and Practices - WISTP 2011, 5th International Workshop, Heraklion, Greece, June 1-3, 2011, Proceedings.*, volume 6633 of *Lecture Notes in Computer Science*, pages 144 – 159, 2011.

[88] Sun Microsystems, Inc. *Application Programming Interface. Java Card^{TM} Platform, Version 2.2.2*, March 2006.

[89] Martin E. Hellman. A cryptanalytic time-memory trade-off. *IEEE Transactions on Information Theory*, 26(4):401–406, 1980.

[90] Philippe Oechslin. Making a faster cryptanalytic time-memory trade-off. In *CRYPTO*, pages 617–630, 2003.

[91] Boyeon Song and Chris J. Mitchell. Scalable RFID security protocols supporting tag ownership transfer. *Comput. Commun.*, 34(4):556–566, apr 2011.

[92] Boyeon Song and Chris J. Mitchell. RFID authentication protocol for low-cost tags. In *Proceedings of the first ACM conference on Wireless network security*, WiSec '08, pages 140–147, New York, NY, USA, 2008. ACM.

[93] Shaoying Cai, Yingjiu Li, Tieyan Li, and Robert H. Deng. Attacks and Improvements to an RIFD Mutual Authentication Protocol and Its Extensions. In *Proceedings of the second ACM conference on Wireless network security*, WiSec '09, pages 51–58, New York, NY, USA, 2009. ACM.

[94] Kun Peng and Feng Bao. A Secure RFID Ticket System for Public Transport. In Sara Foresti and Sushil Jajodia, editors, *Data and Applications Security and Privacy XXIV*, volume 6166 of *Lecture Notes in Computer Science*, pages 350–357. Springer Berlin Heidelberg, 2010.

[95] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer Berlin Heidelberg, 1999.

[96] Chiu Chiang Tan, Bo Sheng, and Qun Li. Severless Search and Authentication Protocols for RFID. In *PerCom 2007*, pages 3–12, 2007.

[97] Gildas Avoine, Cédric Lauradoux, and Tania Martin. When Compromised Readers Meet RFID. In Heung Youl Youm and Moti Yung, editors, *Information Security Applications*, volume 5932 of *Lecture Notes in Computer Science*, pages 36–50. Springer Berlin Heidelberg, 2009.

[98] Andy Rupp, Gesine Hinterwalder, Foteini Baldimtsi, and Christof Paar. P4R: Privacy-Preserving Pre-Payments with Refunds for Transportation Systems. In *Financial Cryptography and Data Security (FC)*, 2013.

[99] NXP Semiconductors. PUF – Physical Unclonable Functions. Protecting next-generation Smart Card ICs with SRAM-based PUFs. `http://www.nxp.com/documents/other/75017366.pdf`. Last access on 21.11.2014.

[100] G.E. Suh and S. Devadas. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, pages 9–14, June 2007.

[101] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul Syverson. Universal Re-encryption for Mixnets. In Tatsuaki Okamoto, editor, *Topics in Cryptology – CT-RSA 2004*, volume 2964 of *Lecture Notes in Computer Science*, pages 163–178. Springer Berlin Heidelberg, 2004.

[102] Giuseppe Ateniese, Jan Camenisch, and Breno de Medeiros. Untraceable RFID tags via insubvertible encryption. In *Proceedings of the 12th ACM conference on Computer and communications security*, CCS '05, pages 92–101, New York, NY, USA, 2005. ACM.

[103] Jaap-Henk Hoepman and George Huitema. Privacy Enhanced Fraud Resistant Road Pricing. In Jacques Berleur, MagdaDavid Hercheui, and LorenzM. Hilty, editors, *What Kind of Information Society? Governance, Virtuality, Surveillance, Sustainability, Resilience*, volume 328 of *IFIP Advances in Information and Communication Technology*, pages 202–213. Springer Berlin Heidelberg, 2010.

[104] Josep Balasch, Alfredo Rial, Carmela Troncoso, Bart Preneel, Ingrid Verbauwhede, and Christophe Geuens. PrETP: Privacy-preserving Electronic Toll Pricing. In *Proceedings of the 19th USENIX Conference on Security*, USENIX Security'10, pages 5–5, Berkeley, CA, USA, 2010. USENIX Association.

[105] Alfredo Rial and George Danezis. Privacy-preserving Smart Metering. In *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*, WPES '11, pages 49–60, New York, NY, USA, 2011. ACM.

[106] George Danezis, Markulf Kohlweiss, and Alfredo Rial. Differentially Private Billing with Rebates. In *Proceedings of the 13th International Conference on Information Hiding*, IH'11, pages 148–162, Berlin, Heidelberg, 2011. Springer-Verlag.

[107] E. Haselsteiner and K. Breitfuß. Security in Near Field Communication (NFC). Strengths and Weeknesses. In *Workshop on RFID Security 2006 (RFIDSec'06)*, Graz, Austria, 2006.

[108] David Chaum and Eugène Heyst. Group Signatures. In Donald Davies, editor, *Advances in Cryptology – EUROCRYPT '91*, volume 547 of *LNCS*, pages 257–265. Springer Berlin Heidelberg, 1991.

[109] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable Signatures. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 571–589. Springer Berlin Heidelberg, 2004.

[110] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, 1988.

[111] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.

[112] Jan Camenisch and Anna Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In *Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '02, pages 61–76, London, UK, UK, 2002. Springer-Verlag.

[113] PeterC. Johnson, Apu Kapadia, PatrickP. Tsang, and SeanW. Smith. Nymble: Anonymous IP-Address Blocking. In Nikita Borisov and Philippe Golle, editors, *Privacy Enhancing Technologies*, volume 4776 of *Lecture Notes in Computer Science*, pages 113–133. Springer Berlin Heidelberg, 2007.

[114] VDV Kernapplikations GmbH&Co.KG (VDV-KA KG). Spezifikation von Luftschnittstellen in einem VDV-Kernapplikations-konformen interoperablen Mobile Ticketing in Verbindung mit einer passiven Near Field Communication (NFC) Verkaufs- und Erfassungsinfrastruktur. `http://www.eticket-deutschland.de/spec-luka-nfc-1.0_1.pdfx`, April 2011.

[115] Chan Mo Lim. *Tactical Implementation Model for the Smart Card Payment System for Metro Operator.* PhD thesis, City University of Hong Kong, 2010.

[116] Jan Camenisch, Susan Hohenberger, Markulf Kohlweiss, Anna Lysyanskaya, and Mira Meyerovich. How to Win The Clonewars: Efficient Periodic n-times Anonymous Authentication. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS '06, pages 201–210, New York, NY, USA, 2006. ACM.

[117] Oded Goldreich. *Foundations of Cryptography: Volume I – Basic Tools.* Cambridge University Press, 2004.

[118] Tatsuaki Okamoto and Shigenori Uchiyama. A New Public-Key Cryptosystem as Secure as Factoring. In *Advances in Cryptology - EUROCRYPT '98, Espoo, Finland*, volume 1403 of *LNCS*, pages 308–318. Springer, 1998.

[119] TorbenPryds Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In Joan Feigenbaum, editor, *Advances in Cryptology — CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer Berlin Heidelberg, 1992.

[120] Latanya Sweeney. k-anonymity: A Model for Protecting Privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, October 2002.

[121] Joseph Lutgen. The Security Infrastructure of the German Core Application in Public Transportation. In *ISSE/SECURE 2007 Securing Electronic Business Processes*, pages 411–418. Vieweg, 2007.

[122] Bundesministerium des Inneren. Der neue Personalausweis. `http://www.personalausweisportal.de/DE/Home/home_node.html`.

[123] Jens Bender, Dennis Kügler, Marian Margraf, and Ingo Naumann. Privacy-friendly Revocation Management without Unique Chip Identifiers for the German National ID Card. *Computer Fraud & Security*, September 2010.

[124] BSI. Technical Guideline TR-03110-3. Advanced Security Mechanisms for Machine Readable Travel Documents – Part 3 – Common Specifications, Version 2.10, March 2012. Bundesamt für Sicherheit in der Informationstechnik, Deutschland.

[125] Jan Camenisch and Anna Lysyanskaya. A Signature Scheme with Efficient Protocols. In Stelvio Cimato, Giuseppe Persiano, and Clemente Galdi, editors, *Security in Communication Networks*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer Berlin Heidelberg, 2003.

[126] Andreas Poller, Ulrich Waldmann, Sven Vowé, and Sven Türpe. Electronic Identity Cards for User Authentication – Promise and Practice. *Security Privacy, IEEE*, 10(1):46–54, jan.-feb. 2012.

[127] IHS Inc. NFC-Enabled Cellphone Shipments to Soar Fourfold in Next Five Years. `http://press.ihs.com/press-release/design-supply-chain/nfc-enabled-cellphone-shipments-soar-fourfold-next-five-years`, February 2014. Last accessed on 07.11.2014.

[128] libnfc community. Libnfc: Official wiki. `http://nfc-tools.org/index.php`. Accessed on 12.06.2014.

[129] Manuel Weißbach. Entwicklung und Implementierung eines Konzepts zur interaktiven Datenübertragung zwischen einem Android-basierten, NFC-fähigen Smartphone und einem computergesteuerten NFC-Lesegerät. TU Dresden, Faculty of Computer Science, June 2014. Großer Beleg, supervised by Ivan Gudymenko and Katrin Borcea-Pfitzmann.

[130] CyanogenMod. `http://www.cyanogenmod.org/`, 2014. Last accessed on 10.11.2014.

[131] C. Saminger et al. An NFC Ticketing System with A New Approach of An Inverse Reader Mode. In *5th International Workshop on NFC*, pages 1–5, Feb 2013.

[132] Pavel Plakhin. Privacy-preserving eTicketing-Validierung basierend auf NFC. TU Dresden, Faculty of Computer Science, Mai 2014. Diplomarbeit, supervised by Ivan Gudymenko and Katrin Borcea-Pfitzmann.

[133] Bundesamt für Sicherheit in der Informationstechnik (BSI). Entwurf zum Algorithmenkatalog 2015. `https://www.bsi.bund.de/DE/Themen/weitereThemen/ElektronischeSignatur/TechnischeRealisierung/Kryptoalgorithmen/kryptoalg.html`, October 2014.

[134] Klaus Potzmader, Johannes Winter, Daniel Hein, Christian Hanser, Peter Teufl, and Liqun Chen. Group Signatures on Mobile Devices: Practical Experiences. In Michael Huth, N. Asokan, Srdjan Čapkun, Ivan Flechais, and Lizzie Coles-Kemp, editors, *Trust and Trustworthy Computing*, volume 7904 of *Lecture Notes in Computer Science*, pages 47–64. Springer Berlin Heidelberg, 2013.

[135] Oracle®. Java card platform specification 2.2.2. `http://www.oracle.com/technetwork/java/javacard/specs-138637.html`. Last accessed on 17.11.2014.

[136] Multos® platform. `http://www.multos.com/`. Last accessed on 17.11.2014.

[137] Pim Vullers and Gergely Alpár. Efficient Selective Disclosure on Smart Cards Using Idemix. In Simone Fischer-Hübner, Elisabeth de Leeuw, and Chris Mitchell, editors, *Policies and Research in Identity Management*, volume 396 of *IFIP Advances in Information and Communication Technology*, pages 53–67. Springer Berlin Heidelberg, 2013.

[138] Patrik Bichsel, Jan Camenisch, Thomas Groß, and Victor Shoup. Anonymous credentials on a standard java card. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09, pages 600–610, New York, NY, USA, 2009. ACM.

[139] Axel Engelmann. Kryptografische Methoden auf einer Javacard. TU Dresden, Faculty of Computer Science, September 2012. Praktikum, supervised by Ivan Gudymenko.

[140] Álvaro Giménez Serrano. Privacy-Preserving E-Ticket Validation for Public Transportation Systems Based on RFID/NFC Technologies. Master's thesis, TU Dresden, Faculty of Computer Science, September 2013. Supervised by Ivan Gudymenko and Stefan Köpsell.

[141] Marlon Baeten. Improving smart grid security using smart cards. Master's thesis, Radboud University Nijmegen, Faculty of Computer Science, August 2014.