A Privacy Preserving ECommerce Oriented Identity Management Architecture

Elli Androulaki

Thesis Mentor: Steven Bellovin

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

May 2011

ABSTRACT

A Privacy Preserving ECommerce Oriented Identity Management Architecture

Elli Androulaki

We discuss the construction of a deployable and privacy-preserving identity management architecture addressing all aspects of electronic commerce using the existing privacy preserving cryptographic primitives, while at the same time guaranteeing compatibility with current business models. Absolute privacy, which in our case is consumers privacy, is defined as the combination of consumers anonymity and their transaction unlinkability; this is very commonly violated in today's online eCommerce world. It is apparent that, because of the monetary nature of most popular online activities, accountability is a prerequisite for every applicable privacy-enhancing mechanism. We present an architecture which addresses privacy issues raised in all aspects of eCommerce, including online advertising, online payments, delivery of online purchases, and merchant-buyer evaluation systems, and aims to prevent any unauthorized entity from building and distributing consumers profiles or tracing their transactions. In addition, as online transactions can affect consumers credit scores, and are strongly associated with consumers normal activities such as bank account management and taxation, we extended our privacy-preserving protocols to a card-based identity management architecture; this deals with many types of financial activities and consumers attributes. Card-loss related mechanisms, such as advanced card-owner authentication techniques, privacy-preserving card-content recovery, and automatic and recursive credentials invalidation are also addressed in our work. It is noteworthy that privacy is incorporated in our protocols as an option, i.e, it is guaranteed only if the individual choses to; the consumer always has the option of using the existing non-privacy preserving methods.

The combination of privacy with accountability and deployability to achieve proper operation of such a variety of user activities in a centralized manner constitutes both the main

innovation and contribution of this work. Accountability is a critical requirement in all monetary eCommerce activities, while deployability is a prerequisite for protocols applicability. We consider deployability in three ways: (a) in our attack model, for which we make real world assumptions, (b) in the designed architecture, where we did not introduce changes in current systems structure, and (c) by integrating in our protocols useful properties that are currently supported, while incentivizing our protocols' application by offering monetary benefits to most system entities.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First of all I would like to express my deepest gratitude to Dr. Steven Bellovin, Professor of Computer Science Department of Columbia University, who has been my supervisor since the beginning of my study. He provided me with many helpful suggestions, important advice and constant encouragement during the course of this work.

Special gratitude is due to Professors Tal Malkin and Sal Stolfo, Computer Science Department of Columbia University, for taking intense academic interest in this study, provided valuable suggestions and useful discussions that improved its quality.

I wish to express my cordial appreciation and sincere thanks to Dr. Moti Yung, and to Prof. Jonathan Smith, Computer Science Department of University of Pensilvania, for devoting their precious time and making many valuable suggestions which indeed helped improve this thesis.

I also wish to express my sincere appreciation to Prof. Angelos Keromytis, Computer Science Department of Columbia University, who being my co-advisor at the beginning of my study, made many valuable suggestions and always gave constructive advice.

Many special thanks go to my friends (alphabetically) Alexandros Iliadis, Georgios Mol and Angelika Zavou, who supported me a lot and made my staying in New York an unforgettable and constructive experience.

FInally, I would like to express my heartiest thanks to my parents Katerina and Merkourios Androulakis and my brother, Dimitrios Androulakis, for their invaluable support and encouragement throughout this course.

To my parents, Katerina and Merkourios, and to my brother Dimitrios

Chapter 1

Introduction

Increasingly, the online and offline worlds are converging. Not only do they rely on one another; requirements for one often become requirements for the other. As internet usage continues to grow into new areas, i.e., banking and commerce, security and privacy become two concepts of growing importance. Although not yet accurately defined in an environment with such a broad range of activities, security becomes crucial for dealing with identity theft and data confidentiality while privacy is a fundamental right of all individuals. Combining the two is not an easy task, primarily because of the contradictory nature of their application mechanisms. In this thesis, we aim to define the notions of both terms and combine them in a centralized identity management architecture which expands to most online activities of users and especially eCommerce.

Privacy is generally defined as the fundamental right of the individual to determine the degree to which he will interact and share information with his environment and is in multiple ways violated in online procedures. More specifically, eCommerce — consisting of online advertising, online payment systems, product delivery services, all enhanced with risk management mechanisms — raises many privacy concerns:

- *Advertising: Targeted Advertising.* To utilize internet originated user information for increasing their ad-banners' effectiveness, *publishers* — usually service oriented websites, which are paid to add advertising spots of other companies — would choose their ads based on a user's browsing activity. Various publishers may combine the user infor-

mation they possess to make profiles more accurate[Krishnamurthy and Wills, 2006]. Targeted advertising constitutes a serious privacy violation as the constructed profiles reveal sometimes sensitive user information without the latter being able to control it, while the privacy breach becomes bigger when online credit card purchases take place and a profile is linked to a particular identity.

- *Online Payment: Credit Cards.* Online Payment Systems are ultimately, with very few exceptions, credit card based. Being closely related to their owners' identities, credit cards' extended use constitutes a serious threat to consumers' privacy. Frequent occurrences of credit card losses, credit card number based-impersonation attacks as well as human nature errors, i.e. overcharge of a client, make it necessary for cardholders to be able to monitor their own transaction activity and for merchants to provide banks with detailed description of each credit card transaction. Under the umbrella of the need of immediate charge justification/correction, each bank, which is no more trusted than the people operating it, acquires a global view of its customers' transaction activity.
- *Product Delivery Systems.* Privacy concerns in this case derive primarily from the information a delivery company acquires from the merchant. Given the (usually) long-term monetary relationship between the two, the delivery company knows the type of products the merchants sell, the name and shipping address of the person the product is for and the exact object shipped, if it is fragile or of great value. In addition, since it is likely that the same delivery company serves a variety of other websites, the former may obtain a very good approximation of the transaction profile of customers who often make purchases online.
- *Risk Management.* ECommerce risk management refers to fraud prevention techniques protecting merchant from a dishonest consumer (credit risk) and consumer from a dishonest merchant (merchant rating systems) or another consumer (credit risk). Many credit risk management techniques performed by the banks, address the first, while various types of transaction evaluation systems, e.g., reputation systems, address the second case. Currently, the credibility of both systems is strongly connected to traceability of consumers' activity, which complicates the application of privacy in eCommerce procedures.

Security seems to enhance the complexity of privacy application. Aiming to provide a combination of confidentiality, integrity and availability of the exchanged data, *security* is a strong requirement for all monetary online activities: lack of data integrity or confidentiality in online activities such as bank account management or medical record logins would have serious ramifications both in finance and integrity of personal data, which may lead to identity theft. Strong authentication and accountability seem to constitute a powerful measure towards individuals' protection against any type of identity theft. Authentication means that each individual has a provable identity, while accountability requires that misbehaving individuals are identified. It is, thus, becoming increasingly clear that we will move towards mandatory strong authentication as a means to securing online interactions for critical cyber infrastructures [members, 2008]. Centralization is implied which, by its nature threatens individual users' privacy, as it enables a degree of surveillance.

In this thesis, we explore issues related to the combination of the two properties security (authentication, accountability) and privacy, in the online world. More specifically, we aim to offer the *option* of privacy to the degree to which it can coexist with accountability in real world, for a big range of users' online activities, i.e., online banking, payments, etc. [1] We respond to the security-need for centralization by embodying an anonymous and master-secret based credential system, which constitutes the core authentication mechanism in our architecture; we build the rest of the services around it: each individual is ultimately identified online through a single secret stored in a card, which authenticates the former multiple times anonymously and unlinkably and authorizes him for most of his activities. More specifically, there is a single digital identity card per individual which can be used in addition or in substitution to the existing offline authentication credentials, i.e., identity cards, passports, etc. Using his card and the secret connected to it, a user may obtain memberships to various organizations, ranging from banks to commercial websites, and build a tree of unlinkable credentials ultimately connected to his base one.

Unlike many existing anonymous credential systems, user-blacklistability and card-memberships recoverability are supported. Depending on the type of the organization a

[1] As regarded as a *right* of individuals, it is important to note that we target at privacy being offered as an option and not as a non-disposable property of individuals.

user is interacting with and despite cards' untraceability, a card may be globally or locally blacklisted if its owner misbehaves, e.g., a bank may globally add a user to the global list of bad creditors, while a blog-administrator may add a user to a local blacklist if the latter does not adjust to that blog's policy. If lost, a card can be deactivated while its legal owner *only* may recover the full set of activities he has been involved in using his card. His credentials may then be updated accordingly, in a way so that user's activity is not revealed to unauthorized individuals.

In this thesis, we provide a template for the interactions of a user, i.e., a holder of a card (and its secret) with non-commercial websites, while we present in detail more specific privacy-preserving protocols for monetary user-activities in online banking and online transactions. It is apparent that accountability is crucial in both cases. In terms of functionality there are four important sections in the recommended architecture:

- *A card-based authentication mechanism* which realizes the anonymous credential system we described before and allows the users to manage in a privacy-preserving, nevertheless accountable, fashion their multiple identities across their online activities (see chapter 5).

- *Online banking protocols*, which take place between the bank and an identified individual, who has already registered to the bank. They involve mechanisms for an authorized user to open and manage anonymous accounts, to be fairly taxed, to issue credit cards whose activity cannot be traced by any individual or the bank, while enhanced with credit score update mechanisms. It is crucial to note that the privacy provided here is conditional on proper individual's behavior. Users who attempt to take advantage of their "anonymity" to spend more than their accounts' balance, lie for their taxes or use another user's account, will be identified. Measures are taken so that anonymous accounts' ownership can be revealed in emergency situations.

- *Online transactions' protocols,* where we address privacy issues in three separate operations: transaction payment, product delivery and transaction evaluation. Privacy preserving online payments are introduced in chapter 9 as part of the online banking set of protocols. More specifically, we present an anonymous — but accountable —

physical object delivery system for online purchases, where the recipient of a package remains anonymous towards both the delivery company and its mail stations, while (if applicable) he maintains his anonymity towards the merchant (see section 8.1) and the collaborations of the latter with the delivery company.

- *Complementary privacy mechanisms* As the leak of information through browsing cannot be effectively avoided by existing means, and since targeted ads seem to be widespread throughout all user online activities, we support our protocols with mechanisms that deal with both of these issues. In particular, we incentivize participation in existing anonymizing networks to make their anonymity provisions stronger, while we introduce a targeted ad mechanism which protects consumer's privacy without deactivating existing click fraud detection technologies.

There are many innovations in this work but the main ones may be summarized in the following two: the incorporation of real world in the requirements, threat model and solutions of privacy issues raised in users' monetary activities, and the variety of the user-activities addressed in a privacy-preserving and centralized architecture. Both the wide range of activities covered and the real world assumptions imply the need for defining, applying and combining different levels of privacy — corresponding to various types of activity — in a unite architecture, which substantially differentiates this work from previous in the field of privacy.

In many cases, real world assumptions has lead us to solving existing problems with extended requirements. The concept of anonymous master-secret based credentials has rather been introduced by Brands, Camenish and Lysyanskaya in the past. However, they lack deployability properties, such as complete recoverability of the functionality of a lost card, privacy preserving, recursive blacklistability of all the credentials connected to compromised card. In a similar way, although anonymous payments have been introduced in the past by Chaum[Chaum *et al.*, 1988b], they only operated in prepaid rather than in credit fashion. Thus, existing anonymous payments lacked the credit systems' advantages. Computational efficiency and Incentivising participation in anonymizing networks through payments has also been suggested in the past. However, the existing systems did not consider efficiency and accountability important, which is among our main focuses.

On the other hand, it is conceivable how the interconnection among the different operations of our architecture introduced privacy issues that not been investigated before. Evidently, to completely apply and maintain transactions' privacy it is important to address taxation and product delivery, while for the security of transactions, privacy-preserving evaluation methods had to be developed. We, thus, constructed privacy preserving protocols to achieve fair taxation of individuals' anonymous accounts, fair credit score calculation, fair evaluation of transactions and accountable delivery of online purchases.

In the following chapters we present in detail our requirements, building blocks and achievements. More specifically, we initially specify the requirements and restrictions implied by the conflicting goals of user privacy and security in the face of a *deployable* card-based identity management architecture that addresses many financial activities ranging from online purchases to taxation and employment. In particular,

- we define consumer's privacy and security in online activities against "real world" adversarial models and discuss at what degree these concepts can coexist in a unite and multilateral architecture
- we consider deployability in three ways: (a) w.r.t. our attack model, for which we make real world assumptions, (b) w.r.t. the designed architecture, where we did not introduce considerable changes in existing structures and (c) w.r.t. the functionalities offered, as we preserve useful properties that current non-privacy-preserving systems support, while we incentivize its application by offering realistic (monetary) benefits to most system entities.

Based on the aforementioned general requirements, we make use of the secure ecash schemes, credential systems and other crypto primitives which are known to provide (un)conditional anonymity, to construct privacy preserving and deployable protocols

- for user-authentication and revocable authorization in multiple online services, i.e., online banking, employment, taxation etc.
- for all fields of eCommerce, providing similar functionalities with current systems ranging from online browsing to transaction payment, risk management and product delivery.

We, then, combine all the aforementioned protocols, in a single *deployable, accountable* and privacy-preserving eCommerce-wise identity management architecture which may not substantially degrade existing systems' performance. It is important to note that although the aforementioned privacy-preserving mechanisms are presented as parts of a single and unite architecture, each of them can be be applied independently to provide privacy against a *less global* attacker.

Part I

Problem Description

Chapter 2

Motivation: Privacy vs. Security

In this chapter we will focus on privacy issues invoked via online activities of individuals, which constitutes the core motivation of this work. We also discuss extensively the online security-privacy contradictions, which unavoidably posit this work especially challenging. More specifically, we provide a map of most common online activities of individuals and describe each of them separately, while exhibiting how these activities interconnect in the face of their operator. Considering privacy as equivalent to individual's activity untraceability, and security as the combination of accountabillity and fairness, we demonstrate how currently privacy is compromised in each case and emphasize on specific security constraints that derive from applying privacy-mechanisms.

2.1 Current Online Activities: A user-centralized System

In most communities, individuals have a single identity, register with a public authority, and — depending on their family or financial status, family, age etc. — obtain government-issued credentials. These credentials can be used by the individuals to participate in a number of real-world *interactions*, which may include — without being limited to — the most important (id-based) activities of an individual, such as handling of employment, management of bank accounts, fair and accurate income tax reporting, fair credit score update, verification of specific attributes of the individual — for example, that the individual is above the drinking-age limit — and registration in multiple online or offline clubs,

associations or services. We may, thus, identify the following entities:

- *Users*, who may interact with other users or organizations in order to perform various tasks. Users represent the citizens and other residents of a country.
- A *Registration Authority* (RA), which is responsible for registering the legal users and manages the construction, modification, and destruction of government-issued users' credentials. Given the fact that users represent a country's citizens, RA may represent the official citizens' registry, i.e., Social security office for USA.
- *Banks*, who allow users to open (possibly pseudonymous) accounts for the purpose of storing cash and handling financial transactions. They are responsible for reporting interest for income tax purposes.
- A *Tax Authority* (TA), which is responsible for ensuring that correct income taxes are paid by all users.
- *Employers*, who form employment relationships with users and are responsible for reporting income and corresponding tax withholding. Employers may be any type of real-world employers.
- *Non-financial organizations*, who may wish to extend membership to users and are not responsible for tax reporting. Such organizations may be hospitals, gym centers, schools, any electronic commerce oriented website, etc.

Fig.2.1 depicts in high level the current interactions between the entities, showing the series of transactions in current identification systems. Citizens collaborate with an authority similar to RA to issue a national identity card, which they use later on to open accounts in banks, to be employed and receive their payments, and to prove their age. However, we can see that in online communications — and if payment is guaranteed wherever required — proof of identity is not required (i.e., to obtain membership to a website). In this way, users suffer no consequence if behaving dishonestly, i.e., if they visit a website access or manipulate online information for which they are not authorized.

Constituting a large percentage of users' online activities, electronic commerce (eCommerce), can be defined as any type of commercial transaction, that involves information flow

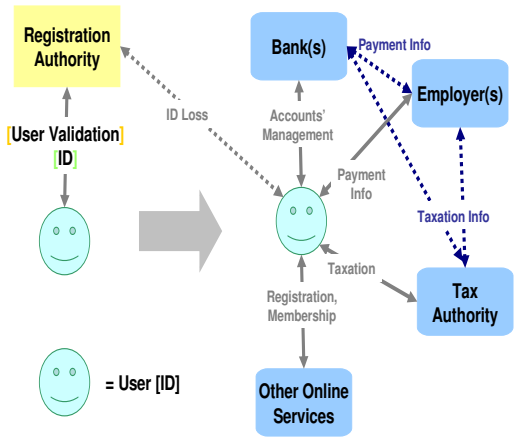

Figure 2.1: Chain of procedures in real world.

across the internet. In particular, eCommerce refers to the the purchasing, selling, and exchanging of goods and services over the internet through which transactions are performed electronically. In fact, according to a governmental eCommerce classification [Andministration,], there are four eCommerce types: (a) B2B (Business-to- Business), which refers to companies doing business with each other such as manufacturers selling to distributors, (b) B2C (Business-to-Consumer), which are "virtual storefronts" on websites with online catalogs, sometimes gathered into a "virtual mall" for the user to visit, see the various products available and make orders, (c) C2B (Consumer-to-Business), where consumers post service requests or particular projects with the budget available for companies to bid on, and (d) C2C (Consumer-to-Consumer), where consumers participate in various online marketplaces to sell or buy products one from the other. Our work aims mainly the B2C part of eCommerce, while it may require small modifications to be applied to C2C case.

In the general case of a B2C eCommerce system we identify the *merchant* and the *consumer*. The merchant is basically the website demonstrating the products on sale, while equipped with online payment systems to accept payments. The consumer or user or customer, is the person making online purchases.

In an informal breakdown of an eCommerce system, we may identify the following sections:

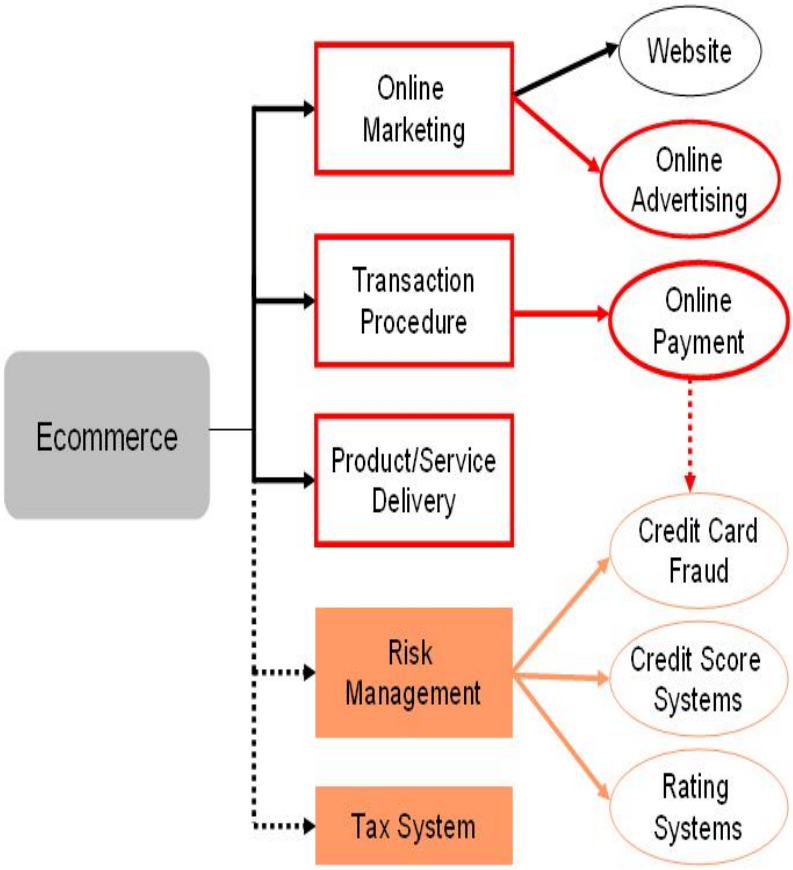

Figure 2.2: Chain of procedures in eCommerce.

1. **Online Marketing**, which consists of the entire campaign for the promotion of a product/service. Except for *users* — the online consumers — in a typical advertising mechanism, the principle parties are *advertisers*, *ad networks* and the *publishers*. *Advertisers* are the companies selling and promoting a particular product or group of products. *Publishers* are usually service- oriented websites paid to *publish* advertisements of advertisers' products. *Ad networks* are paid by advertisers to choose the list of advertisements which will appear on publishers and filter the clicks the ads receive. Typical examples of ad-networks are Doubleclick (owned by Google), Atlas Solutions

(owned by Microsoft), Brightcove, and more. It is often the case that an ad network offers various services and also acts as a publisher. Publishers and ad-networks may be organized in different layers and are paid in accordance to the advertisements' effectiveness.

2. **Online Transaction**, which consists of the product selection on customer's part as well as the payment for it. Most existing online payment systems are credit card based and are usually operated through *gateways*, which are special software systems in constant collaboration with banks, i.e., consumer's *card issuing bank* (CIB) and merchant's *acquiring bank* (AB), so that users' accounts are automatically checked and updated.

3. **Product delivery**, which is currently performed through various delivery companies in agreement with the merchant. Product delivery systems consist of *delivery companies* (DC) and *mail stations*(MS). Delivery companies are the courier companies paid by a merchant to deliver the product to an address specified by the customer. Mail stations receive and forward the mail traffic according to their destination. Although they may be affiliated with the delivery companies, mail stations constitute autonomous entities.

4. **Risk Management** refers to mechanisms which target to protect a customer towards malicious merchants and merchants, banks and honest consumers towards malicious customers: a malicious merchant may try to deceive a client into making him pay more, not deliver the purchased product or "steal" client's information (identity theft case); a malicious consumer may try to avoid payment to the merchant or to the bank or attempt to frame another consumer for paying for purchases the latter did not make. Based on the relationships they protect, eCommerce context-ed *risk* can be classified into two categories:

   - *credit risk*, which refers to the protection of bank towards cardholders who do not pay their debts or cardholders being framed by unrelated parties; it is currently addressed through credit card fraudulent purchases detection techniques, credit score calculation mechanism.

- *transaction risk*, which refers to protection of the parties involved in a transaction; it is currently addressed through various merchant/consumer rating systems. Credit score mechanisms can be also classified as transaction risk related.

Money Laundering is another risk we consider the state has to take w.r.t. consumers. However, we do not deal with it in this work.

2.2 Concerns and Restrictions: Privacy vs. Security

Privacy violation taking place in common monetary activities and the corresponding user authentication and authorization operations is the principal motivation of this work. Unfortunately, application of privacy tends to impede security. Roughly defining security as the combination of accountability, fairness, and privacy as the fundamental right of an individual to control the degree to which he shares information with its environment [Shirey, 2000], we emphasize on how critical the aforementioned concepts are in the online world and in what ways they contradict each other. ECommerce activities are indicative of our claim:

2.2.1 Online Marketing.

Targeting advertising method has become very popular in current online advertising systems and constitutes the most serious privacy concern raised in online marketing. More specifically, to increase the efficiency of the advertisements they "publish" — and, thus, their revenue — publishers (and ad-networks) usually profile users who visit their website or use their services and adjust ads they present to these profiles. The information flow regarding users' profile is depicted in fig. 2.3. The profiles are technically constructed through cookies which enable the accumulation of information regarding user browsing or searching activity across various websites. As illustrated in fig. 2.3, when a user visits a website (publisher), the browser sends to the publisher some pieces of information called cookies, which link multiple visits of the same user. A special type of cookies, the *third party cookies*, are sent during the publishers' visit to the corresponding ad-networks, who can now trace a user's browsing activity across multiple websites. In this way, especially

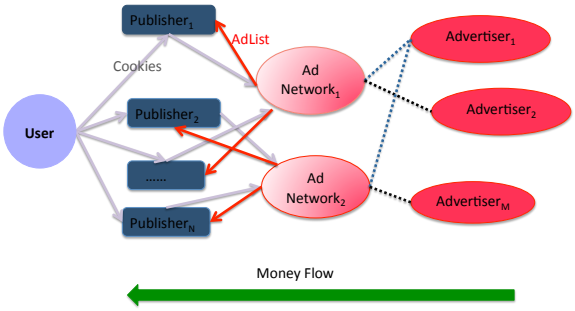

Figure 2.3: Targeted Advertising Mechanism.

as they collaborate with many publishers, ad-networks construct very accurate user profiles and target ads accordingly. There are many policies regarding how ad-networks and publishers are paid. The most popular one is the "cost per click" (CPC), where both parties are paid by the advertisers in proportion to the number of clicks the latters' ads receive.

It is conceivable how such profiling violates individual's privacy. First of all, since most people don't reset their cookies regularly, publishers maintain a particular client's purchase profile for a long time even if the last has not recently made a purchase. To make matters worse, when client clicks on an ad, publishers often send client's profile to the corresponding advertiser such that the latter further provides more profile-specific ads. Such collaborations constitute a serious privacy compromise, as the user has no control of where his browsing activity profile is spread. This becomes even more crucial when a user decides to make a purchase from the advertiser's website using his credit card. In this case, his purchase profile will be directly linked to the owner of the credit card, while the advertiser knows much more information for the person making the transaction than just the payment related details.

The monetary dependence between the parties involved (see fig. 2.4), in targeted advertising makes the problem even more difficult to deal with. For a candidate solution to be adopted in real systems, it should counter offer privacy-preserving advertising techniques

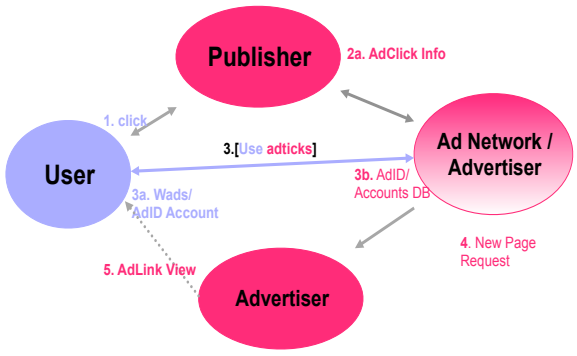

Figure 2.4: Targeted Advertising money flow.

which guarantee similar revenues to all entities involved.

2.2.2 Online Payments

Credit card payments tend to be very popular in online purchases, since they have many useful properties. Apart permitting delayed payment, credit cards provide users with logs of their own transactions, receipts, and the opportunity to challenge and correct erroneous charges. However, being closely related to their owners' identities, credit cards constitute a serious threat to consumers' privacy. Frequent occurrences of credit card losses, credit card number based-impersonation attacks as well as human nature errors, i.e. overcharge of a client, make it necessary for cardholders to be able to monitor their own transaction activity and for merchants to provide banks with detailed description of each credit card transaction. Under the umbrella of the need of immediate charge justification/correction, each bank, which is no more trusted than the people operating it, acquires a global view of its customers' transaction activity.

None of the currently deployed credit card systems offer consumer's privacy towards banks. Given the fact that the percentage of credit card-based purchases is increasing, a

deployable privacy preserving credit card system has become quite important. At the same time accountability and risk management requirements posit the same problem particularly complicated.

2.2.3 Product Delivery

As noted, the delivery company is usually under contract with the merchant. Given the (usually) long-term monetary relationship between the two, the delivery company knows: (a) the type of products the merchants sell, (b) the name and shipping address of the person the product is for, who may may or may not be the one who bought the product, and, (c) the exact object shipped, if it is fragile or of great value.

Certainly, the courier company knows the person to whom the product is delivered, as well as the type of the product. In addition, since it is likely that the same delivery company serves a variety of other websites, the former may obtain a very good approximation of the transaction profile of customers who often make purchases online.

2.2.4 Risk Management

Because of the large scale of online customers and merchants, developing credible online reputation systems, credit card risk management services and online money laundering prevention mechanisms are considered to be critical.

- **Reputation Systems** Reliability is strongly connected to credibility: trustworthy credentials for merchant's services or client's purchase behavior are important for deciding whether to commit in a transaction with either of them. However, absolute trustworthiness to a recommendation, derives from the name of the person signing it, which itself — depending on the product/service purchased — has many privacy implications: nominal recommendations at the end of a transaction would reveal the identities of participants, endangering their privacy. There is thus a strong need to combine anonymity and credibility in a pre-knowledge–based (recommendation or reputation) system, where

  - the parties to be involved in a transaction know nothing more than the pre-knowledge value (reputation) of the other party,

  - the pre-knowledge value cannot be reset or faked for by a user,
  - the pre-knowledge value should be representative of the corresponding user's purchase behavior.

- **Credit risk** Existing credit risk management mechanisms base their performance on the absolute control banks have on consumers' transactions: the more accurate picture the credit bureau acquires for consumers' transactions, the more efficiently may the latter detect a mischarge coming from a malicious party; on the other hand customers' credit score may also be calculated more accurately.

2.2.5 Taxes

Taxes is another aspect of commercial activity of individuals, which posits restrictions against privacy. Fairness in taxation tends to be proportional to the details regarding the individual's income the taxation authority acquires. Taxation of bank accounts is even more challenging as banks report to the tax authority the exact taxes withheld by each account, revealing in this way users' financial information. It is apparent how any act towards anonymization of bank accounts would impede fairness in taxation procedure. A privacy preserving and accountable taxation mechanism, thus, constitutes a challenge.

2.2.6 Online Security: Authentication, Accountability

Access control tends to be strictly connected to security. Defined as the combination of integrity, confidentiality and availability of data exchanged over the internet, security, is now an online concern of growing importance. As internet usage continues to spread into new areas, lack of data integrity or confidentiality in online activities such as bank account management or medical record logins would have serious ramifications both in finance and integrity of personal data, which may may lead to identity theft. Strong authentication and accountability seem to be the key properties of a system aiming to achieve security.

Authentication means that each individual has provable identity and been used a lot in the past to provide security: people need to authenticate themselves both to computer systems and to other people or organizations. While there are definitely challenges [Kent and Millett,

2003], there is sufficient commonality that some people have sought to combine the two. In particular, the suggestion has been made that a single credential could serve as a login token — "something you have" — and as a national identity document as well. Either alone is difficult (see [Kent and Millett, 2002] for a discussion of some of the issues with national identity documents); combining the two complicates the issue even more. Nevertheless, many countries already have such documents; some of those that do not, such as the United States [Hsu,] and the United Kingdom [BBC, 2009], are considering introducing them.

In the online world, the advantages of token-based authentication to computer systems have long been known. In fact, a recent major report *Securing Cyberspace for the 44th Presidency* [members, 2008], though, took it further: it asserted that for nationally-important systems (such as government systems or those connecting to critical infrastructure computer networks, such as those controlling the power grid), a token tied to a real person was essential.

Accountability requires that misbehaving individuals are identified, and constitutes non-trivial requirements in a system dealing with the following:

- critical infrastructure sites with the corresponding access control restrictions, such as public electricity or gas organizations, intelligence companies, etc.
- online banking, where money is transferred from one account to the other, individuals use their bank accounts to pay their bills or online purchases, issue credit cards, etc.
- online browsing or offline cases, where an individual needs to provide proof of his/her age, i.e., in a bar or restricted website where mistakes may endanger that individuals' (mental) health.

It is conceivable how strong authentication in a centralized environment makes privacy difficult to apply, while existing privacy mechanisms posit accountability, accountability of monetary systems hard to deal with. As we will see in the following section, this is exactly the challenge in this work.

2.3 Moving to Identity Management

Security and privacy are hard to combine as their application mechanisms seem to contradict each other in most of the eCommerce procedures we described before:

1. privacy preserving browsing impedes accountability required in online marketing as it weakens click fraud detection mechanisms,

2. anonymous payments complicate bank operations such as credit risk management and taxation,

3. transactions' anonymity maintenance intricate accountability mechanisms in auxiliary procedures such as the delivery of the online purchases, the construction of credible reputation systems, product taxation etc.

Referring to the more generalized map of online user activities described in section **??**, combining accountability and strong authentication leads unavoidably towards the universal institution of "strong government-issued credentials", which will be used in every online transaction activity (online banking, signing up with an ISP) of *importance*[1]. However, a centralized authentication scheme of this nature appears to threaten individual users' privacy in many cases as it enables surveillance: activities could ultimately be traced back to a single individual. In fact, the authors of the report [members, 2008] mention: "the United States should allow consumers to use strong government-issued credentials ... for online activities, consistent with protecting privacy and civil liberties". From the report, it is quite clear that such a privacy preserving identity system is little more than a *digitally-enabled national identity document.* As we will see in the following chapter, this is exactly the direction of this thesis. More specifically, in the following chapters we aim to present the specifications and cryptographic design of an architecture that can be summarized to the following:

The construction of a deployable and privacy-preserving identity management architecture addressing all aspects of electronic commerce using the existing privacy preserving crypto-

[1]We may consider *important* every infrastructure which involves money or exchange of confidential information

graphic primitives, while at the same time guaranteeing compatibility with existing business models.

Privacy, as we will see in the following chapter, is defined as the combination of consumers anonymity and their transaction unlinkability. Conditionality of privacy on proper user behavior is required for the former to be compatible with accountability and fairness mechanisms. Being eCommerce oriented, our system addresses privacy issues raised in all operations of eCommerce, i.e., online advertising, online payments, delivery of online purchases, merchant-buyer evaluation systems and aims to prevent any unauthorized entity from building and distributing consumers profiles or tracing their transactions. At the same time, the centralized "identity management" nature of our system, enables the update and maintenance of consumers global nominal credentials, such as credit score, tax credibility etc. Being card based, card-loss related mechanisms such as advanced card-owner authentication techniques, privacy-preserving card-content recovery, automatic and recursive credentials invalidation are side issues addressed through this work.

Chapter 3

An Identity Management Model

As mentioned in the previous chapter, the theme of this thesis can be summarized to the following: *The construction of a deployable and privacy-preserving identity management architecture addressing all aspects of electronic commerce using the existing privacy preserving cryptographic primitives, while at the same time guaranteeing compatibility with current business models.*

Privacy, which in our case is consumers' privacy and conditional on consumer's proper behavior, is defined as the combination of consumers anonymity and their transaction unlinkability and very commonly violated in current electronic commerce (eCommerce) activities. Being eCommerce-oriented, our system addresses privacy issues raised in all operations of eCommerce, i.e., online advertising, online payments, delivery of online purchases, and merchant-buyer evaluation systems. The ultimate goal is to prevent any unauthorized entity from building and distributing honest consumers' profiles or tracing their transactions. In opposition to existing work in the field, privacy issues here are treated more fundamentally so as to preserve fairness and accuracy in the management of consumers nominal credentials of global nature, e.g., credit scores, taxes, etc. Such variables are strongly associated with consumers monetary activities such as bank account management, payments, etc. and posit privacy application a challenging problem to deal with. To achieve proper authentication, we introduce a privacy-preserving and card-based identity management architecture, which deals with all financial activities and global variables of consumers, while enhanced with card recoverability and blacklistability mechanisms: card-loss related mechanisms such as

advanced card-owner authentication techniques, privacy-preserving card- content recovery, automatic and recursive credentials invalidation are side issues addressed through this work. Although each set of protocols designed — card-based authentication protocols, transaction protocols and banking protocols — may provide privacy independently, we use the card based-authentication mechanisms to unite the privacy-preserving protocols we built for accountable transactions and banking under a single privacy-preserving architecture.

In high level, we suggest an architecture similar to the one presented in figure 3.1. In this architecture, each valid user interacts with the registration authority RA to obtain credentials stored in an identity card IDC. Each user with an IDC can prove his validity without revealing his identity, open bank accounts, register to many other online and offline organizations, while being held accountable for misbehavior, depending on each organization's policy. Organizations may be commercial websites, in which case, the user should be able to browse on products of his preference, make online purchases, which he has delivered, evaluate his collaborators, all in a privacy-preserving but accountable way.

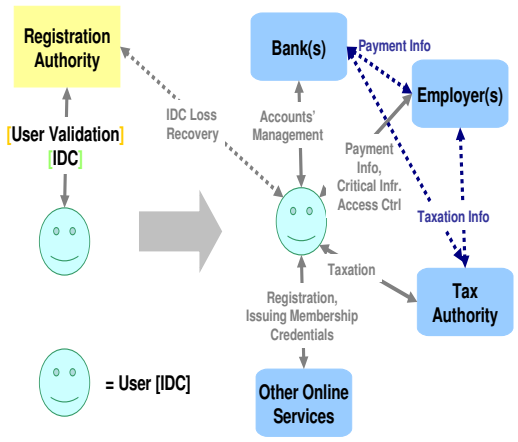

Figure 3.1: Proposed Identity Management System.

It is important to note that we are interested in the user having the *option* of privacy. More specifically, as privacy is an undeniable *right* of each individual, the latter should be

able to chose for his privacy rather than privacy being imposed on him. As depicted in figure 3.2, our system offers two types of authentication mechanisms: a privacy-preserving one, where users hide their identity completely when authenticating themselves and maintain their interactions' unlinkability, and a hybrid one, where a user is required to reveal his identity to obtain credentials of organizations such as banks, which he can later use to hide himself among the owners of such credentials. Users may use the first authentication method to make anonymous online purchases and the hybrid one to register to financial institutions, i.e., banks.In any case, if privacy is not a concern of theirs, users may use the traditional identification scheme to interact with organizations and any type of institutions. In terms of credentials, our identity management system may be considered as a mechanism generating a tree of credentials for each user, as depicted in figure 3.3, rooted secretly at each user's master IDC credential. Information for each tree is stored in the IDC and is only accessible by the IDC's owner. There are mechanisms for credentials' recovery in cases of IDC loss or of recursive credential blacklisting in cases of IDC compromise, so that attackers are left powerless. Credential blacklisting is also supported when a credential's owner misbehaves. In fact, depending on the credential type and credential-issuer institution, the blacklisting may be global (blacklisting is extended to the entire IDC) or local (only within the issuer-institution).

In this chapter we clarify the specifications of the identity management system proposed. First, in section 3.1 we define privacy and security, while we depict deployability in the context of our system. In sections 3.2 and 3.3, we introduce our threat model and the particular requirements of our scheme, while section 3.4 provides a detailed explanation of the role of following chapters.

3.1 Definitions

Privacy, *Security* and *Deployability* are the principal requirements of the system proposed. In this section we define each of them individually and show how their definitions adjust in the context of our identity management system.

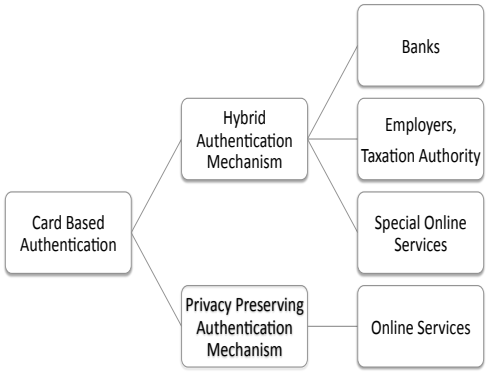

Figure 3.2: Authentication Mechanism.

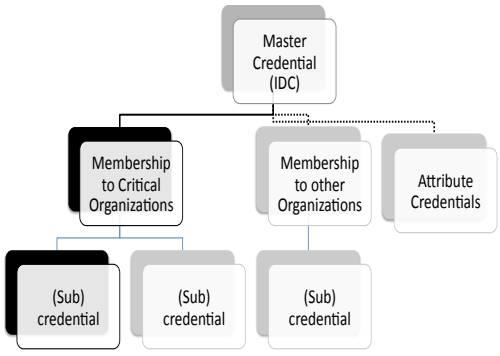

Figure 3.3: Credentials' Mechanism.

3.1.1 Privacy

According to Network Security Dictionary[Shirey, 2000], *privacy* is the right of an entity (normally a person), acting in its own behalf, to determine the degree to which it will interact with its environment, including the degree to which the entity is willing to share information about itself with others. Being so closely connected to respect towards human nature, privacy is considered fundamental in medicine, in court, in personal communication, in personal opinions, in transaction activities and in every aspect of life.

We will focus on consumer's privacy as most of the online activities of individuals involve management of their finance. A definition of the latter has been attempted in [Smith and Shao, 2007] by Smith and Shao, which is close to ours. Our privacy definition derives from the way it is currently endangered: Online transaction necessitate the divulgence of large amounts of personal information, a big part of which is desired by the merchants and the banks, as it will enable them to analyze it, discover trends and increase the efficiency of e-business dealings. In particular, both parties are able to construct the transaction profiles of their customers and either sell them (banks) or use them to increase their advertisements' efficiency (merchants).

An important property of our privacy definition is conditionality. As we aim to provide privacy to the degree the latter can be combined with accountability, we require that is only provided towards honest parties, i.e., users who misbehave should endanger their privacy. *Misbehavior* depends on the part of the system it refers to: w.r.t. our authentication system, a cheating user may attempt to impersonate another user or use more credentials than he has legally obtained to register or even lie for having lost his card; w.r.t. online purchases a misbehaving party may try to use more funds than his account balance to pay, award bad reputation to other honest users, lie for not having received his online purchase or for having paid, attempt click frauds; w.r.t. banking a user may attempt to lie for his tax his accounts have been withheld or for his payment credibility. We will refer to misbehavior in each section individually.

Although the interpretation of privacy may vary because of its conditionality as the former is applied in different parts of our system, the definition of absolute (pure) privacy we adopt consists of:

- *User Anonymity.* No system entity (registration/taxation authorities, merchants, banks etc.) may acquire any identifiable information (name/ address or any type of unique user identification) of an individual participating in a transaction or any other *activity* unless authorized by the latter;

- *Activity Unlinkabilty.* No system entity (registration/taxation authorities, merchants, banks, gateways etc.) should be able to construct the transaction or browsing profile of any consumer without the latter's consent, i.e., link two or more individual *activities* as having originated by the same user;

Activities here represent all valid operations of our system, e.g., any type of online account management operation, online purchases, product browsing, authentication procedures etc. This definition of pure privacy will be merged with security requirements to produce the various privacy definitions for each operation in the system. Evidently, as we will see in later sections, depending on our deployability and security requirements, where we make real world assumptions, in many parts of the system unlinkability may not constitute part of the definition of local privacy. However, it is essential that the privacy definition provided in each part of our architecture is preserved when the adversary has a broader view of users' interactions, i.e., entities from various parts of the system collaborate (see chapter 11 for more details).

3.1.2 Security

According to the network security dictionary[Shirey, 2000], security is defined as the combination of data availability, integrity and confidentiality. More specifically, *confidentiality* is defined as the property that information is not made available or disclosed to unauthorized individuals, entities, or processes (i.e., to any unauthorized system entity), [definitions from RFC 2828] *integrity* as the property that data has not been changed, destroyed, or lost in an unauthorized or accidental manner, and *availability* as the property of a system or a system resource being accessible and usable upon demand by an authorized system entity, according to performance specifications for the system; i.e., a system is available if it provides services according to the system design whenever users request them. In more high

level operations, security tends to acquire more coefficients and be strongly associated with correctness, accountability, fairness, and many other concepts whose strict definitions vary depending on the system. In the context of our monetary system, security consists of the following properties:

- *Correctness.* If all parties are honest, each of the system's operations will be performed in a way so that the goals of the system are achieved.
- *Fairness.* Parties in our system will be paid according to their services, i.e., if and only if they do their duty properly. Users or services who fail to act as the protocol requires will not receive any benefit, which — depending on the situation — may be a payment, a positive reputation point, or the physical object purchased.
- *Accountability.* Misbehaving parties should be detected and identified. As mentioned in previous section, misbehavior has many definitions and levels depending on the part of our architecture it refers to. As we will refer on each of them individually in the following chapters, we may now assume that for each part of our system, there is a policy defining different types of misbehavior and punishment. Accountability is achieved through the conditionality of privacy property on proper user behavior.
- *Unframability.* No user should be able to frame an honest user for being responsible for a misbehavior of the former. It is conceivable that strong accountability implies unframability.
- *Mis-Authentication Resistance.* Unless authorized, no user should be able to make use of our system or parts of it.

3.1.3 Deployability

As mentioned in the introduction, our system aims to offer the possibility of being applied in real world. Thus, *deployability* becomes important and we examine it from two aspects: (a) w.r.t. our system's applicability, where we require that our system is scalable and offers same functionalities and profits as current systems, and (b) w.r.t. the assumptions we make

for our threat model, which should be realistic. More specifically, deployability derives from the following properties:

- *Scalability.* We require that our protocols do not considerably change the current architecture and the existing communication graph of the entities participating in the system. Furthermore, we consider scalability from the aspect of computation, as we want it to scale for large numbers of users.

- *Similar Functionality.* We require that the operations addressed in our system provide the same quality of services as the existing ones and reflect most financial activities of current citizens, ranging from banking, taxation, to eCommerce.

- *Similar Profitability* for all the entities in the system. We aim in this way, to provide incentives for our system's application.

- *Real World Threat Model.* We require that the assumptions we make on our threat model represent the real world environment. In particular, we "follow the money" to construct our adversaries' motives and powers.

3.2 Threat Model

Our real world deployability assumption defines our threat model completely. In particular, we assume that

- *Users may try to cheat.* A user may try to avoid paying taxes or paying for his purchases, lie for not having received his payment. In addition, a user may try to impersonate other users to use their funds (impersonation attack) or frame other users to appear guilty for the his malicious actions. Under the aforementioned assumptions, we further assume that a user is motivated enough to attempt any type of forgery.

- *Banks are "honest but curious".* Aiming to maintain their clientele, banks are trusted to perform all their functional operations correctly, i.e., they issue credentials, open and update accounts as instructed by their customers. On the other hand, we assume that banks may use the information they possess for their customers for other reasons, i.e., to sell credit card based profiles to advertising companies, while they may

collaborate with tax authority or employers to reveal the identity behind a (swiss) anonymous account.

- *Employers may be either "honest" or "malicious".* In the general case of powerful employers, we assume "honesty" in payments towards the users, while they may try to avoid paying taxes properly. On the other hand, smaller employers may try to avoid paying employees on time or avoid paying amounts due to former employees.

- *Tax* (TA) *and Registration* (RA) *Authorities are considered to be "honest but curious".* Although we assume that are operated by the government who wants to protect honest users, the authority officials are not assumed to protect privacy; indeed, there have been a number of incidents in the U.S. of privacy violations by tax authorities or by unscrupulous individuals employed by the tax authorities.

- *Online Commercial Websites are considered to be "honest but curious".* They have to be "honest" in their functional operations to attract more users, while they use any related internet-provided information and collaborate with other websites to trace individuals' transaction activity. This is particularly beneficial for targeted ads' techniques, which base their ad-efficiency on ads' relevance to users' profiles.

- *Only if they are financially dependent, entities may combine the information they have obtained honestly to compromise users' privacy*, if, i.e., that would increase their income. In terms of collaboration, we also assume that *all entities may collaborate only in case of emergency.*

Having mentioned what our assumptions are regarding the entities in our centralized identity management system, in the following section we will elaborate on the specific requirements.

3.3 Requirements

In this section we illustrate how privacy, security and deployability are incorporated in our system's requirements. We refer to the requirements of the general card-based identity system architecture (General ID System Requirements), as well as to the ones related

to each activity individually: Bank Account Management, Taxation, Employment Access Control and Online Subscriptions. Although, online subscriptions capture the requirements of users' online commercial activities, we will examine the very specific requirements for each operation in later chapters.

3.3.1 General ID System Requirements.

First of all, as we require a centralized strong authentication scheme, each identity should be ultimately uniquely identified. In particular, we require that each user U in our system

1. interacts with the registration authority RA *once* to issue a *single* valid identity card. We will refer to this card as IDC. Privacy requires that IDCs are created with citizens' collaboration such that only them can reissue them. The requirement for the one-to-one correspondence between users and IDCs resembles the way current citizenship logs operate: *each citizen is only registered once to his country of citizenship.* As birth certificates or current citizenship-related identity cards, IDCs should be enough to prove users' validity. (We recognized that this is a very difficult process. Solving it is outside the scope of this work; we assume that governments have sufficient motive for doing it regardless. Again, see [Kent and Millett, 2002] for some of the issues.)

2. should not be able to create an IDC by himself or by collaborating with any other user or organization (IDC-unforgeability). This requirement is linked to the per person uniqueness of the IDC.

3. should be the only one able to prove ownership of his IDC. In current identity systems, this is achieved through the photo attached to the ID card. However, as we want IDCs to provide no information regarding its owner to any unauthorized third party, photos cannot be used in this case.

4. can demonstrate ownership of his IDC multiple times, in a way so that no one can reveal U's name (untraceability). In addition no one should be able to link two IDC ownership demonstrations as been originated by the same user (unlinkability).

5. may use his IDC in order to prove specific attributes related to himself without re-

vealing anything more than what required. Attributes of this type may be his age or adulthood in bars in various countries, his driving license, or his criminal record.

6. may use his IDC to obtain membership to different types of services, i.e., online customer services, banks, football clubs, etc. In fact, depending on the type of the organization and the corresponding user-registration procedure, organization-issued membership credentials may be linked to its members' IDC. In this way local misbehavior of a member will lead to overall user-identification.

We deal with these issues in chapter 5.

3.3.2 Bank Account Management.

In terms of user interaction with banks, we require that

1. each valid user should provide strong identification credentials to the bank to be able to make use of its services.

2. each user should be able to open no more bank accounts than the ones he is entitled to open based on his financial status. For privacy purposes, we may consider enabling the bank system to support two types of accounts:

   - *Anonymous but traceable accounts*, namely accounts which cannot be linked to a particular identity and more specifically to their owners, but whose activity may be observable by the bank — i.e., through credit cards. It is critical that anonymity in this case does not violate the accountability requirement. Users should only be able to open such accounts if they are financially eligible to, while they should be taxed accordingly.
   - *Accounts with someone's name attached*, which are similar to the accounts currently supported by most banks.

3. A user should be taxed on the overall amount of money he keeps in a bank regardless of in how many accounts his savings are distributed. In the special case of anonymous accounts, taxation should be done in a way so that no privacy breach is caused.

We deal with these issues in chapter 9.

3.3.3 Employment Access Control.

It is likely that complete user anonymity is not an issue towards employers. Employers do need to know the full identity of their employees. In fact, in some extreme cases of national security organizations, details of the personal life of the individual are necessary to decide for that individual's credibility. Consequently, the privacy definition in the context of users' employment systems is restricted to the avoidance of any bank-account privacy breach due to employers' interaction with the banks or taxation authority throughout employees' payment and taxation respectively.

In particular, we require that payment proceeds in such a way that the employer-bank collaboration will not reveal the identity of the owner of a particular anonymous account to which the payment takes place. However, the payment should be conducted in an accountable and fair way. In addition, the employer must interact with the taxation authority TA to provide payment information regarding its employees. No privacy issues arise from this interaction since both entities know the identities of the users and the taxes withheld.

Another employment-wise security concern is be access control for specified sites. We emphasize the common case where we need to have strong authentication within the company but complete anonymity outside the company. A typical example for that would be in critical infrastructure systems: when an employee of a particular company logs in to a critical infrastructure's website, such as a SCADA system, the employee's exact identity should be knowable by that particular company's department, while for any entity outside the company, the employee should be hidden among the employees of that department (company). In any case, it should be possible to trace a misbehaving party.

3.3.4 Taxation.

Tax Authority (TA) should be able for each individual user to verify the latter's income and verify that the appropriate bank-account withholding taxes have been applied. Privacy, as in employers' case, does not require that TA does not know the identities of the users. However, we want to enforce that his employment or bank-account privacy is not compromised through TA's interactions with the banks or employers. Taxation is among the issues we deal with in online banking.

3.3.5 Other Online Services.

Online services may include users' subscriptions to various websites, such as magazines, travel agencies, concert venues, gyms, etc. A user's medical account login may also constitute another online activity of his. In all these cases, we do require that

1. a user's identity is not revealed, unless with the latter's consent.

2. a user's online activity cannot be monitored, i.e., we want to enforce that no third party — other than the user himself and in many cases the online service system — should be able to link two different browsing or purchase activities of the same user. In this way, we want to avoid any user-profiling without user's consent.

In addition, we require that *misbehaving* entities are punished. Both, misbehavior and punishment depend entirely on the nature of the websites. Banks or commercial websites which merchandize high-end products may require revocation of the anonymity of a customer who tried to spend more than his bank account balance or did not pay for his purchases. On the other hand, magazine and blog-oriented websites may require the exclusion of users who have used bad language on their comments, depending on their policy. We will further elaborate on eCommerce activities and requirements in chapters 6, 7 and 9.

3.4 Organization

In the following chapters, we will show how we dealt with all the aforementioned privacy issues through a privacy preserving nevertheless accountable identity management system. In fact, as our protocols address various online users' activities, our contribution in the field of realizable privacy acquires a multidimensional character. In this dissertation, we address the following

- *Anonymous Online Browsing*, as it constitutes a prerequisite for enforcing privacy in any type of online activity. Anonymous networking has been known since 1981 [Chaum, 1981], while a more practical scheme, Onion Routing, was first described in 1995 [Goldschlag *et al.*, 1996]. Unfortunately, current deployments of anonymization

networks depend on a very small set of nodes that volunteer their bandwidth. Assuming that the main reason is the practical limitations in bandwidth and latency that stem from limited participation, we propose providing economic incentives, which historically have worked very well. In specific, in chapter 6 we demonstrate a payment scheme that can be used to compensate nodes which provide anonymity in Tor, an existing onion routing, anonymizing network. Our system claims to maintain users' anonymity, although payment techniques mentioned previously – when adopted individually – provably fail.

- *Privacy in Online Marketing.* Thanks to its ability to target audiences combined with its low cost, online advertising has become very popular throughout the past decade, and has become an essential source of income for many websites. The challenge in this case is the fact that although current profile-based advertising techniques raise privacy risks and may contravene users' expectations, privacy-preserving techniques, e.g., anonymous browsing, create many opportunities for fraud. In chapter 7 we analyze the privacy concerns raised by online advertising as well as the subsequent security issues, and propose a privacy preserving set of protocols that provide targeted ads with guaranteed fraud detection.

- *Privacy preserving Authentication* is the card based authentication system which constitutes the core of our system. In particular, as we will see in detail in chapter 5, we introduce a unique per user digital identity card, with which its owner is able to authenticate himself, prove possession of attributes, register himself to multiple online organizations (anonymously or not) and provide proof of membership. Unlike existing credential-based identity management systems, this card is revocable, i.e., its legal owner may invalidate it if physically lost, and still recover its content and registrations into a new credential. This card will protect an honest individual's anonymity when applicable as well as ensure his activity is known only to appropriate users.

- *Privacy in Online Banking.* As mentioned before, banking is one of the operations slowly transferred to the online world and involves the management of bank accounts, issue of credit cards and the corresponding tax and credit risk mechanisms. In chap-

ter 9, we extensively elaborate on banking privacy issues and present a series of protocols allowing individuals to issue anonymous credit cards to participate in online or offline transactions (see section 9.2) or open anonymous bank accounts for which they are taxed fairly (see section 9.1), while all the operations are enhanced with a privacy preserving credit score update mechanism (see section 9.3). The anonymity provided in these cases is conditional, i.e., users who attempt to take advantage of their "anonymity" to spend more than their accounts' balance, lie for their taxes or use another user's account, will be identified. Measures are taken so that anonymous accounts' ownership is revealed in emergency situations.

- *Privacy in Online Transactions.* In the context of online transactions, in this dissertation, we deal with three separate operations: transaction payment, product delivery and transaction evaluation. Privacy preserving online payments are introduced in chapter 9 as part of the online banking set of protocols. In chapter 8 we deal with the other two. More specifically, we present an anonymous — but accountable — physical object delivery system, were the recipient of a package remains anonymous towards both the delivery company and the merchant (see section 8.1), while in this dissertation we demonstrate protocols addressing each one of these procedures.

- *Privacy in Employment Payment and Access Control.* In particular, in chapter 10 we exemplify a series of protocols that establish an employee management architecture, where employees may be permitted access to systems according to their responsibilities and roles, while their activity within those systems being conditionally untraceable, i.e., the identity of an actor is revealed only when a functional error takes place. In our system we embody techniques that guarantee fair and taxable monthly payments flows from employers to their employees, such that the banks holding employees' accounts are not able to link the accounts to their owners even when collaborating with employers.

Before presenting our protocols, in the following chapter we make a brief presentation on the building blocks of our schemes and their security properties.

Part II

Methodology

Chapter 4

Cryptographic Primitives

For the construction of our protocols we utilized many existing cryptographic primitives. In particular, we made extended use of multiple types of anonymous credentials, e.g., [Camenisch and Lysyanskaya, 2001; Tsang *et al.*, 2007], various types of digital cash [Camenisch *et al.*, 2005; Camenisch *et al.*, 2006], (blind) group and digital signatures, etc. In this chapter we elaborate on all the building blocks we used while we emphasize on the exact properties of theirs which posited them as the most appropriate tools for our purposes.

4.1 Electronic cash

An E-Cash (EC) [Camenisch *et al.*, 2005][Camenisch *et al.*, 2006] system consists of three types of players: the *bank*, *users* and *merchants*. The input and output specifications of the basic operations are as follows. For convenience, we will assume that the operations take place between a merchant M, a user U and the Bank B.

- $(pk_{\mathsf{B}}, sk_{\mathsf{B}}) \leftarrow \mathsf{EC.BKeyGen}(1^k, params)$ and $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) \leftarrow \mathsf{EC.UKeyGen}(1^k, params)$, which are the key generation algorithm for the bank and the users respectively.
- $\langle W, \top \rangle \leftarrow \mathsf{EC.Withdraw}(pk_{\mathsf{B}}, pk_{\mathsf{U}}, n)$ $[\mathsf{U}(sk_{\mathsf{U}}), \mathsf{B}(sk_{\mathsf{B}})]$. In this interactive procedure, U withdraws a wallet W of n coins from B.
- $\langle W', (S, \pi) \rangle \leftarrow \mathsf{EC.Spend}(pk_{\mathsf{M}}, pk_{\mathsf{B}}, n)$ $[\mathsf{U}(W), \mathsf{M}(sk_{\mathsf{M}})]$. In this interactive procedure, U spends a digital coin with serial S from his wallet W to M. When the procedure is over, W is reduced to W', M obtains as output a coin (S, π), where π is a proof of a valid coin with a serial number S.
- $\langle \top/\bot, L' \rangle \leftarrow \mathsf{EC.Deposit}(pk_{\mathsf{M}}, pk_{\mathsf{B}})$ $[\mathsf{M}(sk_{\mathsf{M}}, S, \pi),\ \mathsf{B}(sk_{\mathsf{B}}, L)]$. In this interactive procedure, M

deposits a coin (S, π) into its account in the bank. If this procedure is successful, M's output will be $\top$ and the bank's list L of the spent coins will be updated to L'.

- $(pk_{\mathsf{U}}, \Pi_G) \leftarrow \mathsf{EC.Identify}(params, S, \pi_1, \pi_2)$. When the bank receives the two coins with the same serial number S and validity proofs π_1 and π_2, it executes this procedure, to reveal the public key of the violator accompanied with a violation proof Π_G.
- $\top/\bot \leftarrow \mathsf{EC.VerifyGuilt}(params, S, pk_{\mathsf{U}}, \Pi_G)$. This algorithm, given Π_G publicly verifies the violation of pk_{U}.
- $\{(S_i, \Pi_i)\}_i \leftarrow \mathsf{EC.Trace}(params, S, pk_{\mathsf{U}}, \Pi_G, D, n)$. This algorithm provides the list of serials S_i of the ecoins a violator pk_{U} has issued, with the corresponding ownership proofs Π_i.
- $\top/\bot \leftarrow \mathsf{EC.VerifyOwnership}(params, S, \Pi, pk_{\mathsf{U}}, n)$. This algorithm allows to publicly verify the proof Π that a coin with serial number S belongs to a user with public key pk_{U}.

[Camenisch *et al.*, 2006] is a money-laundering prevention version of [Camenisch *et al.*, 2005], where anonymity is revoked when the spender spends more coins to the same merchant than a spending limit. In this case ecoins are upgraded to

$$C = (S, V, \pi),$$

where V is a merchant-related locator, while EC.Identify and EC.VerifyGuilt procedures are upgraded to the DetectViolator and VerifyViolation to support the extended violation definition.

Security Properties: *(a) Correctness.* The protocols work if all players are honest. *(b) Balance.* No collection of users and merchants can ever spend more coins than they withdrew. *(c) Identification of Violators.* Given a violation and the corresponding proofs of guilt, the violator's public pk_{U} key is revealed such that EC.VerifyViolation accepts. *(d) Anonymity of users.* The bank, even when cooperating with any collection of malicious users and merchants, cannot learn anything about a user's spendings other than what is available from side information from the environment. *(e) Exculpability.* An honest user U cannot be accused for conducting a violation such that EC.VerifyViolation accepts. *(f) Violators' Traceability.* Given a violator Uwith a proof of violation Π_G, this property guarantees that EC.Trace will output the serial numbers of all coins that belong to U along with the corresponding proofs of ownership, such that for each one of them VerifyOwnership accepts.

4.2 Blacklistable Anonymous Credentials

The entities in the blacklistable credential system BAC of [Tsang *et al.*, 2007] are the Group Manager GM, a set of service providers SPs and users. The procedures supported are the following:

- $\langle \mathsf{gpk}, \mathsf{gsk} \rangle \leftarrow \mathsf{BAC.Setup}[\mathsf{GM}(1^k)]$. This algorithm generates a group public key gpk and the GM's secret group information gsk.
- $\langle \mathsf{cred_U}, \mathsf{JLog_U} \rangle \leftarrow \mathsf{BAC.Register}(\mathsf{gpk})[\mathsf{U}, \mathsf{GM}(\mathsf{gsk})]$. When this interactive registration ends, U has obtained his membership credential $\mathsf{cred_U}$.
- $\langle \top/\bot \rangle \leftarrow \mathsf{BAC.Authenticate}(\mathsf{gpk})\ [\mathsf{U}(\mathsf{cred_U}),\ \mathsf{SP}(\mathrm{BL})]$. In this interactive procedure, U proves to SP that he is a valid (non-blacklisted) member of the group.
- $\langle BL' \rangle \leftarrow \mathsf{BAC.BLAdd}[\mathsf{SP}(BL)]$, where a service provider ads a credential (ticket) to the blacklist BL.
- $\langle tick \rangle \mathsf{BAC.BLExtract}[\mathsf{SP}(BL)]$, where SP extracts an element from the blacklist.
- $\langle BL' \rangle \leftarrow \mathsf{BAC.BLRemove}[\mathsf{SP}(BL)]$, where SP removes a credential from the blacklist.

Security Properties: *(a) Correctness.* The protocols work if all parties are honest. *(b) Misauthentication Resistance.* No unregistered user or collection of unregistered users should be able to authenticate themselves. *(c) Blacklistability.* SPs may blacklist any misbehaving user of the system and restrict him from any ability of authenticating himself. *(d) Anonymity.* SPs may only learn whether a user is blacklisted or not; no identification information may be leaked. *(e) Non-framability.* An honest user should never be blocked from access.

4.3 Anonymous and Unlinkable Credential System, Pseudonym Systems

Pseudonym systems (PS) have three types of players: *users, organizations,* and *verifiers.* Users are entities that receive credentials. Organizations are entities that grant and verify the credentials of users. Finally, verifiers are entities that verify credentials of the users. See [Lysyanskaya *et al.*, 1999; Camenisch and Lysyanskaya, 2001] for more details. The standard operations supported are the following:

- $(pk_{\mathsf{O}}, sk_{\mathsf{O}}) \leftarrow \mathsf{PS.OKeyGen}(1^k)$. This procedure generates a public/secret key pair for an organization. We denote a key pair for an organization O by $(pk_{\mathsf{O}}, sk_{\mathsf{O}})$.

- $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) \leftarrow \mathsf{PS.UKeyGen}(1^k)$. This procedure generates a public/secret key pair for a user. We denote a key pair for a user U by $(pk_{\mathsf{U}}, sk_{\mathsf{U}})$. Sometimes we refer to the secret key of a user as a master secret key for the user.

- $\langle (N, \mathsf{NSecr}_N), (N, \mathsf{NLog}_N) \rangle \leftarrow \mathsf{PS.FromNym}(pk_{\mathsf{O}})\, [\mathsf{U}(pk_{\mathsf{U}}, sk_{\mathsf{U}}),\ \mathsf{O}(sk_{\mathsf{O}})]$. This interactive procedure between a user and an organization generate a pseudonym (or simply nym). The common input is the public key of the organization O. The output for the user is a nym N and some secret information NSecr_N, and for the organization the nym N and some secret information NLog_N.

- $\langle \mathsf{cred}_N, \mathsf{CLog}_{\mathsf{cred}_N} \rangle \leftarrow \mathsf{PS.GrantCred}(N, pk_{\mathsf{O}})\ [\mathsf{U}(pk_{\mathsf{U}}, sk_{\mathsf{U}}, \mathsf{NSecr}_N),\ \mathsf{O}(sk_{\mathsf{O}}, \mathsf{NLog}_N)]$. This interactive procedure between a user and an organization generate a credential for a nym N. The common input is N and pk_{O}. The output for the user is the credential cred_N for the nym N. The output for the organization is some secret information $\mathsf{CLog}_{\mathsf{cred}_N}$ for the credential.

- $\langle \top, \top \rangle / \langle \bot, \bot \rangle \leftarrow \mathsf{PS.VerifyCred}(pk_{\mathsf{O}})\ [\mathsf{U}(N, \mathsf{cred}_N), V]$. In this interactive procedure between a user and a verifier, the user proves that he has a credential on the nym N issued by the organization O.

- $\langle \top, \top \rangle / \langle \bot, \bot \rangle \leftarrow \mathsf{PS.VerifyCredOnNym}\ (N, pk_{\mathsf{O}}, pk_{O_1})\ [\mathsf{U}(N_1, \mathsf{cred}_{N_1}), \mathsf{O}(\mathsf{NLog}_N)]$. In this interactive procedure between a user and the organization O, the user proves that N is his valid nym of the organization O and that he has a credential cred_{N_1} on the nym N_1 issued by the organization O_1.

Security Properties. *(a) Unique User for Each Pseudonym.* Even though the identity of a user who owns a nym must remain unknown, the owner should be unique. *(b) Unlinkability of Pseudonyms.* Nyms of a user are not linkable at any time better than by random guessing. *(c) Unforgeability of Credentials.* A credential may not be issued to a user without the organization's cooperation. *(d) Consistency of Credentials.* It is not possible for different users to team up and show some of their credentials to an organization and obtain a credential for one of them that the user alone would not have gotten. *(e) Non-Transferability.* Whenever Alice discloses some information that allows Bob to user her credentials or nyms, she is effectively disclosing her master secret key to him.

4.4 Group Signature Scheme

In a typical group signature scheme GSS, there is a group manager (GM), the group-members, who act as signers (let each be S) and produce signatures on behalf of the group. The procedures supported are the following:

- $(\mathsf{gpk}, \mathsf{gsk}) \leftarrow \mathsf{GS.Setup}(1^k)$. This algorithm generates a group public key $\mathsf{gpk}$and the GM's secret group information gsk.
- $\langle \mathsf{bgusk_S}, \mathsf{JLog_S} \rangle \leftarrow \mathsf{GS.Join}(\mathsf{gpk})[\mathsf{S}, \mathsf{GM}(\mathsf{gsk})]$. When this interactive join procedure ends, an S obtains a secret signing key $\mathsf{bgusk_S}$, and the GM (group manager) logs the join transcript in the database D.
- $\sigma \leftarrow \mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{bgusk_S}, m)$. This algorithm generates a group signature on a message m.
- $\langle \top/\bot \rangle \leftarrow \mathsf{GS.Verify}(\mathsf{gpk}, m, \sigma)$. This is a verification algorithm.
- $\mathsf{ms} \leftarrow \mathsf{GS.Open}(\mathsf{gsk}, \sigma, D)$. With this algorithm the GM determines the identity of the group member who generated the signature σ.

Security Properties: *(a) Anonymity.* Given a signature and two members, one of whom is the originator, the adversary can identify its originator among the group members no better than randomly. *(b) Unforgeability.* The adversary cannot produce a valid group signature without owning group membership information. *(c) Non-framability.* The adversary cannot create a valid group signature that opens to another group member.

4.5 Blind Signature Scheme

In a typical blind signature scheme BSS, there are signers (let each be S) who produce signatures on messages of users (let each be U), without knowing the exact message they are signing. The procedures supported are the following:

- $(pk_\mathsf{S}, sk_\mathsf{S}) \leftarrow \mathsf{BS.KeyGen}(1^k)$. This is a key-generation algorithm that outputs a public/secret key-pair $(pk_\mathsf{S}, sk_\mathsf{S})$.
- $\langle \top/\bot,\ \sigma/\bot \rangle \leftarrow \mathsf{BS.Sign}(pk_\mathsf{S})[\mathsf{S}(sk_\mathsf{S}),\ \mathsf{C}(m)]$. At the end of this interactive procedure, the output of the S is either *completed* or *not-completed* and the output of U is either the signature (σ) or a failure sign $(\bot)$.
- $\langle \top/\bot \rangle \leftarrow \mathsf{BS.Verify}(m, \sigma, pk_\mathsf{S})$ is a verification algorithm.

Security Properties: *(a) Unforgeability. (b) Blindness.* S does not learn any information about the message m on which it generates a signature σ.

4.6 Blind Group Signature Scheme

In a typical group signature scheme BGS we can identify the group manager(GM), who maintains the BGS group administration information, the group-members who produce group signatures on users' messages. For now we will assume that a user U, has requested group member S to produce a signature on message m. The procedures supported are the following:

- $(\mathsf{bgpk}, \mathsf{bgsk}) \leftarrow \mathsf{BGS.Setup}(1^k)$. This algorithm generates a group public key bgpk and the GM's secret administration information bgsk .
- $\langle \mathsf{bgusk_S}, bcert_\mathsf{S}, \mathsf{BJLog_S} \rangle \leftarrow \mathsf{BGS.Join}(\mathsf{bgpk})[\mathsf{S}, \mathsf{GM}(\mathsf{bgsk})]$. When this interactive join procedure ends, S obtains her secret signing key $\mathsf{bgusk_S}$, her membership certificate $bcert_\mathsf{S}$, and the GM logs the join transcript in the database D.
- $\sigma \leftarrow \mathsf{BGS.Sign}(\mathsf{bgpk})[\mathsf{S}(\mathsf{bgusk_S}), \mathsf{U}(m)]$, where U obtains a signature on m.
- $\langle \top / \bot \rangle \leftarrow \mathsf{BGS.Verify}(\mathsf{bgpk}, m, \sigma)$. This is a verification algorithm run by a verifier.
- $\mathsf{S} \leftarrow \mathsf{BGS.Open}(\mathsf{bgsk}, \sigma, D)$. This algorithm is run only by GM and determines the identity of the S which generated the signature σ.

Security Properties: They combine the properties of group and blind signature schemes: *(a) Anonymity, (b) Unforgeability, (c) Non-framability, (d) Undeniable Signer Identity* towards the group manager, *(e) Signatures' Unlinkability* and *(f) Blindness.*

4.7 Zero Knowledge Proof of Knowledge

In a typical zero knowledge proof of knowledge(ZKPOK) scheme there are two types of players, the provers who need to prove possession of one or more secret number(s), that satisfy a particular property to one or more verifiers and the verifiers. In what follows, we will use the notation introduced by Camenisch and Stadler in [Camenisch and Stadler, 1997] for the various proofs of knowledge of discrete logarithms and proofs of the validity of statements about discrete logarithms. In particular,

$$PK\{(\alpha, \beta, \gamma) : y_1 = g_1^\alpha h_1^\beta \wedge y_2 = g_2^\alpha h_2^\gamma \wedge (u \leq \alpha \leq u)\}$$

denotes a "zero-knowledge-proof-of-knowledge" of integers α, β and γ such that $y_1 = g_1^{\alpha} h_1^{\beta}$ and $y_2 = g_2^{\alpha} h_2^{\beta}$, where $u \leq \alpha \leq u$ and $y_1, g_1, h_1, y_2, g_2, h_2$ are all elements of two groups G_1 and G_2 respectively. We make use of the following ZKPoK schemes:

A proof of knowledge of a representation of an element $y \in G$ with respect to bases $z_1, \ldots, z_v \in G$ [Chaum *et al.*, 1988a], i.e.,

$$PK\{(\alpha_1, \ldots, \alpha_v) : y = z_1^{\alpha_1} \cdot \ldots \cdot z_v^{\alpha_v}\}.$$

A proof of equality of discrete logarithms of $y_1, y_2 \in G$ to the bases $g, h \in G$ respectively, [Chaum, 1991; Chaum and Pedersen, 1993] i.e.,

$$PK\{(\alpha) : y_1 = g^{\alpha} \ \wedge \ y_2 = h^{\alpha}\}.$$

A proof of knowledge of a discrete logarithm of $y \in G$ with base $g \in G$ such that $log_g y$ lies in the interval [a,b], [Boudot, 2000],i.e.,

$$PK\{(\alpha) : y = g^{\alpha} \ \wedge \ \alpha \in [a, b]\}.$$

Proof of knowledge that the discrete logarithms of two group elements $y_1 \in G_1$, $y_2 \in G_1$ to the bases $g_1 \in G_1$ and $g_2 \in G_2$ in the different groups G_1 and G_2 are equal [Brickell *et al.*, 1988], [Camenisch and Michels, 1999], i.e.,

$$PK\{(\alpha, \beta) : y_1 =^{G_1} g_1^{\alpha} \ \wedge \ y_2 =^{G_2} g_2^{\alpha} \wedge C =^{G} g^{\alpha} h^{\beta} \ \wedge \ \alpha \in [0, min(q_1, q_2)]\},$$

where q_1, q_2 are the order of the groups G_1, G_2 respectively, $G =< g >=< h >$ is a group to which the commitment C of α, β is computed.

Properties. *(a) Correctness.* The protocol works when both prover and verifier are honest. *(b) Zero-Knowledge.* The verifier learns nothing regarding the secret values the prover proves knowledge of. *(c) Proof of Knowledge.* The protocol accepts only iff the prover knows the secret value he claims to know.

4.8 Commitments

In a typical commitment scheme, there are provers (let each be P who are required to commit to a particular value towards verifiers (let each be V), who may be able to see the committed value when provers decide to. The procedures supported are the following:

- $(params) \leftarrow \mathsf{CS.Setup}(1^k)$. This is a parameter-generation algorithm that outputs the public parameters of a commitment scheme.
- $(\mathsf{C}/false) \leftarrow \mathsf{CS.Commit}(params)[\mathsf{P}r, m)]$. At the end of this procedure, the public output is either the committment itself to a value m or *not-completed*. Ps private input is the committed message m and randomness r.
- $\langle \top/\bot, m/\bot \rangle \leftarrow \mathsf{CS.Open}(\mathsf{C})[\mathsf{P}m), \mathsf{V}]$. In this operation the P shows the committed value m to the verifier. The verifier accepts it if m is the value matching C.

Security Properties: (a) Binding. It should be computationally impossible for the prover, after having committed to a message m, to generate another message m' that has the same commitment value C; in this way, the prover is bound to the value he initially committed to. *(b) Hiding.* It should be computationally impossible for a verifier who knows C to get any information regarding m.

Part III

Solution

Chapter 5

Privacy-Preserving Strong Authentication

Currently in the real world, citizens collaborate with a public authority to issue a national identity card. This card provides a physical proof of identity which they use to open accounts in banks, to be employed and receive their payments, and to prove their age. However, users are not currently required to prove their national identity in online communications and, thus, cannot be punished for misbehaving in these systems. As implied in the introductory chapters, we deal with this by incorporating a card-based identity management set of protocols as the central authentication scheme in our architecture. The option of privacy is also supported. More specifically, we propose an architecture, where each valid user interacts with a registration authority RA to obtain credentials stored in an identity card IDC. An IDC can be used only by its owner in combination with its owner's national identity card or completely independently to provide privacy, i.e., for its owner to prove that he is associated with a valid national identity without revealing it.

Existing master secret based privacy preserving identity systems offer many advantages for such an architecture. Users in these systems may obtain anonymous certificates from a central certificate authority, use these certificates to register anonymously in various organizations and obtain pseudonymous membership credentials. The credentials issued are unforgeable and can provide conditional anonymity and user-activity untraceability

that is revokable for misbehaving users. Concurrently, master secret being so important for their online activity, users are motivated not to share their master secret; thus, credentials cannot be used by others. Unfortunately, current anonymous credential systems fail to consider many "real world" issues and this is the theme of this chapter.

First of all, most credential systems do not have a scalable solution for *credential blacklistability*. This is important given the large number of identity theft cases in the real world. Another "real world" issue is *variety* in organization registration policies. For example, a bank will probably require a lot more information to register a user than an online subscription service. However, the most important deficiency of current credential systems is the the lack of a proper *secret information update* procedure. In "master-secret"-driven centralized systems, master secrets may be compromised or stolen and, currently, there is no efficient way for a user to transfer his old registrations and credentials to his new account, meaning that he has no advantage over an attacker who has compromised his old secret.

In this chapter we address all of these issues in an identity management architecture which will provide a means of privacy-preserving but centralized means of authentication for all functionalities of our system. In particular, we present a privacy-preserving and master secret-based identity management system, where every individual can generate a single master secret, which he can

- use to prove identity multiple times unlinkably and anonymously,
- use to register to multiple organizations (or services) along with other credentials depending on each organization's policy
- update in a manner that recursively updates all generated sub-credentials
- recover, when lost or destroyed

We emphasize the fact that our system is deployable, in other words:

- we make "real world" assumptions on our adversary's powers,
- we consider "real world" settings for the various organizations
- we propose protocols that scale for a large group of participants

Organization. This chapter is organized as follows: in sections 5.1.1 and 5.1.3 we present the architecture, threat model and requirements of our system; in section 5.2 we present our protocols, while in section 5.3 we elaborate on how privacy, security and security are incorporated in our protocols.

5.1 A model for Privacy-Preserving Identity Management

In our identity management system individuals have a single identity, register with a public authority, and obtain government-issued credentials. Individuals may use these credentials to participate in a number of real-world *interactions*, which include, without being limited to, the most important id-based activities of an individual, such as handling of employment, management of bank accounts, verification of specific attributes of the individual (e.g. legal drinking age), and registration in multiple online or offline clubs, associations or services. As discussed in chapter 2, our identity management system consists of the following entities:

- *Users*, who may interact with other users or organizations in order to perform various tasks. A single user should map directly to a citizen.
- The *Registration Authority* (RA), which is responsible for registering users and managing the construction, modification, and destruction of government-issued credentials. In keeping with the mapping between users and citizens, this will likely be realized by a country's official citizenship registry, for example the social security office.
- *Special financial organizations*, who include employers that form employment relationships with users and banks that allow users to open accounts for the purpose of storing cash and handling financial transactions.
- *Non-financial organizations*, who may wish to extend membership to users and are not responsible for tax reporting. Such organizations may include gym centers, schools, commercial websites, etc.

To achieve the required *user-activity centralization* — imposed by our needs for accountability and strong authentication — in our system, each user U generates a unique

master secret $\mathsf{ms_U}$, which will be accounted for when the former misbehaves.[1] Users must use their master secret to authenticate themselves towards multiple organizations and are thus highly motivated not to share it. Seemingly contradictory to accountability, privacy requires that IDC-ownership demonstration does not leak any information regarding its owner or link multiple interactions of the latter. To achieve the aforementioned properties we use [Tsang *et al.*, 2007] (see section 4.2) as a base for the user-membership in our system (RA-registration). The registration authority RA maintains two important databases: the $\mathsf{DB_{RA}}$, where the registered-user information is stored, and the $\mathsf{BL_{RA}}$, which serves card and user blacklistability purposes. In addition RA maintains $\mathsf{BL_{RA}^{hold}}$, a temporary list of the anonymous owners of accounts in debt.

Apart from the generation and validation of his $\mathsf{ms_U}$, each user U collaborates with the RA to issue three types of credentials, which are stored in U's IDC and can be used to demonstrate U-validity multiple times anonymously:

1. a registration credential regtick, which authorizes U as a valid user of the system and serves user-blacklistability purposes,

2. a wallet of one-time-use credentials perm-credentials, which can be used for U to register to various organizations, and

3. a wallet of one-time-use cred-credentials, which can be used for U to register to organizations, who limit registrations to one per user, i.e., hospitals etc.

Every time U registers to an organization, he makes use of his $\mathsf{ms_U}$ to demonstrate knowledge of his regtick, and to use one of his credentials. The transcripts of these demonstrations will be used as the local identity of U within the organization and serve blacklistability purposes when U misbehaves. Afterwards, U may have credentials generated within that organization using any known anonymous credential system depending on the type of the organization.

In addition to his registration credentials, U may apply for a number of attribute credentials from the RA, such as possession of driving license or of a car, adulthood, etc. and their

[1] As we will see later on, in our system *misbehavior* is a concept, whose definition varies across different organizations.

time of validity may vary. After verifying U's real-world validity, RA issues blind attribute related permissions, att-credentials, bound to U's $\mathsf{ms_U}$, for the former to deposit to the corresponding attribute services. Ultimately, proof of possession of attribute att is realized through proof of U-membership to the corresponding (att) group. U contacts the attribute-agencies anonymously but using his regtick and att-credential and obtains membership to that group the same way he obtains membership to organizations.

To fully recover his $\mathsf{ms_U}$ and the content of his card, when lost or compromised, U participates in a recovery setup protocol which has two important phases. The first one is performed at the end of his RA-registration, where U shares his $\mathsf{ms_U}$ and a recovery secret key $\mathsf{dk_U}$ in a shared secret fashion; $\mathsf{dk_U}$ will serve U-authentication purposes. The second phase takes place at the end of each organization-registration procedure, when the organization U collaborates with uses a central database, $\mathsf{DB_{MS}}$, to store confidentially subscription information anonymously but authoritatively information. The latter is encrypted with the encryption key $\mathsf{ek_U}$ which corresponds to $\mathsf{dk_U}$. Second level authentication information stored in RA ($\mathsf{RegInfo_U}$) and the organizations (AuthData) for each member, authorize an honest user, who has lost his card and properly recovered his $\mathsf{dk_U}$ to order the blacklisting of his old regtick, trace his subscriptions, prove ownership and blacklist all the credentials issued with his old master secret. An attacker in this case, would not have the appropriate information to succeed in this authentication and will thus be unable to make use of the card.

5.1.1 A "real world"-based threat model

The threat model of our identity management system has been presented in chapter 3 The assumptions we make regarding our adversaries are based on "real-world" settings and motives. In particular, we assume that:

- *Users may try to cheat.* A user may try to avoid paying for his purchases or bills or falsely claim he has not received a payment. In addition, he may try to impersonate other users to use their funds (impersonation attack) or frame other users for various misbehaviors.

- *(Non-)Financial Organizations are "honest but curious".* Aiming to maintain their clientele, banks, and other organizations, are trusted to perform all their functional operations correctly, i.e., they issue credentials, open and update accounts as instructed by their customers. However, they may attempt to learn more than is appropriate from their views of these transactions (this could be motivated by profiting from this information, i.e. selling behavior profiles to advertising companies). They may also try to collaborate with each other to improve their information advantage.

- *The Registration Authority is considered to be "honest but curious".* We assume that is operated by the government who wants to protect honest users, but it is not trusted to protect users' privacy.

5.1.2 Operations

To define the operations of our system more strictly we will use the following notation: when an operation is an interactive procedure (or a protocol consisting of multiple procedures) between two entities A and B, we denote it by $\langle O_A, O_B\rangle \leftarrow \mathsf{Pro}(I_C)[A(I_A), B(I_B)]$, where Pro is the name of the procedure (or protocol). O_A (resp. O_B) is the private output of A (resp. B), I_C is the common input of both entities, and I_A (resp. I_B) is the private input of A (resp. B).

1. $(pk_{\mathsf{RA}}, sk_{\mathsf{RA}}) \leftarrow \mathsf{RAKeyGen}(1^k)$ is the key generation algorithm for the registration authority.

2. $(pk_{\mathsf{O}}, sk_{\mathsf{O}}) \leftarrow \mathsf{OKeyGen}(1^k)$ is the key generation algorithm for the an organization O.

3. $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) \leftarrow \mathsf{UKeyGen}(1^k)$ is the key generation algorithm for users. We call pk_{U} the (master) public key of U, and sk_{U} the master secret key of U.

4. $\langle (si_{\mathsf{regtick}}, \mathsf{W}_{\mathsf{U}}^{\mathsf{att}}, \mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}}), (\mathsf{regtick}, \mathsf{DB}_{\mathsf{RA}}{}')\rangle / \langle \perp, \perp\rangle \leftarrow$

   $$\leftarrow \mathsf{RARegistration}(pk_{\mathsf{RA}})[\mathsf{U}(sk_{\mathsf{U}}), \mathsf{RA}(sk_{\mathsf{RA}}, \mathsf{DB}_{\mathsf{RA}})]$$

   is the registration of a user to the registration authority RA. The common output of this procedure is U's public registration and recovery information $\mathsf{regtick}$ and $\mathsf{RegInfo}_{\mathsf{U}}$

respectively. U's private output is the corresponding secret information si_{regtick}, and wallets of various credentials $\mathsf{W}_{\mathsf{U}}^{\mathsf{att}}, \mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}}$. RA's private output is information related to U's (and credentials') blacklisting and U's activity tracing which are stored in $\mathsf{DB}_{\mathsf{RA}}$.

5. $\langle si_{\mathsf{att}}^{\mathsf{U}}, \mathsf{CLog}_{\mathsf{att}} \rangle \leftarrow \mathsf{GrantAtt}(\mathsf{att}, \mathsf{gpk}_{\mathsf{att}})\ [\mathsf{U}(pk_{\mathsf{U}}, sk_{\mathsf{U}}),\ \mathsf{att}(\mathsf{gsk}_{\mathsf{att}})]$. This interactive procedure between a user and an attribute service of RA and the user U to generate a certificate for ownership of attribute att. The output for the user is the secret information $si_{\mathsf{att}}^{\mathsf{U}}$ regarding the att-credential $\mathsf{cred}_{\mathsf{att}}$. The output for the organization is some information $\mathsf{CLog}_{\mathsf{att}}$ for the credential.

6. $\langle \top, \top \rangle / \langle \bot, \bot \rangle \leftarrow \mathsf{ShowAttribute}(pk_{\mathsf{V}}, pk_{\mathsf{RA}}, \mathsf{att})\ [\mathsf{U}(si_{\mathsf{att}}^{\mathsf{U}}, sk_{\mathsf{U}}), \mathsf{V}(sk_{\mathsf{V}})]$. A user U shows possession of attribute att to a verifier V whose secret public and secret identification information is pk_{V} and sk_{V} respectively.

7. $\langle \top, \mathsf{SID} \rangle / \langle \bot, \bot \rangle \leftarrow \mathsf{UserAuthenticate}(\mathsf{BL})[\mathsf{U}(sk_{\mathsf{U}}, si_{\mathsf{regtick}})\mathsf{V}(sk_{\mathsf{V}})]$, where a user U proves to a verifier V that he is not among the users in the blacklist BL. The result for the V is the transcript of U-authentication SID.

8. $\langle (\mathsf{W_U}', \mathsf{S}, \pi), (\mathsf{S}, \pi) \rangle / \langle \bot, \bot \rangle \leftarrow \mathsf{CredDemonstrate}(pk_{\mathsf{RA}}, pk_{\mathsf{V}})[\mathsf{U}(sk_{\mathsf{U}}, \mathsf{W_U}), \mathsf{V}(sk_{\mathsf{V}})]$. A user U, using a wallet $\mathsf{W_U}$ of some type of registration credentials of his ($\mathsf{perm/cred/rev}$), demonstrates ownership of one of them to a verifier V. Here S is the serial number of the credential and π is the valid proof of credential's ownership.

9. $\langle (\mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}'}, \mathsf{MemSec}_{\mathsf{U}}^{\mathsf{O}}), (\pi, \mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}}, \mathsf{SID}, \mathsf{AuthData}, \mathsf{RecData}, \mathsf{D_O}') \rangle / \langle \bot, \bot \rangle \leftarrow$

$$\leftarrow \mathsf{OrgRegistration}(pk_{\mathsf{O}}, pk_{\mathsf{RA}}, \mathsf{BL}_{\mathsf{RA}})[\mathsf{U}(sk_{\mathsf{U}}, si_{\mathsf{regtick}}, \mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}}), \mathsf{O}(sk_{\mathsf{O}}, \mathsf{D_O})]$$

is the registration of a user U to an organization O. U's private input consists of his master secret sk_{U}, the secret related to his RA-registration si_{regtick}, as well as the wallets of his credentials $\mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}}$. His private output is secret information regarding his O-membership $\mathsf{MemSec}_{\mathsf{U}}^{\mathsf{O}}$. The common output consists of U's membership public information $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}}$, and information to enable U to authenticate himself in case of IDC-loss $\mathsf{AuthData}$. O obtains U-blacklisting information SID, credential validity

information π and updates his members' database $\mathsf{D_O}$ with the U's membership data: $\mathsf{MemPub_U^O}$, π, SID, AuthData. Both parties also obtain obtain RecData, which will be anonymously uploaded to $\mathsf{DB_{MS}}$ by O.

10. $\langle(\mathsf{TempSec}, \mathsf{W_U^{rev}}, \mathsf{RegInfo_U}), (\mathsf{TempPub}, \mathsf{BL_{RA}}')\rangle/\langle\bot, \bot\rangle \leftarrow$

$$\leftarrow \mathsf{LossReport}(pk_{\mathsf{RA}}, pk_{\mathsf{U}}, \mathsf{regtick})[\mathsf{U}(sk_{\mathsf{U}}, si_{\mathsf{regtick}}), \mathsf{RA}(sk_{\mathsf{RA}}, \mathsf{BL_{RA}})],$$

where a user U reports the loss of his IDC, recovers his registration data $\mathsf{RegInfo_U}$, and blacklists his membership credential regtick. U obtains revocation credentials $\mathsf{W_U^{rev}}$ and a temporary RA-membership (TempPub, TempSec) to update his registrations.

11. $\langle\top, (\mathsf{BL_{RA}}')\rangle/\langle\bot, \bot\rangle \leftarrow \mathsf{RegBlackList}(pk_{\mathsf{RA}}, \mathsf{regtick})[\mathsf{O}(sk_{\mathsf{O}}, \mathsf{SID}, proof), \mathsf{RA}(sk_{\mathsf{RA}}, \mathsf{SID})]$, where an organization O blacklists a userU with session id SID, who has misbehaved. O provides the session id of the misbehaving party along with proof of his misbehavior.

12. $\langle\top, \mathsf{BL_{RA}}'\rangle/\langle\bot, \bot\rangle \leftarrow \mathsf{AttBlackList}(\mathsf{S^{att}})[\mathsf{U}(sk_{\mathsf{U}}, \mathsf{TempSec}, \mathsf{W_U^{rev}}), \mathsf{RA}(sk_{\mathsf{RA}}, \mathsf{BL_{RA}})]$, which takes place in the case of IDC-loss. A user U users his temporary RA-registration and his revocation-credentials to authorize himself to order the attribute services to invalidate all of U's attribute-credentials.

13. $\langle(\mathsf{O}, \mathsf{RecData}), \top\rangle/\langle\bot, \mathsf{BL_{RA}}'\rangle \leftarrow \mathsf{OrgRegistrationRecovery}$

$$(pk_{\mathsf{RA}})[\mathsf{U}(\mathsf{TempSec}, sk_{\mathsf{U}}, \mathsf{W_U^{rev}}, \mathsf{S}), \mathsf{DB_{MS}}(\mathsf{BL_{RA}}, \mathsf{RecData})]$$

In this procedure, the user U utilizes his revocation permissions and the recovered serial numbers to trace the organizations he has registered to. U uses a piece of information contained in RecData to authenticate himself and immediately advertise blacklisting information for O.

14. $\langle\mathsf{MemSec_U^{O'}}, (\mathsf{BL_O}', \mathsf{MemPub_U^{O'}})\rangle/\langle\bot, \bot\rangle \leftarrow \mathsf{OrgRegistrationUpdate}$

$$(pk_{\mathsf{RA}}, \mathsf{RecData})[\mathsf{U}(sk_{\mathsf{U}}, \mathsf{TempSec}, \mathsf{W_U^{rev}}), \mathsf{O}(sk_{\mathsf{O}}, \mathsf{AuthData}, \mathsf{D_O})],$$

where the user U contacts the organization to change his credentials. U authenticates himself using RecData and the corresponding AuthData stored in $\mathsf{D_O}$. He then obtains new membership credentials.

15. $\langle \top, \mathsf{DB_{RA}}' \rangle / \langle \bot, \bot \rangle \leftarrow \mathsf{CredValidityCheck}(pk_{\mathsf{RA}}, \mathsf{S}, \pi)[\mathsf{E}(sk_{\mathsf{E}}), \mathsf{RA}(sk_{\mathsf{RA}}, \mathsf{DB_{RA}})]$. An entity E (which may be either a user or an organization) deposits the credential (S, π) to the RA, to check its validity. If the the credential (S, π) is valid and not double-used, then the credential is stored in the history database of $\mathsf{DB_{RA}}$.

16. $(pk_{\mathsf{U}}, \Pi_G) / \bot \leftarrow \mathsf{Identify}(\mathsf{S}, \pi_1, \pi_2)$. If a perm or cred credential is double-used with (S, π_1) and (S, π_2), the RA can find the person who double-used a credential using this operation. Here, Π_G is a proof that pk_{U} double-used the credential with the serial number S.

17. $\top / \bot \leftarrow \mathsf{VerifyGuilt}(\mathsf{S}, \Pi_G, pk_{\mathsf{U}})$ outputs $\top$ if the user U (represented by pk_{U}) indeed double-used the credential with the serial number S.

5.1.3 Requirements

Privacy and *security* are our system's core requirements, which we adjust to the context of a centralized identity management system.

Privacy consists of *user anonymity* and *user activities' unlinkability. User anonymity* requires that

1. given the transcript of the demonstration of ownership of a RA-membership credential that does not belong to a corrupted party SID, the adversary can learn which user owns SID no better than guessing at random among all non-corrupted users that appear consistent with SID.

2. given the transcript of proof of ownership of a perm/cred credential by a non-corrupted user, $\mathsf{CredTrans}(\pi, \mathsf{S})$ the adversary can learn which user owns $\mathsf{CredTrans}$ no better than guessing at random among all non-corrupted users.

3. given a public organization membership information $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}}$ that does not belong to a corrupted party, the adversary can learn which user owns $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}}$ no better than guessing at random among all non-corrupted users that appear in the system.

4. given the transcript of the demonstration of ownership of an attribute AttTrans, that does not belong to a corrupted party, the adversary may identify the user who owns AttTrans no better than guessing at random among all non-corrupted users who have been granted that attribute.

5. assuming that RA is not corrupted, given the transcript of the authentication procedure at the OrgRegistrationRecovery procedure of a non corrupted user, the adversary cannot learn which user participates in that procedure no better than guessing at random among all non-corrupted users; if we now include RA in the adversary, user anonymity is satisfied if the adversary cannot distinguish the user among all honest users who have run the LossReport procedure.

6. assuming that RA is not corrupted, given the transcript of OrgRegistrationUpdate operation of a non-corrupted user U with an organization O, the adversary cannot learn U's identity no better than guessing at random among all non-corrupted users; if we now include RA in the adversary, user-anonymity is satisfied when the same holds among users who have run the LossReport procedure.

User activity unlinkability requires that activities of the same user cannot be linked as been committed by the same individual. In particular, we require:

- given the transcripts of two UserAuthenticate procedures SID_1 and SID_2, that do not belong to a corrupted party, the adversary has no advantage in telling whether SID_1 and SID_2 belong to the same user or not.

- given the transcripts of two executions of ShowAttribute procedure for the same attribute att $\mathsf{AttTrans}_1$ and $\mathsf{AttTrans}_2$ that do not belong to a corrupted party, the adversary has no advantage in telling whether $\mathsf{AttTrans}_1$ and $\mathsf{AttTrans}_2$ belong to the same user or not.

- given the transcripts of proof of ownership of two perm/cred/rev credentials (CredDemonstrate operation) $\mathsf{CredTrans}_1$ (π_1, S^1) and $\mathsf{CredTrans}_2$ (π_2, S^2) that do not belong to corrupted parties, the adversary has no advantage in telling whether $\mathsf{CredTrans}_1$ and $\mathsf{CredTrans}_2$ belong to the same user or not.

- given two memberships $\mathsf{MemPub}_{\mathsf{U}_1}^{\mathsf{O}_i/\mathsf{RA}}$ and $\mathsf{MemPub}_{\mathsf{U}_2}^{\mathsf{O}_j/\mathsf{RA}}$ to any organizations O_i, O_j or the RA, that do not belong to corrupted parties, the adversary has no advantage in telling whether $\mathsf{MemPub}_{\mathsf{U}_1}^{\mathsf{O}_i/\mathsf{RA}}$ $\mathsf{MemPub}_{\mathsf{U}_2}^{\mathsf{O}_i/\mathsf{RA}}$ belong to the same user or not.

- given the transcripts of two OrgRegistrationRecovery or OrgRegistrationUpdate operations, not performed by corrupted users, the adversary has no advantage in deciding whether they belong to the same user or not.

In any case, *privacy* is conditional on proper user behavior. On the other hand, *security*, consists of:

- *Strong authentication*; no party can lie about his identity. It consists of *credential unforgeability*, *credential non-transferability* and *mis-authentication resistance*:

  - *Credential unforgeability* requires that no user or coalition of users may issue RA-membership credentials or any other type of credentials (perm/cred/rev) such that the UserAuthenticate and CredValidityCheck accept.

  - *Mis-authentication resistance* requires that the probability that UserAuthenticate accepts when a user does not have valid RA-membership is negligible, as well as the probability that UserAuthenticate rejects when the user has valid membership credentials.

  - IDC *and Credential non-transferability* implies that no one but the legal owner of the card may demonstrate ownership of the card or possession of the corresponding credentials. More specifically, assuming two users U_1 and U_2 and a credential cred_1 of U_1, we require that the probability that U_2 runs CredValidityCheck, CredDemonstrate or UserAuthenticate using cred_1 successfully is negligible.

- *Accountability*; misbehaving parties are punished. More specifically, we assume a misbehaving user U with an RA-membership regtick. Accountability requires that the probability that UserAuthenticate accepts is negligible.

- *Unframability - Denial of Service attack resistance*; no party should be able to frame another user for misbehavior or cause malfunction of his card's credentials. In particular, we require that no coalition of users, even with the collaboration of the RA, can forge a

proof Π_G that $\mathsf{VerifyGuilt}(pk_{\mathsf{U}}, S, \Pi_G)$ accepts where pk_{U} is an honest user U's public key who did not double-used a credential with the serial number S.

- *Forward secrecy*, even if IDC is compromised, no entity or collaboration of entities, except for the IDC's owner can learn his memberships or activities.

Because our system is intended for real-world use, we also have the requirement of *Deployability*. This is partially expressed in our threat model and security requirements, which are made to reflect a real world environment. Furthermore, we require that our protocols scale for large systems.

5.2 A centralized privacy-preserving credential system

Moving on to the details of our protocols, for the system **setup**, the RA generates through $\mathsf{PS.OKeyGen}$ a digital signature key-pair $(pk^s_{\mathsf{RA}}, sk^s_{\mathsf{RA}})$ to identify itself. RA also generate a blind signature key-pair $(pk^b_{\mathsf{RA}}, sk^b_{\mathsf{RA}})$) and runs $\mathsf{EC.Setup}$ multiple times: once for each digital cash-based credential system (perm, cred, rev credentials) and once for each attribute supported. In addition, it runs $\mathsf{BAC.Setup}$ and for the setup of users' RA-registration credentials ($\mathsf{regtick}$).

All the organizations of the system generate a digital signature key-pair to be identified in the system. Let $(pk_{\mathsf{O}}, sk_{\mathsf{O}})$ denote the O-identification key-pair. Organizations' setup depends entirely on the level of accountability they want to enforce:

- Important Financial Organizations, i.e., banks, employers, require high level of both security and privacy. It is thus reasonable to assume that (if adopting privacy) they are implemented as the credential systems of [Camenisch and Lysyanskaya, 2001; Camenisch and Lysyanskaya, 2002a], which combines strong transaction privacy (complete transaction unlinkability of honest individuals) with strong accountability in case of misbehavior (recovery of misbehaving user's master secret). See section 4.3 for more details. Under this context, each organization O runs the $\mathsf{PS.OKeyGen}$ procedure to generate its identification key-pair $(sk_{\mathsf{O}}, pk_{\mathsf{O}})$.

- Non financial Organizations, i.e., online magazines, which do not require any level of security, user may register multiple times, but whose privilege of extending the pre-

scription can be revoked if the user misbehaves, i.e., he uses bad language in a forum. In this case, the identity of the users is not required for their registration. Organizations of this type may be implemented as group signature systems with membership revocation enabled, and — thus — run $\mathsf{GS.Setup}$ to generate group administration information: $(\mathsf{gpk}^{\mathsf{O}}, \mathsf{gsk}^{\mathsf{O}})$.

- Other Financial Institutions, who do not involve large monetary amounts and require a reasonable degree of accountability, i.e, any type of online service providers, may choose any of the aforementioned systems: blacklistable anonymous credentials, pseudonymous systems or group signature schemes.

In what follows we will use the following notation:

- $\mathsf{Sig}_{\mathsf{E}}(\mathsf{M})$ ($\mathsf{Sig}^{\mathsf{H}}_{\mathsf{E}}(\mathsf{M})$) for the signature of entity E on M ($\mathsf{H}(\mathsf{M})$).
- $\mathsf{GSig}^{(\mathsf{H})}_{\mathsf{g},\mathsf{E}}(\mathsf{M})$ for the g-group-signature of entity E on M ($\mathsf{H}(\mathsf{M})$).
- $\{\mathsf{M}\}_K$ for the encryption of M under key K. For efficiency, we induct every asymmetric encryption a symmetric one. Therefore, $\{\mathsf{M}\}_{PK}$ denotes $\{K\}_{PK}||\{\mathsf{M}\}_K$ for a random K.

5.2.1 RA Registration

This procedure takes place between the RA and the user U, who requests to enter the system. After providing strong identification credentials, i.e., birth certificates, passports, etc., U runs $\mathsf{EC.UKeyGen}$ or $\mathsf{PS.UKeyGen}$ to generate his master secret ms_{U} and engages with the RA to the following series of interactions:

1. ms_{U}-*Validation.* $\mathsf{U} \leftrightarrow \mathsf{RA}$: run $\mathsf{PS.FormNym}$ and $\mathsf{PS.GrantCred}$ procedure, to generate a pseudonym and a credential for U which is blindly linked to ms_{U}. The RA stores in $\mathsf{DB}_{\mathsf{RA}}$ all the public ms_{U}-information, $\mathsf{pub}_{\mathsf{U}}$.
2. *Registration Credential's Issue.* $\mathsf{U} \leftrightarrow \mathsf{RA}$: run $\mathsf{BAC.Register}$, to grant U a registration credential, $\mathsf{regtick}$, which will serve blacklistability purposes.

3. *Credentials' Issue.* U $\leftrightarrow$ RA: run EC.Withdraw operation twice to generate wallets of perm-credentials and cred-credentials. Both types of credentials are generated with the use of U's $\mathsf{ms_U}$ and serve authentication purposes for user registrations to organizations: the perm-credentials are realized as accountable ecash [Camenisch *et al.*, 2006], where the constant N is set to two (see section 4.1) and aim for organizations where users may have at most one entry; the cred-credentials, are implemented as any eCash scheme and may be used for any entity or organization. The key attributes of these credentials are that they are non-transferable as their use is strongly connected to $\mathsf{ms_U}$-knowledge, enable anonymous user-authentication, while through their serials provide a degree of traceability to their owner. The latter is particularly useful when their owner loses his card and wants to trace his previous activities.

4. *Recovery Mechanism Setup*, where U creates and stores some credential and master secret recovery information:

   (a) U generates his recovery encryption key pair ($\mathsf{ek_U}$, $\mathsf{dk_U}$).

   (b) U encrypts the serials of both types of credentials into

   $$\mathsf{RegInfo_U} = \{\mathsf{ms_U}, \text{perm-serials}, \text{cred-serials}, \text{att-serials}, \text{date}\}_{\mathsf{ek_U}},$$

   where att-serials are related to attribute-credentials (see section 5.2.3).

   (c) U$\rightarrow$ RA: $\mathsf{RegInfo_U}$; both entities agree on a hash H and exchange proofs of the final form of $\mathsf{RegInfo_U}$: $\mathsf{Sig}^H_{\mathsf{RA}}(\mathsf{RegInfo_U}||\text{date})$ and $\mathsf{Sig}^H_{\mathsf{U}}(\mathsf{RegInfo_U}||\text{date})$.

   (d) U shares in a shared secret fashion[Shamir, 1979] his $\mathsf{ms_U}$ and $\mathsf{dk_U}$.

5.2.2 Organization Registration

This procedure takes place between an organization O_i and a user U, and allows U to obtain membership in O_i. Depending on its setting, O_i may restrict user-registrations to one per user. Thus, depending on the case, membership pre-requisites of O_i may include proof of perm or cred credentials, U's identity or ownership of attributes. We emphasize on the fact that as perm-credentials have the form of accountable ecash [Camenisch *et al.*, 2006] (see,

4.1) if more than two perm- credentials of U are used for the same organization-merchant, U's identity is revealed.

1. U *Authentication.* $\mathsf{U} \leftrightarrow \mathsf{O}_i$:

   (a) run BAC.Authenticate for U to prove that he is among the valid users of the system. Let SID be the transcript of this demonstration.

   (b) engage in an EC.Spend procedure for U to bind one credential of his to O_i. As mentioned before, if there is a restriction regarding how many registrations a user should maintain in O_i, a perm-credential should be used; otherwise, a cred - credential may be used. Let $\mathsf{S}^{\mathsf{perm/cred}}_{\mathsf{U}\rightarrow\mathsf{O}_i}$ denote the credential's serial. It is apparent that if U uses the same credential twice or more than two perm-credentials for the same organization, his identity will be revealed.

2. *Actual Registration.* As mentioned before, the exact procedures that take place in this step depend on the type of the organization and on the level of accountability the latter wants to enforce. For strongly accountable systems, U and O_i run PS.FormNym and PS.VerifyCredOnNym for U to create a pseudonym to O_i, which is blindly connected to his actual master secret. In the general case, U and O_i interact so that the former obtains his secret membership information $\mathsf{MemSec}^{\mathsf{O}_i}_{\mathsf{U}}$ and O_i the corresponding public information $\mathsf{MemPub}^{\mathsf{O}_i}_{\mathsf{U}}$. Henceforth, for simplicity, we assume that U is known to O_i as P_{U}.

3. *Second Level Authentication Mechanism.* In this phase, the user creates and stores information locally which will enable him to authenticate himself and manage his organization credentials in the case where he loses his card and all his organization membership information:

   (a) U creates an O_i-specific recovery encryption key pair: $(\mathsf{ek}^{\mathsf{O}_i}_{\mathsf{U}}, \mathsf{dk}^{\mathsf{O}_i}_{\mathsf{U}})$, computes

   $$\epsilon^{\mathsf{dk}}_{\mathsf{P_U}} = \{\mathsf{dk}^{\mathsf{O}_i}_{\mathsf{U}}\}_{\mathsf{ek_U}}.$$

   (b) Ugenerates secret and computes $H_{\mathsf{O}_i}(\mathsf{secret})$, and

   $$\epsilon^{\mathsf{sec}}_{\mathsf{P_U}} = \{\mathsf{secret}\}_{\mathsf{ek}^{\mathsf{O}_i}_{\mathsf{U}}}, \quad \sigma^{\mathsf{sec}}_{\mathsf{P_U}} = \mathsf{Sig}^{H\mathsf{O}_i}_{\mathsf{P_U}}(\mathsf{secret}),$$

where H_{O_i} is an O_i-specific hash.

(c) $\mathsf{U} \rightarrow \mathsf{O}_i$: $\mathsf{AuthData} = \{\epsilon^{\mathsf{dk}}_{\mathsf{P_U}}, \epsilon^{\mathsf{sec}}_{\mathsf{P_U}}, \sigma^{\mathsf{sec}}_{\mathsf{P_U}}\}$. U is the only one who knows secret. AuthData will be used to authenticate U, when the latter loses his credentials.

Finally, O_i creates U's entry in his $\mathsf{DB}_{\mathsf{O}_i}$, where he stores the following:

$$\mathsf{Entry}^{\mathsf{O}_i}_{\mathsf{U}} = \{\mathsf{S}^{\mathsf{perm/cred}}_{\mathsf{U}\rightarrow\mathsf{O}_i}, \mathsf{MemPub}^{\mathsf{O}_i}_{\mathsf{U}}, \mathsf{AuthData}\}.$$

5.2.3 Attributes Credentials Issue

This is the procedure by which a user obtains attribute credentials (e.g., proof of age, medical status, marital status, etc.). We assume that for each possible attribute there is a separate service-group that U can visit right after his RA-registration. For each of these, RA runs GS.Setup during **setup**. After U proves to RA that he corresponds to $\mathsf{pub_U}$ and that he is entitled of attribute att_i, the following take place:

1. RA$\leftrightarrow$ U: run EC.Withdraw for U to obtain one ecoin-token $\mathsf{AttTick}_i$, using the att_i-related ecash-setting. RA updates U's entry in $\mathsf{DB_{RA}}$ accordingly.

2. U$\leftrightarrow$ att_i-Service:

   (a) U-*Authentication.* U runs BAC.Authenticate to prove that he has registered to RA and that he is not among the blacklisted users in $\mathsf{BL_{RA}}$. Let AuthTrans be the authentication transcript.

   (b) U-*Membership to* att_i-Service-*group.* U $\leftrightarrow$ att_i-Service: run GS.Join, for U to obtain membership to the corresponding group of attribute-possessors. Thus, U issues group membership key-pairs $(\mathsf{gsk}^{\mathsf{att}_i}_{\mathsf{U}}, \mathsf{gpk}^{\mathsf{att}_i}_{\mathsf{U}})$.

   (c) *Registration Recovery Mechanism.* Similar to the organization registration procedure, att_i-RecData is generated for U to be able to authenticate himself and invalidate his membership when his master secret is compromised.

At the end of this procedure, att_i-Service stores the U-attribute related info:

$$(\mathsf{AuthTrans},\ \mathsf{AttTick}_i,\ \mathsf{gpk}^{\mathsf{att}_i}_{\mathsf{U}},\ \mathsf{att}_i\text{-RecData}).$$

5.2.4 Attributes Demonstration

This is the procedure by which a user U proves ownership of an attribute att to a verifier V. V may be either a person or an organization.

1. Both parties participate to generate a challenge R.
2. V downloads the updated att-group public information: $\mathsf{gpk}^{\mathsf{att}}$.
3. U uses his $\mathsf{gsk}^{\mathsf{att}}$ membership (GS.Sign) and sign the dated challenge:

$$\sigma_{\mathsf{att}} = \mathsf{GSig}_{\mathsf{att},\mathsf{U}}(R \,||\, \text{timestamp}).$$

4. V runs GS.Verify to validate the signature produced.

It is apparent that in case someone misbehaves, V may collaborate with the att-Service in GS.Open procedure to (locally) identify and remove the misbehaving group member from the group. In addition, att-Service, may blacklist that user's regtick using the AttTrans associated with the misbehaving $\mathsf{gpk}_{\mathsf{U}}^{\mathsf{att}_i}$.

5.2.5 Master Secret compromise - IDC Content Recovery

This procedure has three phases. In the first phase, a user U contacts the RA to report the loss or compromise of his IDC. To authenticate himself, provides strong identification credentials, i.e., birth certificates, passports, etc. Then U using RA-issued permissions contacts an external database $\mathsf{DB}_{\mathsf{MS}}$ to recover confidential information regarding his registrations, i.e., in which organizations he has registered to, so that he eventually contacts the latters to update his memberships. More specifically, the following take place:

1. U– RA*interaction.*
   (a) U $\rightarrow$ RA: authenticated report for IDC-loss or IDC-compromise.
   (b) U recovers his master secret ms_{U} and recovery dk_{U} using a shared secret recovery protocol[Shamir, 1979].
   (c) RA runs BAC.BLAdd to blacklist the old regtick. In this way, RA aims to prevent the attacker from registering using the card elsewhere.

(d) $\mathsf{RA}(\mathsf{DB_{RA}}) \rightarrow \mathsf{U}$: $\mathsf{RegInfo_U}$; U then uses the recovered $\mathsf{dk_U}$ to recover the serials of his credentials (perm, cred and AttTick).

(e) $\mathsf{U} \leftrightarrow \mathsf{RA}$: run EC.Withdraw once more to issue one-time-use revocation credentials, rev-credentials. Attribute specific rev-credentials, are issued according to the information in U's entry in $\mathsf{DB_{RA}}$, for U to invalidate his old attribute credentials.

(f) $\mathsf{U} \leftrightarrow \mathsf{att}_i$-Service: U anonymously authenticates himself to att_i-Service (through att_i-specific rev-credentials) as the owner of a compromised master secret. After having recovered his $\mathsf{AttTick}_i$ serial number, U uses RecData, to authorize himself to order the blacklisting of the corresponding att_i-group membership information. Both parties collaborate so that U obtains a att_i-Service blind confirmation of their interaction, aimed for RA.

(g) $\mathsf{U} \leftrightarrow \mathsf{RA}$: run the RA Registration procedure again for U to generate a temporary $\mathsf{ms_U}$ and IDC. RA updates U's entry in $\mathsf{DB_{RA}}$.

2. U– $\mathsf{DB_{MS}}$ *interaction.* In this procedure, U recovers the list of his registrations, i.e., which organizations he has registered with his old $\mathsf{ms_U}$. This will be covered extensively in the following subsection.

3. U– O_i *interaction.* For each organization O_i, U has registered to, the following series of procedures take place:

(a) $\mathsf{U} \leftrightarrow \mathsf{O}_i$: run EC.Spend procedure, for U to bind one of his rev-credentials to O_i. It is important to note that because of their ecash nature, rev-credentials are also unlinkable to U's identity, nevertheless non-transferable. U shows the serial $\mathsf{S}^{\mathsf{perm/cred}}$, which he used to register to O_i.

(b) O_i: checks rev-credential's validity; if valid, O_i looks up $\mathsf{S}^{\mathsf{perm/cred}}$ up in its $\mathsf{DB}_{\mathsf{O}_i}$.

(c) $\mathsf{O}_i \rightarrow \mathsf{U}$: ϵ^{sec}.

(d) U: uses the recovered $\mathsf{dk_U}$ to decrypt his secret and demonstrates knowledge of it to O_i.

(e) $\mathsf{U} \leftrightarrow \mathsf{O}_i$: run BAC.Authenticate procedure for the temporary membership of U. The transcript of this interaction SID' will serve blacklistability purposes in cases

U misbehaves. Note that the blacklist corresponding to the temporary master secrets is considerably smaller than the regular one.

(f) O_i blacklists all the credentials issued for the $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}}$ — using a technique similar to [Camenisch and Lysyanskaya, 2002a; Tsang *et al.*, 2007] or [Nakanishi and Funabiki, 2006], depending on its setting.

(g) U contacts O_i using his new credentials to open a new privacy preserving account.

4. U – RA *interaction.* U collaborates with RA in a BAC.Authenticate procedure for all the recently blacklisted items for both his old and temporary RA-registrations. Assuming that organizations report accounts that have not been accessed for a month, in this way, we want to avoid cases where the user falsely claimed loss of his IDC to erase accounts of his without being traced. After being cleared, both repeat a procedure similar to the RA registration for U to issue his new credentials.

5.2.6 User Registrations' Recovery

In this stage, a user U who has already reported the loss of his IDC to the RA, recovers his registrations, i.e., information regarding which organizations he has become a member to using his old ms_{U}. The registrations' recovery procedure is subjected to many caveats. First of all, aiming to maintain U's privacy towards the RA, the serial numbers of the withdrawn credentials are only visible to U and — when eventually used — to the organization they are used for. Revealing the serials to the RA would enable the latter with the collaboration of all organizations to trace U's activities. However, there should be measures to prevent U from claiming ownership of false numbers and causing DoS to honest users who own these numbers. Therefore, we introduce an external database $\mathsf{DB}_{\mathsf{MS}}$ used strictly by authorized organizations or users to upload registration related information. We assume that users maintain anonymous accounts with $\mathsf{DB}_{\mathsf{MS}}$ and can read $\mathsf{DB}_{\mathsf{MS}}$-data from their accounts only through rev-credentials. The registrations' recovery protocol includes two phases, one which takes place after each registration, and the actual recovery procedure. For the purposes of these protocols, we assume that there are two serial-number–specific hash functions $\mathsf{H}_{\mathsf{serial}}$ and $\mathsf{H}_{\mathsf{intserial}}$ and that the user chooses a OWF function F with id number FID from a

public pool of OWFs. The series of actions are the following:

1. *Recovery Setup.* It takes place at the end of each user U- organization O_i registration. Let $\mathsf{S}^{\mathsf{perm/cred}}$ be the serial number of the credential U used (see subsection 5.2.2). The following take place:

   (a) $\mathsf{U} \leftrightarrow \mathsf{O}_i$: choose two hash functions F_1 and F_2 from the pool of hashes with id numbers $FID1$ and $FID2$ respectively.

   (b) $\mathsf{U} \rightarrow \mathsf{O}_i$:

   $$\epsilon^{\mathsf{ser}} = \{\mathsf{S}^{\mathsf{perm/cred}}\}^{\mathsf{H}_{\mathsf{serial}}}_{\mathsf{K_U}},\ \epsilon^{\mathsf{ser},f} = \{\mathsf{H}_{\mathsf{intserial}}(\epsilon^{\mathsf{ser}}) \mid\mid FID,\ FID1,\ FID2\}_{\mathsf{K_U}},\ \text{and } FID,$$

   where $\mathsf{K_U}$ is a $\mathsf{ek_U}$-generated symmetric key.

   (c) O_i, U: compute $\epsilon^{\mathsf{org}}_{\mathsf{O}_i} = \{\mathsf{O}_i\}_{\mathsf{K}_f}$, where $\mathsf{K}_f = F(\mathsf{S}^{\mathsf{perm/cred}})$ is a $\mathsf{S}^{\mathsf{perm/cred}}$-generated key.

   (d) O anonymously uploads to $\mathsf{DB_{MS}}$:

   $$\mathsf{RecData} = \epsilon^{\mathsf{ser}} \mid\mid \epsilon^{\mathsf{ser},f} \mid\mid \epsilon^{\mathsf{org}}_{\mathsf{O}_i}.$$

2. *Serial LookUp* phase, which takes place after the IDC loss declaration phase. In particular

   (a) $\mathsf{U} \leftrightarrow \mathsf{DB_{MS}}$: collaborate in $\mathsf{BAC.Authenticate}$ and $\mathsf{EC.Spend}$ procedures for U to prove that he is a valid user who has reported the loss of his IDC. U makes use of his temporary $\mathsf{ms_U}$ for these purposes. Let SID be the transcript of it.

   (b) For each serial $\mathsf{S}^{\mathsf{perm/cred}}$ U has recovered, he computes the corresponding ϵ^{ser}, which he then uses as a lookup key in $\mathsf{DB_{MS}}$.

   (c) U decrypts the corresponding $\epsilon^{\mathsf{org}}_{\mathsf{O}_i}$ to recover the name of the organization O_i related to that serial.$FID1, 2$ are used for avoiding DoS attacks by users who try to blacklist credentials they never owned.

U may occasionally decide to publish through $\mathsf{DB_{MS}}$ the $\mathsf{S}^{\mathsf{perm/cred}}$ he has recovered. In fact, $\mathsf{DB_{MS}}$ publishes with U's collaboration $(\epsilon^{\mathsf{ser}}, F_1(\mathsf{S}^{\mathsf{perm/cred}})||F_2(\mathsf{S}^{\mathsf{perm/cred}}), \mathsf{SID})$ records. After its regular check O, freezes automatically the credentials that correspond to $\mathsf{S}^{\mathsf{perm/cred}}$'s

registration. If U fails to contact O within a prefixed time, the freezing stops and SID is sent to RA, who depending on the type of O and its policy may blacklist U.

5.3 Discussion

In this section we will illustrate how privacy and security are achieved in our system, while we discuss deployability issues.

5.3.1 Privacy-Security

The following theorem states the correctness, privacy and security of our general scheme:

Theorem. if the underlying primitives (anonymous credential system, e-cash system, blind signatures) are secure, then our scheme satisfies *correctness*, *user anonymity*, *user activity unlinkability*, *credential unforgeability*, *credential non transferability*, *mis-authentication resistance*, *user unframability*, *forward secrecy* and *accountability*.

We use prove this theorem with the following lemmas.

Lemma 1. If the underlying primitives (anonymous credential system, e-cash system, group signatures) are secure, then our scheme satisfies *Correctness*.

Lemma 2. If the underlying primitives (blacklistable anonymous credential system, ecash system) are secure, then our scheme satisfies *user-anonymity* and *user-activity unlinkability*.
Proof. Let that a user U has registered with the RA and obtained a registration membership $\mathsf{regtick}$ and wallets of $\mathsf{perm/cred}$ credentials $\mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}}$ respectively. In addition, U has obtained a wallet of att-related credentials and has been granted attributes $\mathsf{att}_1, \mathsf{att}_2$. U has also registered to organizations O_1 and O_2, with memberships $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}_1}$ and $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}_2}$ respectively.

Anonymity property of the blacklistable anonymous credential and ecash scheme used guarantee that $\mathsf{regtick}$ ownership demonstration(SID) and the honest use of $\mathsf{W}_{\mathsf{U}}^{\mathsf{perm/cred}}$ will

not reveal U's identity, which implies that $\mathsf{MemPub}_{\mathsf{U}}^{\mathsf{O}}$s are also unlinkable to U. In a similar way, the *anonymity* property of ecash schemes guarantee that U's identity is not known to att-service, while the same property of group signatures guarantees that demonstration of possession of any of $\mathsf{att}_1, \mathsf{att}_2$ (AttTrans) will not be linked to U's identity. *User Anonymity* in case of card loss is achieved through the blacklistable anonymous credential nature of the temporary registration credentials and the ecash nature of the revocation authorizations.

In a similar way, *unlinkability of user-activity* is satisfied through the unlinkability properties of the underlying ecash and blacklistable anonymous credentilas' schemes. More specifically the unlinkability property of blacklistable anonymous credentials guarantees that the transcripts of two UserAuthenticate procedures SID_1 and SID_2 of the same user would be unlinkable one to the other. In addition the unlinkability property of ecash and group signatures imply the unlinkability of two CredDemonstrate of revcredrev credentials or of two ShowAttribute operations for an attribute att performed by the same user. Consequently, organization memberships of the same user remain unlinkable as such. It is essential to note that as opposed to the unlinkability of credentials which is achieved even towards the RA-collaborations with organisations, the unlinkability of proofs of attribute ownership does not hold towards the RA, as the latter acts as the group manager of each attribute service. Nevertheless for the RA to construct a user's profile, the latter should collaborate with all the organizations, which is out of scope.

Lemma 3. If the underlying primitives (blacklistable anonymous credential system, ecash system) are secure, then our scheme satisfies *credential unforgeability*, *card non transferability* and *mis-authentication resistance*.

Proof. Credential unforgeability is directly satisfied through the *unforgeability* property of the underlying ecash and blacklistable anonymous credential system, while *mis-authentication resistance* is achieved through the *mis-authentication resistance* property of the blacklistable anonymous credentials, according to which the transcript of a demonstration of regtick ownership is enough to effectively blacklist a user. Non transferability property is implicitly achieved in our system. In particular, we adopt an "all or nothing" method according to which for a user U_1 to be able to lend his credentials to another user U_2, U_1 should reveal his master secret to U_2, which is a property supported by the underlying ecash schemes used.

Lemma 4. If the underlying primitives (blacklistable anonymous credential system, ecash system) are secure, then our scheme satisfies *accountability* and *user-unframability.*
Proof. It is directly achieved through the *Identification of Violators* and *exculpubility* properties of the anonymous ecash scheme, which guarantee that a user cannot be framed by any other party as, while if a user U uses the same credential twice or uses more that one perm credentials to the same organization, U's identity is revealed.

Lemma 5. If the underlying primitives (blacklistable anonymous credential system, ecash system) are secure, then our scheme satisfies *forward secrecy* and *resistance to DoS attacks.*
Sketch Proof. Accountability and Credential non- transferability provides a degree of protection against DoS attacks at the IDC-registration recovery phase. Users trying to blacklist credential serials of other users, will fail to identify the corresponding organization and FIDs; will thus be reported and blacklisted.

Forward secrecy in case of IDC-loss or compromise has been discussed in the previous section. Because of the *anonymity* property of ecash (rev-credentials) and anonymous credentials (TempPub), the RA, even when collaborating with organizations or $\mathsf{DB}_{\mathsf{MS}}$, cannot link a particular serial to a user. FID-based user authentication for the temporary serial organization-membership blacklisting does not reveal also any information leakage regarding U.

5.3.2 Deployability

It consists of applicability and scalability. Regarding applicablity, as shown in previous sections, we have taken in consideration "real world" in our threat model and architecture. Scalability is achieved through (a) the tree-structure of the suggested credential system, and (b) the scalability of the underlying primitives. As demonstrated before, the credential-issuing procedure of each user may be parallelized with a tree, whose root is the user's regtick and whose leaves are attribute or organization credentials. Each organization maintains local blacklists according to its policies and notifies the RA-blacklist when necessary. In this way, user-authentication does not create bottlenecks: for regtick-authentication

the user-perceived delay for a blacklist of 1600 entries is 4 seconds[Tsang *et al.*, 2007]. User-authentication against the global RA-blacklist, i.e., in cases when a user applies for a passport or a visa, does not create a bottleneck either since these procedures currently take much longer than a week. For attribute demonstration, [Nakanishi and Funabiki, 2006; Choi *et al.*, 2005] group signature schemes offer the possibility to prove group-membership in $O(1)$ time regardless of the size of the group. These schemes also provide the possibility for the attribute service to activate a user's group-membership at a later time from his request without the need of that user to update his card. This is particularly useful in cases of age credentials. In the recovery protocol case, the detection of the credential serials and their blacklisting are done immediately through the secret sharing protocol and the use of the hashes at the $\mathsf{DB}_{\mathsf{MS}}$.

5.4 Related Work

There has been some work indicating the problem of online privacy. Brands [Brands, 2000] and Camenisch and Lysyanskaya [Camenisch and Lysyanskaya, 2001] were the first to provide a big overview of privacy issues caused by the extended online use of PKI and provided a series of constructions of privacy preserving credentials, tickets and certificates based on blind signatures and zero knowledge proofs. There has been some work on blacklistable anonymous credentials [Camenisch and Lysyanskaya, 2002b; Tsang *et al.*, 2007; Akagi *et al.*, 2008]. Although the privacy provided in the aforementioned schemes is very strong, they do not refer to systems with multiple operations each requiring a different privacy level.

Centralized identity management systems applying the primitives of [Brands, 2000; Camenisch and Lysyanskaya, 2001] have been suggested in the past. In Idemix [Camenisch and Van Herreweghen, 2002] , Camenisch and Herreweghen developed additional functionality for service providers and credential issuers to configure and enforce resource access control and credential issuing decisions. Higgins [Foundation,], OpenID [Foundation and publish vision for open government through open identity technologies,] and iCard [Jiang *et al.*, 2003] Foundation are examples offrameworks handling many identities of the same

user across different websites. The PRIME project [Group,] is a European initiative for privacy preserving identity management for online commercial interactions. Although the existing work in the field refer to multiple types of user-interactions, they do not provide accountability when the user misbehaves, or consider real world issues deriving from master identity compromise, i.e., the complete recovery of the user's online subscriptions, automatic invalidation of the corresponding compromised credentials, advanced user-authentication to manage these operations etc.

5.5 Conclusion

In this chapter we presented a centralized, card-based identity management system which addresses many online and offline activities of individuals achieving different levels of privacy. It thus constitutes the core of the system presented in this dissertation. As opposed to most existing credential-based identity management systems, in our system the card is recoverable: when lost or compromised the card's legal owner may recover its content completely — his master identity and the subscriptions he has obtained through the latter — or even invalidate it. Including the cases of card loss or compromise, this card will protect an honest individuals anonymity when applicable as well as ensure his activity is known only to authorized users.

Chapter 6

Anonymous Browsing: A Key-Privacy Mechanism

Anonymous browsing is a prerequisite for enforcing privacy in any type of online communication. Considering it as a key mechanism for privacy in our identity management system, in this chapter we demonstrate ways of strengthening the anonymity provisions of current anonymizing networks.

Anonymous networking has been known since 1981 [Chaum, 1981]. A more practical scheme, Onion Routing, was first described in 1995 [Goldschlag *et al.*, 1996]. Currently there is little practical use of network anonymity systems. Some of the problem is undoubtedly sociological: most people do not feel the need to protect their privacy that way; this is one reason that companies such as Zero Knowledge Systems [Back *et al.*, 2001; Boucher *et al.*, 2000] and Digicash [Rudolf,] failed. Another problem, though, is that strong anonymity against traffic analysis requires cooperation by and implicit trust in many different parties. Any single entity, no matter how trustworthy it appears, can be subverted, whether by technical means, corrupt personnel, or so-called "subpoena attacks". All known solutions require, and in fact enforce, routing through multiple parties. This, though, introduces another problem: economic incentives. In a single-provider anonymity scheme, that problem is conceptually simple: the party desiring privacy pays a privacy provider. This payment can be protected by digital cash [Chaum *et al.*, 1988b]. Unfortunately, in a multi-provider

Mixnet or onion routing network, the problem is more complex, since each party must be paid. By examining existing digital cash schemes, we show that they do not provide the necessary cost or privacy properties required to maintain anonymity. For example, in Chaum's original e-cash scheme [Chaum *et al.*, 1988b] a double-spender's identity is exposed. This is perfectly acceptable – double-spending is a form of cheating that should be punished – but in the context of an onion routing network, detecting double spending gives an adversary clues to path setup.

To address these problems, we propose a novel hybrid payment scheme by combining features from Micali's micropayment system [Micali and Rivest, 2002] and a lightweight, blind signature-based e-cash scheme. Our goal is to create incentives for the network participants to act in a cooperative manner based on their personal interests. We show that any solution must be sound in several dimensions. First, it must protect privacy. This is not trivial; witness the many (partial) attacks on various anonymous networking protocols [Øverlier and Syverson, 2006; Kesdogan *et al.*, 2006a]. That said, we do not claim to have fixed those problems. Rather, our aim is avoid introducing any new vulnerabilities that stem from the payments scheme.

Second, we want a system that is in principle deployable. That is, though we assume such things as anonymous payment systems, we do not assume, for example, incorruptible banks. More importantly, we want a system that is compatible with known economic behavior. Therefore, while our system assumes that people are willing to pay for privacy, we want a system where customer payment – the profits of forwarding nodes – are related to privacy desired and effort expended. In essence, there must be a profit motive and the opportunity for market forces to work. To deter exploitation of the payment scheme, we provide mechanisms to detect cheaters: those parties who accept payment but do not provide services.

Third, we do not attempt to achieve absolute financial security. Instead, we are willing to accept small amounts of cheating, by senders or forwarders, as long as the amount is bounded and limited (possibly with some trade-off) by the party who is exposed to loss. Finally, we want a system that is acceptably efficient in practice and does not impose unreasonable resource consumption. To that end, we evaluate the operations of a prototype PAR – which

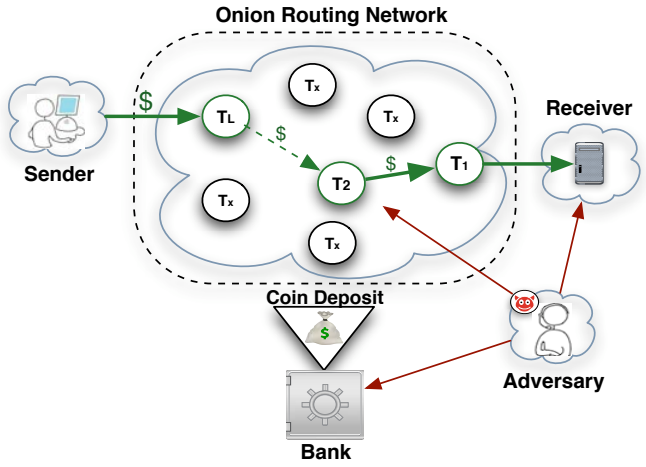

Figure 6.1: The PAR architecture combines an onion routing anonymity network (Tor) with a payment scheme. Each node $T_1, T_2, T_3, \cdots, T_L$, where L is the path length, in the path from the sender to the receiver receives payment in coins for its service.

stands for Payment for Anonymous Routing. Our initial performance evaluation indicates that PAR is highly configurable and can operate with acceptable communication and CPU overhead. As opposed to previous work on incentivised anonymity, which used mixnets ([Franz *et al.*, 1998], [Figueiredo *et al.*, 2003], [Reiter *et al.*, 2005]), our system guarantees usable efficiency, accountability and maintains anonymity against traffic analysis attacks.

6.1 System Considerations

We will examine current anonymizing networks and payment schemes and show why current payment schemes, when applied to onion routing schemes, fail to maintain anonymizing network properties, while our hybrid scheme succeeds. Furthermore, we set up the threat model and we identify the individual components and the properties required by a payment scheme to provide the same protection the network anonymity system was designed for.

6.1.1 Anonymizing Network

An anonymizing network is a particular type of peer-to-peer network, in which peers communicate anonymously. Anonymizing networks aim to offer sender anonymity even against the recipient as well as sender-receiver unlinkability. Neither the recipient nor any other

participant should be able to detect the actual sender with a better probability than selecting the sender at random. As a proof of concept, we use Tor [Dingledine *et al.*, 2004], the second generation onion routing anonymity network, a well-known and deployed network anonymity system.

6.1.2 Adversarial Model

The participating entities of our system are the *Tor relays*, the outside *users*, and a clearance entity, *i.e.* a *Bank*, where monetary units are deposited/withdrawn. We inherit Tor's local adversary model where users can only observe the traffic going through them and a limited amount of the rest of the network traffic. In addition, we assume that malicious users can manipulate any packet going through them and use this information to compromise anonymity. The Bank, on the other hand is assumed "honest but curious". Therefore, although trusted to be honest in all of its functional operations – cash withdraw and deposit – the Bank can collaborate with any number of users in order to disclose the initiator of a communication or active communication paths. We do not consider covert channels for anonymous communication with routers without paying as a part of our threat model.

6.1.3 System Requirements

Our primary requirement is that the overall system should maintain the *anonymity* provided by Tor even when the payment deposit information is exposed to a third party including the Bank. Anonymity, however, should not be achieved at the expense of efficiency. Moreover, the payment scheme should meet the requirement necessary for any payment system such as *accountability*, *correctness*, and *robustness*.

6.1.4 Payment Analysis

For our analysis, we classify current payment schemes in two categories: *Identity-bound payments* and *Anonymous payments*. In Figure 6.1, the sender provides payment for all nodes T_1, T_2, T_3, $\cdots$, T_L[1] that forward the sender's traffic to the receiver. We will show

[1]In Tor, intermediate communication path nodes are chosen randomly by the communication initiator.

that both of the current payment schemes, when applied to a Tor network, render the anonymity system vulnerable to attacks that compromise the anonymity of the senders.

Identity-bound Payment Schemes. Identity-bound payments constitute signed endorsements from the payer to the payee. Accountability and robustness are the two main features of this class. The micropayment scheme [Micali and Rivest, 2002] is an example of an Identity-bound payment. It was designed to be efficient for small, online transactions. When used to pay Tor nodes, identity-bound payments provide immediate accountability because invalid payments from any entity can be easily accounted for. However, when applied in the context of the Tor network, this property has adverse implications: upon clearance, the Bank obtains global knowledge about all transactions in the anonymity network. If the sender uses his own coins to pay the nodes in the path, his identity is exposed to them. Therefore, any node in the path to the receiver can identify him with the help of the Bank. To make things worse, the last node in the path – who may suspect that he is the last node if the receiver is outside Tor – can link the sender to the receiver. A potential way to work around this problem is to distribute payments only to immediate neighbors. With this payment strategy, the sender pays T_L with L coins, T_L pays T_{L-1} with $L-1$ coins etc. This approach makes path tracing much harder and leaks less information but it is far from secure: deposits made by the sender to the first Tor node are still available to the Bank. Counting the coins bound to the sender's identity, the Bank can infer with high confidence the number of packets communicated to the sender and link the sender to the receiver. This analysis indicates that having identity-bound coins reveals too much information, enabling an adversary with access to payment information to break the system's anonymity using simple inference techniques.

Anonymous Payment Schemes. In this scheme, the payment does not carry any identification information of its initial owner. Chaum's Digital cash [Chaum *et al.*, 1988b] and the later versions [Tunstall, 1989; Hayes, 1990; **?**] of Tunstall et al. and Camenisch et al. are perfect examples of such anonymous payment schemes. In the general case of digital cash systems, a user withdraws money from a Bank, which he can only spend himself and which

when legally spent can never be linked to his identity. Merchants deposit the coins they have received to check whether any of them has been spent more times than its nominal value (double-spending). If the later occurs, the identity of the double-spender is revealed. However, all the anonymous payment schemes demand excessive communication overhead for each transaction because there are a lot of messages that need to be exchanged between the sender and the path nodes.[2] This requirement makes e-cash schemes impractical for our system.

An alternative solution would be for all users to withdraw a special kind of anonymous coin from the Bank, which can simply be Bank blind endorsements [Okamoto, 2006], and use these coins to pay the intermediate Tor nodes. Ideal as it might initially seem, using a completely anonymous payment scheme with Tor has its drawbacks. First of all, there is no immediate accountability, since double-spending in this case will not reveal the double-spender. Thus, to prevent double-spending, any payments received should be immediately checked and deposited in the Bank. Unfortunately, immediate coin deposits could lead to deposit timing attacks exposing Tor's anonymity. More specifically, the timing of deposits by the nodes along a Tor path discloses to the Bank the path as well as an estimated of the number of packets transferred. Accumulating deposits for appropriately long time intervals – sufficiently long that many connections are established, to mitigate timing attacks – would increase the amount of unchecked coins and thus of double-spending. Indeed, since anonymous coins are not traceable beyond the first Tor node, sending valid coins only to the first node is enough to prevent it from been traced. For the rest of the nodes, the cheater uses double-spent coins, exploiting this deposit strategy by transmitting many packets in a short period of time.

Our Contribution: Hybrid Approach. Both of the two aforementioned classes of payment schemes have advantages and disadvantages. Our approach creates a hybrid payment scheme by combining the two payments methods into a single one. In particular, nodes out-

[2]In the compact e-cash payment scheme [Camenisch *et al.*, 2005], which is considered efficient a single "spend" procedure in e-cash systems would requires at least two rounds of message exchange between the sender and every node in the path.

side the anonymizing network withdraw an initial number of anonymous coins (A-mcoins) from the Bank and use them to pay the first node in the Tor-path (T_L) they have chosen. T_L then uses micropayments[3] to pay T_{L-1}, who also uses micropayments to pay its neighbor. Each time, the amount of money paid decreases according to each node's price. Nodes participating in the Tor network follow the same protocol with the option to use either anonymous or micropayments for the first node in their forwarding path.

In addition, each of the payment coins in the scheme has a corresponding receipt and becomes valid only when it is submitted for deposit together with the receipt. As we will show in the following sections, our payment scheme combines all the desirable properties of the existing payment schemes, but without maintaining any of the problem each one of them causes when used individually and in this way it provides sender-receiver unlinkability along with accountability and efficiency.

6.2 High-Level Description of PAR Protocol

Here we provide a high-level description of our payment scheme. To help the reader, we start with a brief description of the Tor circuit setup; we then present our payment scheme.

6.2.1 Tor

Tor is formed by a set of relay nodes (onion routers) that act as traffic indirection points. The region in the dotted lines in Figure 6.1 depicts a typical communication in Tor. Each onion router maintains a TLS [Dierks and Allen, 1999] connection to every other onion router. To establish communication, the sender selects a random sequence of Tor relays to form a path to the receiver or what is called a circuit. In Figure 6.1, the sender selected nodes $T_1, T_2, T_3, \cdots T_L$, where L is the path length. The sender constructs circuits incrementally, by negotiating a symmetric key with each onion router on the path, one hop at a time. Initially, the sender contacts the first path node, T_L, and they both commit in a Diffie Hellman (DH) key agreement procedure. Once this initial circuit has been created, sender uses T_L to extend the circuit to T_{L-1}. In particular, T_L and T_{L-1} establish a circuit

[3]Identity-bound payment

– through the TLS channel they share – which T_L relates to the one with the sender. Sender commits anonymously (using T_L as mediator) in a Diffie Hellman (DH) key exchange procedure with T_{L-1}. Repeating this process through the extended tunnel, the sender may add more Tor nodes to the circuit. At the final stage, the last node in the path, T_1, opens a data stream with the receiver and a regular TCP connection is established between the sender and the remote site's IP address. At the end of the circuit setup procedure, every relay in the path shares a secret key with the anonymous path initiator, as well as with each of his path neighbors. The key a path node shares with each of his neighbors is only used for securing their part in the communication path. Each transmitted Tor message along a path, contains an unencrypted header with a circuit ID and a multiply-encrypted payload. At each hop, the corresponding path node decrypts the payload – using the key that node and the sender share – and replaces the circuit ID with the one that corresponds to his circuit with next node in the path.

6.2.2 PAR

We introduce the hybrid payment scheme from the previous section to the Tor network; again, see Figure 6.1. In our scheme, payments are conducted between consecutive nodes on the forwarding paths and added inside the transmitted messages using an additional encryption layer. Each forwarding node T_i creates payment coins for its path successor T_{i-1} using sender S's directions and adds these payment coins to the onion message to be forwarded to T_{i-1}. Payment information is provided to each T_i through the secret channel it and the sender share. To avoid exposure as in Tor, T_i further encrypts the resulting message with the key it shares with its successor. To complete the payment transaction and for the coins to become valid, every relay node has to receive the receipts for its payment by its successor. Therefore, each node, other than the last one, upon validating the received message, sends to its predecessor the payment receipt. S controls the payments made along the forwarding path by supplying the receipts for all the coins used.

To avoid cheating, S provides each path node T_i with additional information for it to verify that the payment received from T_{i+1} is indeed valid. Receipts are forwarded to T_{i+1} if and only if the the payments are valid. Since the circuit is used in both directions (*i.e.*

to both receive and transmit messages, the last node can either be pre-paid or paid after the delivery of the message by the sender depending on the acceptable bounded risk. In either approach misbehaving nodes will be detected within the first round of sent messages and will be excluded from the forwarding path, which will cause them more loss than the expected gain from fraudulent behavior and they will have no incentive for cheating.

The initial setup stage for Tor circuits will be extended with nodes sharing some hash function that will be used prevent third party manipulations in the payment protocol.

6.3 A Hybrid Payment Scheme

In this section, we present a detailed description of our payment protocol. However, before proceeding, we first define three properties required to preserve anonymity in an onion routing network:

- *Sender-Receiver Unlinkability.* Let S be a user, who may or may not be a member of the anonymizing network, who sends a message M anonymously[4] to a user R. Then nobody except a global adversary, even with the collaboration of a third party and R, should be able to link sender and receiver or reveal the path between them.

- *Usable Efficiency.*This refers to the fact that the overhead in the packet exchange for the payment scheme and the CPU overload with additional cryptographic operations will be reasonable and will not impede the normal functioning of the system.

- *Accountability.* This property ensures that any cheating node trying to forge messages or double-spend coins is caught and expelled from the network.

6.3.1 Payment Coins

We use two types of payments that consist of two parts: a payment part, which we will call a coin, and a receipt part. A coin becomes valid only when it is accompanied by the corresponding receipt. The receipt is a random number that is bound to the coin by incorporating its hash value in the coin. Thus a random number r serves as a receipt for the

[4]Here, "anonymously" means "using the anonymizing network".

coin that contains the hash $H(r)$. Although similar in structure, the two types of payments have different properties and that is why they are named differently: micro-coins (*S-coins*) and anonymous coins (*A-coins*).

S-coins(Signed microcoins). S-coins are generated and used for payments between Tor participants. They are based on the micropayments introduced in [Micali and Rivest, 2002] but with the addition of receipts. An S-coin is an extension of a microcoin MC :

$$SC_{T_i \to T_j} = sig_{T_i}\{MC, H(r), T_j\}.$$

As in the microcoin case, an S-coin is strongly bound to both the identity of the node T_i, who generates it by signing its content, and the identity of the payee T_j. Finally, it contains the hash of the receipt $H(r)$ that makes the coin valid. The microcoin part of the S-coin MC contains the transaction details τ as well as a sequence number – according to micropayment scheme [Micali and Rivest, 2002] – without containing any timing information.

S-coins inherit the properties of microcoins. Only a predetermined fraction of them are payable, while no participants in the payment scheme can find out in advance which coins will become payable.

A-coins (Anonymous coins). A-coins use the idea of e-cash ([Chaum *et al.*, 1988b]). They are generated by the Bank upon users' requests. Users outside Tor buy a predetermined number of A-coins from the Bank and pay with them for using the anonymizing network. Members of Tor also acquire a number of A-coins and may also use them. All A-coins are of the form

$$AC(r) = sig_B\{r\},$$

where r is a random number generated by the User, and $sig_B\{r\}$ is the blind signature of the Bank of r. A-coins are all payable and subjected to double-spending checks.

6.3.2 Payment Protocol

Figure 6.2 presents in detail the messages exchanged in the payment protocol. We further analyze the individual protocol stages.

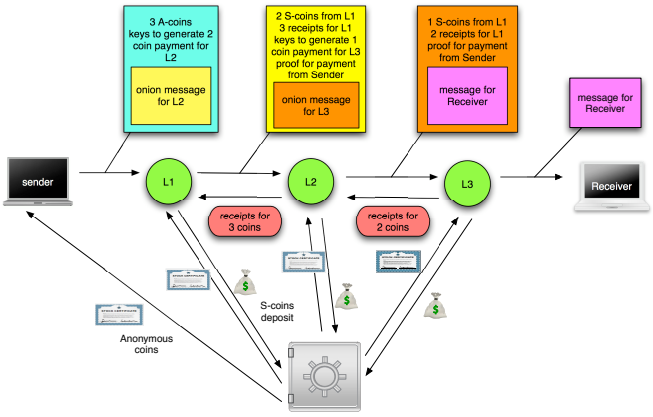

Figure 6.2: The intuition behind our payment protocol is that Tor participants use S-coins to avoid exposing the forwarding path; outside senders, by contrast, use A-coins to maintain their anonymity.

6.3.2.1 Initial Set-up

All nodes participating in Tor acquire a public-private signature key pair (sk_U^s, pk_U^s) and a public-private encryption key pair (sk_U^e, pk_U^e), used to interact with the other members in the network. Bank generates a blind signature key pair (sk_B^b, pk_B^b) for signing A-coins. In addition to the hash H already used in Tor for integrity purposes, we establish another collision resistant hash function H_r for the coins' receipts. At the end of the circuit setup procedure in Tor, the sender shares with each node T_i in the path a secret key K_{ST_i} while any two consecutive nodes in a path share a secret key $K_{T_iT_{i+1}}$. In our system, the sender agrees with each path node on a hash function H_{ST_i}. The shared keys are used for communication encryption whereas the hash functions for integrity checks. We use M_k to denote message M encrypted under key K; $sig_U M$ is the signature of user U on M.

6.3.2.2 Payment Generation

A-coins are generated in cooperation with the Bank. When user U wants to obtain A-coins for payment, he generates a fixed set of random numbers $r_1, r_2, \ldots, r_n$, which serve as the receipts for the coins. Then, the user submits to the Bank the hashes $H_r(r_1), H_r(r_2), \ldots, H_r(r_n)$ which in turn signs them and generates coins of the form:

$$AC_i = sig_B(H_r(r_i)).$$

The resulting A-coins can be used for payment to any node in the network.

In the case of S-coins, users can generate them but they have to specify the payee. When user U wants to pay a node T_i with an S-coin, he generates the random number receipt r and its microcoin-like part MC which consists of a number that increases by one per S-coin payed by U to T_i and no timing information at all. The final form of the S-coin is:

$$SC_{U \to T_i}(r) = sig_U(MC, H_r(r), T_i).$$

6.3.2.3 Communication Protocol Description

Let S send to R a message M through the path $T_\ell, \ldots, T_1$. The following sequence of payments occurs for the transfer of the message:

- S pays T_ℓ ℓ coins, which may be A-coins or S-coins. Nodes outside Tor can only pay by A-coins while Tor nodes can use either type of coin.
- each node T_{i+1} on the forwarding path pays its successor T_i i S-coins.

The sender S chooses the receipts that will be used by the nodes on the path to generate payments for their successors. It also sends proofs to each of the nodes T_i in the form $H_r(r_1), \ldots, H_r(r_i)$ where $r_1, \ldots, r_i$ will be the receipts for the coins the node will get from its predecessor.

A node T_{i+1} gets the receipt for its payment coins from its successor T_i on the path.

Exchanged Messages The general form of the message that a node T_{i+1} sends to a node T_i on the forwarding path between sender and receiver is the following:

$$(\ \{T_i,\ \text{coins for } T_i,\ sig_{T_{i+1}}\{H(\text{coins for } T_i)\},\ \{M_{S \to T_i}\}_{K_{ST_i}}\}_{K_{T_{i+1}T_i}}\ )$$

- T_i specifies the receiver of the message
- "coins for T_i" is the payment the node gets for forwarding the packet. The coins here are either A-coins if the sender was an outside node and T_i is the first node in the path, or S-coins of the form $SC_{T_{i+1} \to T_i}$
- $sig_{T_{i+1}}\{H(\text{coins for } T_i)\}$ is mainly needed in the case of A-coins[5] and serves account-

[5] it can be eliminated in the case of S-coins

ability purposes when double-spending has been detected and

- $\{M_{S \to T_i}\}_{K_{ST_i}}$ is the part of the onion message from the sender that has to be read by T_i.

Now consider the last part of the message $M_{S \to T_i}$, which has the following form:

$$(\ T_{i-1}, \ \ T_{i+1} \text{ receipt}, \ \ \text{payment guarantee for } T_i,$$

$$\text{values for generation of coins for } T_{i-1}, \ \ \{M_{S \to T_{i-1}}\}_{K_{ST_{i-1}}} \ )$$

- T_{i-1} is the successor of T_i on the path
- the receipts for T_{i+1} are the random numbers that the sender generated encrypted with the key $K_{ST_{i+1}}$; T_i sends them back to its predecessor on the path
- the guarantees that T_i receives for its payment are of the form: $H_{ST_i}(r_1), \ldots, H_{ST_i}(r_j)$, where $r_1, \ldots, r_i$ will be the receipts for the coins he was paid with
- $\{M_{S \to T_{i-1}}\}_{K_{ST_{i-1}}}$ is the part of the onion message from the sender that has to be forwarded to T_{i-1}. In the case when T_i is the last node on the forwarding path, $M_{S \to T_{i-1}}$ is the message to the receiver.

After receiving its message from its predecessor, the node T_i acquires its payment, which is verified using the guarantees received from the sender. Then, it sends the receipts for T_{i+1} to its predecessor. Next, the node uses the values from the sender to generate payment coins for its successor T_{i-1}. It adds the coins to $\{M_{S \to T_{i-1}}\}_{K_{ST_{i-1}}}$, signs the whole resulting message and forwards it to its successor.

Deposit The deposit of all coins is handled by the Bank, which checks their validity and depositability. The validity of S-coins can be checked immediately by each node which is paid with them while the validity of A-coins is established at the Bank that checks for double-spending. At each deposit time the nodes deposit all coins that they have received during the period. Detailed analysis of the deposit period is provided in a later section.

Here, we define the procedure for deposit. Coins are considered for deposit if and only if they are accompanied by the corresponding receipt. The valid coins will be handled in two different ways: The deposit of S-coins is, in essence, a deposit of the underlying microcoins. This means that only a fraction of them will become depositable [Micali and Rivest, 2002]. All A-coins are depositable at their nominal value.

6.3.3 Comments

We preserve Tor's anonymity by allowing each node on the path to know only its predecessor and its successor. To this end, we harness the layered structure of the message passed by the sender to the forwarding path and the fact that payments are made between consecutive nodes. However, the sender still has control of the payments made along the path by sending the receipts used for their generation. A node that attempts to cheat can be easily identified by its successor. Since the successor holds the receipts for the cheater's payment there is no incentive for the cheater to either mangle or drop the message. Finally, Tor encryption guarantees both the confidentiality and integrity of all transmitted messages.

6.4 Security Analysis

There has been a wealth of research related to attacks against onion routing systems including Tor. Our goal is to ensure that PAR does not introduce new types of attacks, especially ones that can target either the anonymity or the robustness of an onion routing system. In addition, we prove the security properties of PAR using the augmented Tor threat model introduced earlier.

6.4.1 Sender-Receiver Unlinkability and Deposit Rate

We provide a formal model of information leakage of the payment scheme that can expose anonymity when combined with known attacks against anonymity networks. Although two differentiable types of payments are used in PAR this does not bring any higher risk than currently exists in Tor for the identity of the senders, which can be recognized as such if they use A-coins. The reason for this is that only nodes outside the system are required to pay

the first node in their forwarding path with A-coins and currently lists of the relay nodes in Tor are publicly available and therefore outside nodes using the anonymizing system can be also recognized by the first relay that they use.

We will consider attacks that have access to the deposit information in addition to corrupted nodes. In our payment scheme, the Bank can be considered a global adversary since it observes the deposits of coins made at all nodes. That is why in the analysis of possible attacks we will speak in terms of whether the Bank can disclose any of the anonymization that occurs in Tor's forwarding paths, with or without cooperation from malicious nodes.

The most serious type of attack for an anonymization network is one that manages to link senders and receivers communicating over the network. Since the senders using PAR pay with anonymous coins if they are outside nodes, the Bank cannot identify the start of the path that they choose to use. If the sender is a Tor node that forwards other traffic as well, the payments for all of its own and forwarded traffic are indistinguishable; hence the Bank cannot trace the traffic originating at the node just by observing deposits. The receivers are also unidentifiable by the Bank, since there is no monetary transaction between the last node and the receiver.

We have shown that the Bank by itself cannot link sender and receiver. Now we must consider the question whether an adversary observing the deposits can obtain partial information about a forwarding path by discovering three consecutive inside nodes in the path, i.e., being able to guess to where a node forwards packets received from a particular predecessor. Consecutive nodes in a path can be inferred from the signed coins deposits, but the only thing that this means is that there is at least one path that has that pair; nothing more is learned about which connection this path serves.

For the purposes our analysis let $cp^{T,\tilde{T},i}_{<T_\ell,\ldots,T_1>}$ be the packets transferred on a connection path such that $T = T_i$ and $\tilde{T} = T_{i-1}$. We denote the packets on all connection paths that have $\tilde{T}$ as a successor of T by

$$C(T,\tilde{T}) = \{cp^{T,\tilde{T},i}_{<T_{i_l},\ldots,T_{i_1}>} | 1 < i \leq \ell\}.$$

Then the number of coins that a node $\tilde{T}$ will receive from T will be

$$G(T,\tilde{T}) = \sum_{\forall cp^{T,\tilde{T},i}_{<T_\ell,\ldots,T_1>} \in C(T,\tilde{T})} i * c^{T,\tilde{T},i}_{<T_\ell,\ldots,T_1>}.$$

If we denote the number of anonymous coins that a node T deposits with $G_{ac}(T)$, we can calculate the number of packets forwarded by T (assuming that a node is paid with one coin for each packet forwarded):

$$\sum_{T'} G(T',T) + G_{ac}(T) - \sum_{T''} G(T,T'').$$

In order to hide the exact number of packets that it has forwarded, a node can deposit some of its own anonymous coins; thus the above expression will no longer be a correct estimate. Not knowing the rate of packet transfer nor the number of connections in which two nodes are consecutive, an adversary cannot receive enough information just from the deposits of coins to determine three consecutive nodes in a path.

Let us now assume that there is a malicious node that colludes with the Bank in order to reveal more about a path. The malicious node can disclose his predecessor and his successor on a particular connection path, as well as his position in that path. Let $T = T_i$ be such a malicious node in the path $T_\ell, \ldots, T_1$. Now the adversary can find out who are the nodes T_{i+1} and T_{i-1} and the number of packets k that T_i forwarded on that connection. The only thing that it can infer about the identities of T_{i+2} and T_{i-2} is that if

$$(i-1) * k > G(T_{i-1},\tilde{T}) \tag{6.1}$$

then the node $\tilde{T}$ cannot be a successor of T_{i-1} and similarly if

$$(i+1) * k > G(\tilde{T},T_{i+1}) \tag{6.2}$$

$\tilde{T}$ cannot be a predecessor of T_{i+1}. This is true only if we assume that the connections among different nodes have the same forwarding rate. Thus the chance of the adversary finding out anything more about the path than what it would have found out from a malicious node in Tor without any payments is very small.

In the discussion above we have made an implicit assumption that the deposits of coins occur at certain intervals during which enough connections have been established. The

statement "enough connections" means that there are no cases where only one node deposits another node's signed coins and it is clearly its successor in any connection. Also, we minimize the probability of Eq. 6.1 or Eq. 6.2 being true.

Deposit Rate. Now we give an estimate of what we consider "enough" connections and packets transferred during a deposit period. The situation in which an adversary may eliminate a link between two Tor nodes as being part of the path transferring the packets on a particular connection is when the payments made for that link are not enough for the packets that were expected to be sent on the connection. To avoid such situation, we want the expected payments made for packets forwarded along a link between any nodes during a deposit period to exceed the expected payment for the packets forwarded on a single connection.

Let us assume that there are N packets sent across a network consisting of n nodes over C connections during a deposit interval. Let L be the average length of the forwarding path. Then since the probability of a node being in any position on the path is $\frac{1}{n}$, the expected payment that a node will get per packet sent over PAR will be

$$\frac{1}{n}(1 + \ldots + L) = \frac{L * (L+1)}{2n}$$

Now considering that every node will forward on average $\frac{N}{n}$ packets, a node will be paid $\frac{N*L*(L+1)}{2n^2}$, which distributed across the $n-1$ edges going out of it yields $\frac{N*L*(L+1)}{2n^2*(n-1)}$ payment per edge. At the same time the average payment made for the packets on a connection is $\frac{N*L*(L+1)}{2C}$.

We observe that for

$$\frac{N * L * (L+1)}{2n^2 * (n-1)} > \frac{N * L * (L+1)}{2C}$$

to hold, we need $O(n^3)$ connections across the whole network or an average of $O(n^2)$ connections per node. We stress that with so many connections, an adversary would not be able to eliminate even a single possible path route for a given connection. If we now consider the situation when the adversary can narrow the possible successors of a particular node down to some number n_c, there are still n_c^ℓ possible paths for the connection. However in

this case we would want

$$\frac{N * L * (L+1)}{2n^2 * n_c} > \frac{N * L * (L+1)}{2C}$$

and we will need a total of $O(n^2)$ connections across the network or $O(n)$ per node.

In previous discussion we mentioned that each node may deposit some of its own anonymous coins to provide more anonymity of the traffic it is forwarding. We now point out that by having each node deposit anonymous coins we will additionally disguise the entry points for outside traffic being forwarded in the network. Since the ratio of anonymous and signed coins in the payment scheme is $\frac{2}{L-1}$, to preserve this ratio across all nodes each node should add its own anonymous coins to maintain the same deposit ratio.

6.4.2 Usable Efficiency

The efficiency of our payment scheme is comparable to that of micropayments [Micali and Rivest, 2002; Rivest, 2004]: the majority of the payment coins in our system are signed coins based on microcoins with the additions of receipts. These are much more efficient than ecash [Chaum *et al.*, 1988b], which requires zero knowledge proofs. (Even our anonymous coins are lightweight blind signatures.)

6.4.3 Accountability

The accountability property requires that the identity of a node that behaves maliciously – double-spending, forging attempts, message manipulation, etc. – will be revealed along with a proof of his guilt.

No node can tamper with the forwarded onion message since it is protected with layers of encryption that can be opened only in the corresponding order. Thus any attempt for forgery will be exposed by its successor. In addition, no double spending is possible for S-coin payments. Each of the coins is a signature by the spender; furthermore, it specifies the receiver and the payment details.

Double spending for anonymous coins is possible and can only be detected at deposit procedure. However, messages containing A-coins, contain also signed hashes of the coins,

which serve as proof of A-coins' origin if a double-spending has occurred. Thus, the nodes paid with the same coin have an proof for the misbehavior.

There is an issue of whether maintaining logs of coin related message exchanges is necessary after coins' deposit for satisfying accountability in our system. Indeed, keeping some A-coin/S-coin related logs is required to detect malicious actions by the spender/payee; In particular Bank is required to keep a log of the serial numbers of the A-coins that have been deposited so far and as well as the biggest serial number of S-coins each pair of peers has exchanged. The A-coins exchanges are required to be maintained for detecting the double-spender but only for the time of one deposit period.

Thus far, we have showed that our payment scheme abides by its design principles. We now prove that it still satisfies properties common for any viable payment scheme.

6.4.4 Correctness

When all participants act honestly and follow the protocol, our payment scheme fulfills its goals: all packets are delivered, the nodes on the forwarding path are paid, and the anonymity of the sender and receiver is maintained. If all nodes properly forward the onion message that is initiated by the sender it is guaranteed to reach its receiver because each forwarding node knows where exactly to send it. According to the payment scheme, each node receives exactly one coin more than it has to pay its successor per packet. Thus all nodes are paid equally for their service. We have already shown that payments observed by the Bank are not enough to compromise the anonymity of the identities of sender and receiver.

6.4.5 Robustness

Robustness refers to the probability that the path chosen by the sender will be secure in the presence of malicious parties in the network. Let us assume that the fraction of malicious nodes is α. Then the probability that there is no malicious node on a path of length l is $(1-\alpha)^l$. The computed probability, however, is important for the case when we assume that a malicious mode on the path prevents the traffic, i.e. it drops or misdirects it. This also holds in Tor with no payments. Now we restrict our attention to malicious nodes only

in the context of the payment system, i.e. nodes that may expose the connections going through them and the corresponding payments for them. Based on our analysis showing that a node acting in this malicious way can disclose its predecessor and successor in the forwarding path, at least half of the nodes on a path will have to be malicious in order to expose the identities of sender and receiver. Thus the probability of preserving the anonymity of sender and receiver over a path of length l is $(1-\alpha)^{l/2}$.

6.4.6 Monetary Unforgeability

No coin forgery is possible in the payment scheme since both types of coins are protected with signatures. Signed coins contain personal signatures of the payer; anonymous coins contain the Bank's signatures.

6.5 System Performance Evaluation

In this section, we quantify the computational overhead added to Tor by our payment scheme. We execute the `openssl speed` command 1000 times and compute the average estimated running time of blind and digital signatures (RSA), and symmetric key encryption and hashes (SHA1). We will focus on the overhead imposed on the communication initiator S as well as on a random path node T_i.

We define c_h to be the cost of a hash function, c_e the cost of a symmetric encryption procedure, and $c_s(c_{bs})$ and $c_{vs}(c_{bvs})$ the (blind) signature and (blind) signature verification cost. For 1024 byte messages hashed with SHA1, $c_h = 0.0045$ milliseconds. For CBC DES encryption[6] in blocks of 256 bytes and RSA signature and verification in blocks of 1024 bytes the estimated running times are $c_e = 0.020$, $c_s = 3.361$, and $c_{sv} = 0.142$ milliseconds. Assume a path of length L. For each payment round, S has to generate L receipts for the required A-mcoins and have them blindly signed by the Bank, and symmetrically encrypt the A-mcoins' receipts with $K_{ST_{L-1}}$. In addition, S should calculate the content of S-mcoins that each path node T_i will pay its successor T_{i-1}, and encrypt the receipts with $K_{ST_{i-1}}$

[6] We used DES for our tests, precisely because it is slower than AES; we wished to set a lower bound on performance.

key. Thus the overall computational cost for S for each payment round would be:

$$Cost_S = L * (c_{bs} + c_h + c_e) + \frac{L * (L-1)}{2} * (c_h + c_e)$$

For the usual case of $L = 4$, $Cost_S$ averages to 14.24 milliseconds overall, or to 1.4 milliseconds per coin to be paid.

On the other hand, each node T_i in the path, should create $i - 1$ for T_{i-1}'s S-mcoins and verify the validity of S-mcoins it received by T_{i+1} (signature verification and receipt):

$$Cost_{T_i} = i * (c_{vs} + c_h) + (i-1) * c_s$$

In this case T_i will have to spend 0.045 milliseconds for each coin it gets payed and 3.36 milliseconds for each coin it pays.

The performance impact of our scheme is dominated by two factors: the path length and the number of packets per payment. However, the two have very different properties. The number of packets per payment, N, represents the tradeoff between performance and risk. By setting N high, the total cost of our scheme is minimized, since the expense is amortized over a large number of transmissions. However, N also represents how willing nodes are to transmit packets without assurance of payment. If N is too high, a cheater can send a fair amount of data before being caught. Minimizing that risk requires setting N low, and hence increasing the cost.

6.6 Related Work

Previous research on applying payments in anonymizing networks was focused on mixnets: Franz, et al [Franz *et al.*, 1998], Figueiredo et al. [Figueiredo *et al.*, 2003] and Reiter et al. [Reiter *et al.*, 2005] all use a blind signature type of electronic cash to induce mixes to operate honestly. The approach of Franz et al. divides electronic payment and messages into small chunks and allows mixes and users to do the exchange step-by-step, which made the resulting system extremely inefficient. Furthermore, the receiver is required to participate in the payment procedure, which is undesirable: the receiver may not know or care about Tor. Figueiredo provided a completely anonymous payment system for mixnets, but without any accountability and robustness. Reiter et al. proposed a fair exchange protocol for

connection-based and message-based mixnets. However, their protocol assumes that mixes would work properly to receive their payment after they commit to their service. They do not provide any guarantee that participants will indeed get paid beyond the fact that the initiator will have no reason for not paying them. Furthermore, computationally expensive offline zero knowledge computations are required in the case of a message-based mixnet protocol [Clarke *et al.*, 2001], which renders the system inefficient and thus currently non-deployable.

6.7 Conclusions

In this chapter we dealt with anonymous browsing, which is a key property for user's privacy in online activities. In particular, we claim that current anonymity networks appear to lack wide participation due to their volunteer nature and propose to provide economic incentives, to incentivize users to both participate and to use anonymity networks to protect their communications. Unfortunately, current payment schemes cannot be used to enable payments in Tor. To address this, we introduce a novel hybrid scheme and prove that it is possible to add a secure payment scheme to an onion-based anonymity network. Our approach combines features of existing payment schemes in an innovative way, achieving provable sender-receiver unlinkability, accountability and efficiency at the same time.

Chapter 7

Privacy in Online Ads

Thanks to its ability to target audiences combined with its low cost, online advertising has become very popular throughout the past decade. Online shopping malls and commercial websites, news' websites, banking or healthcare websites are equipped with advertisements of various products. However, current profile-based advertising techniques raise privacy risks and may contravene users' expectations, while privacy-preserving techniques, e.g., anonymous browsing, create many opportunities for fraud. Security and privacy seem, thus, to contradict each other which becomes critical given the range of their applications.

In this chapter we show that the aforementioned concepts are not mutually exclusive. In particular, we analyze the privacy concerns raised by online advertising as well as the subsequent security issues, and propose a privacy preserving set of protocols that provide targeted ads with guaranteed fraud detection.

Privacy Concern: Targeted Ads. To increase their banner-ads' effectiveness, *publishers* — usually service oriented websites paid to show advertising spots of other companies' products — choose their ads based on users' browsing activity. More specifically, third party cookies enable special ad networks to track users' browsing activity across multiple websites, construct very accurate user-profiles [Krishnamurthy and Wills, 2006], and target ads accordingly. These advertising models track users even on sensitive sites, such as medical information websites, which could result in embarrassing advertisements appearing on other sites and in other contexts. A recent study [Turow *et al.*, 2009] show broad rejection

of the concept:

> *Contrary to what many marketers claim, most adult Americans (66%) do not want marketers to tailor advertisements to their interests. Moreover, when Americans are informed of three common ways that marketers gather data about people in order to tailor ads, even higher percentages –between 73% and 86%— say they would not want such advertising.*

The study found that over half of Americans felt that the punishment for illegal use of personal information should be jail time for the executives or that the company "be put out of business". The privacy issues become more serious when a conversion takes place, i.e., an online credit-card-based purchase or any activity which requires a login, thus linking a profile to a particular identity.

Security Concerns: Fraudulent Clicks. In the mechanism described before, publishers and ad-networks get paid by the advertisers in proportion to the number of clicks an advertisment receives from users. To dishonestly increase their revenue, publishers often fake clicks on ads. The existing privacy-preserving techniques, such as anonymizing networks, make detection of fraudulent clicks more difficult as all user identification elements are concealed.

Our Contribution. In this chapter, we present an online target advertising technique combining both privacy and security, PPOAd. More specifically,

1. we present a privacy-preserving mechanism for the current ad-system infrastructure guaranteeing similar or better revenues for all the entities involved
2. we present a privacy-preserving mechanism for click-fraud detection and show how this mechanism is applied in our system, and
3. we based our protocols on ecash and unlinkable credential systems

Organization. In the following section we present current ad-systems' architecture. In sections 7.2 and 7.3 we demonstrate our system's requirements, threat model and protocols, while in sections 7.4 and 9.1.5, we elaborate on our system's security, privacy and innovation w.r.t. the exising work.

7.1 Targeted-Ads System Architecture

Except for *users* — the online consumers — in a typical advertising mechanism, the principle parties are *advertisers*, *ad networks* and the *publishers*. *Advertisers* are the companies selling and promoting a particular product or group of products. *Publishers* are usually service-oriented websites paid to *publish* advertisements of advertisers' products. *Ad networks* are paid by advertisers to choose the list of advertisements which will appear on publishers and filter the clicks the ads receive. Typical examples of ad-networks are Doubleclick (owned by Google), Atlas Solutions (owned by Microsoft), Brightcove, and more. It is often the case that an ad network offers various services and also acts as a publisher.

When a user visits a website (publisher), the browser sends to the publisher some pieces of information called cookies, which link multiple visits of the same user. In fact, a special type of cookies, the *third party cookies*, are sent during the publishers' visit to the corresponding ad networks, who can now trace user activity across multiple websites. In this way, especially as ad networks collaborate with many publishers, they construct very accurate user profiles and target ads accordingly. There are many policies regarding how ad-networks and publishers are paid. The most popular one is the "cost per click" (CPC), where both parties are paid by the advertisers in proportion to the number of clicks the latters' ads receive.

As clearly shown before, targeted ads violate privacy, while CPC payment method motivates many attacks: publishers may fake clicks on ads they publish to increase their income, while advertisers may generate clicks on their competitors' advertisements to deplete the latter's daily advertising budget. Detection of click-fraud is currently the responsibility of ad networks. Unfortunately, it is apparent that any conventional mechanism concealing users' browsing activity may strengthen click fraud.

7.2 Requirements-Threat Model

In this section we will define *privacy*, *security* and *deployability* in the context of our system w.r.t. our system's requirements and threat model.

7.2.1 Requirements

Application layer *privacy* and *security* are the core requirements in our system. Privacy refers to the user-protection, while security refers to the protection of the other entities of the system. More specifically, we define privacy, as the union of:

- *User Activity Unlinkability.* No system entity should be able to profile a particular honest user, i.e., link two or more browsing activities as having originated by the same party, and
- *User Anonymity.* No system entity should be able to link a particular browsing activity to an identity.

In addition, we define security as the combination of the following properties:

- *Correctness.* We require that if all parties are honest, advertisers will pay publishers and ad networks in accordance to the number of clicks their ads have received, while privacy is maintained.
- *Fairness.* We require that parties in our system will be paid if and only if they do their duty properly.
- *Accountability.* Our system should also be accountable, i.e., misbehaving parties should be detected and identified.
- *Unframability.* We require that no user can frame an honest user for being responsible for a misbehavior, i.e., for click-fraud. It is conceivable that strong accountability implies unframability.
- *Mis-Authentication Resistance.* Unless authorized, no user should be able to make use of our system.

We can easily see how the click fraud detection requirement is covered through the fairness and accountability requirements: fairness requires that publishers should not receive payments for fake clicks on a particular advertisement, while accountability requires that the attacker is traced.

In addition, we require that our system provide *similar ad-efficiency*, which would result in similar profitability to the parties involved. At least as important, it must be deployable. Similar ad-efficiency and, thus, *similar profitability* for publishers and ad networks aims to eliminate any monetary constraints against the adoption of a new system. *Deployability* is important for the same reasons. We examine deployability from three aspects: (a) w.r.t. our system's architecture: not substantial changes in current ad-system architecture should be required for our protocols to be applied; (b) w.r.t. our threat model, as we will describe later on.

It is essential to note that both privacy and security provisions are required in the application layer. Also, we extend the current ad-system architecture with a single entity — which may or may not be distributed — the User Ad Proxy (UAP), which acts as a mediator between the user and each visiting website.

7.2.2 Threat Model

Ad-systems' strong monetary nature, imposes "following the money" the safest way to define our adversaries' motives and powers. In what follows, we examine our adversary w.r.t. users' privacy and ad-system's security.

Publishers may be "curious" w.r.t. users' privacy, i.e., they may collaborate with ad networks, advertisers or other users in order to reveal the identity of a particular user or to link browsing activities of the same user. In addition, we assume that *publishers are "honest and dishonest" w.r.t. the ad networks and advertisers.* In particular, we assume that they do provide correct user-profile related information to the ad networks, but may attempt to fake clicks to the advertisements they publish in order to increase their revenues.

Ad networks' revenues depend on the efficiency of the way they list ads in the various publishers, as well as on their credibility. Ads' efficiency depends on the accuracy of users-profiling, while credibility depends of the ad network's click frauds' detectability. It is, consequently, reasonable to assume that *ad networks are "honest but curious", w.r.t. users, while they are "honest" w.r.t. advertisers.*

Advertisers are considered to be "curious" w.r.t. the users. In particular, since advertisers have no direct interaction with them, we believe that they may collaborate with

publishers or ad-networks to make user-profiling more accurate.

UAP*is considered to be "honest but curious" w.r.t. the users.* More specifically, we assume that UAP is trusted to perform its functional operations honestly towards the users, but may collaborate with publishers or any other entity to link separate browsing activities of the same user. We also adopt a economic model so that UAP does not have a motive to cheat the advertisers.

7.3 A Privacy preserving Targeted-Ad System

As mentioned in the previous section, we extend the current ad-system architecture with the User Ad Proxy (UAP). UAP may be considered either as a single entity or as a group of collaborating entities and acts as a communication mediator between a user U visiting a publisher-website Pub and Pub. It is important to note that to hide any lower layer information emitted, U interacts with the rest of the system entities through an anonymizing network, while to automatically erase any cookies acquired and to be able to communicate with UAP or an UAP-member (if distributed), user-side installs a piece of software, which basically establishes an anonymous — communication layer — registration of user with the UAP.

The three core operations of our system: *(a)* the registration procedure of a user U at PPOAd, during which U obtains credentials to use the services of UAP, *(b)* the visit to a publisher, where a PPOAd-user requests a webpage, and *(c)* the ad-clicking procedure, where the user clicks on one of the publisher's ads (fig. 7.1). For convenience, we will assume that a user U is interacting with a publisher Pub. In addition, we will assume a single UAP, while in section 7.4, we will refer to the distributed UAP case.

Our scheme is based on the use of two types of tokens, issued by the user-UAP collaboration during the registration procedure: a registration credential regtick, which authorizes U as member of PPOAd multiple times anonymously and unlinkably, and a wallet with adticks, $\mathsf{W}_{\mathsf{adtick}}$, which will enable U to click on ads. regticks are blind towards the UAP, their possession can be demonstrated by their owner anonymously and unlinkably many times, each time resulting in a session-oriented ticket tick. Issued by the valid collaboration

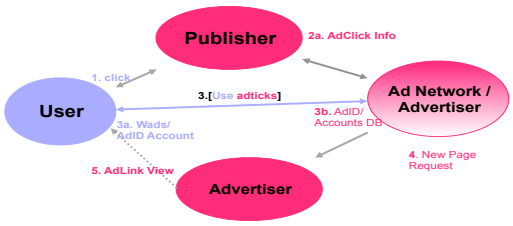

Figure 7.1: Clicking on Ads.

between U and the UAP, adticks are blind towards the UAP and can only be used for a limited number of times (MaxClicks) strictly by the person who issued them. For security purposes, U's identity is revealed in the following two cases: *(a)* when U attempts to make use of the same adtick more than once, or *(b)* if more than MaxClicks adticks of U are used for the same ad. We make use of regticks to achieve privacy w.r.t. UAP and adticks to achieve privacy and security w.r.t. to all the entities. Both tokens have an expiration date, so that users need to update their subscription on a monthly basis. In this way, we avoid unnecessary computations, as misbehaving parties can be detected and removed from the system.

When requesting for a webpage, U sends to UAP his ad-preferences, demonstrates knowledge of his regtick and proves that his regtick is not among the blacklisted ones. UAP contacts the website and provides it with the U-specified ad-preferences. Ads are then shown to U accordingly.

When U clicks on an ad (see fig.7.1), he uses one of the adticks he has obtained at the registration procedure. The adtick is then linked to the following combination:

$$\{\text{publisher} \,||\, \text{ad network} \,||\, \text{product-serial}\},$$

where product-serial is the product identification number within the ad network. It is apparent that the triplet mentioned before identifies the particular ad. If U clicks intentionally on the same ad more than a pre-defined number of times using his adtick s, he will risk his

privacy, as his identity will be revealed. However, U can choose instead to open an account for clicking on that particular ad, which will enable the ad network to decide whether series of U's clicks on that ad are legal or fraudulent. If classified as malicious, U's membership credential will be blacklisted.

As we will see later on, we use the blacklistable version of unlinkable credential system of [Tsang *et al.*, 2007] for the registration credentials regticks and the accountable ecash [Camenisch *et al.*, 2006] scheme for adticks.

In what follows, we describe our scheme in detail.

7.3.1 The PPOAd Protocol in detail

For the *setup* of our scheme, UAP runs EC.BKeyGen twice to establish the two accountable ecash schemes, which will be used for adticks (see section 4.1). In addition to its keys' generation, UAP runs the BAC.Setup procedure of the blacklistable anonymous credential (BAC) scheme for user-registration purposes, while it maintains two blacklists: the TempBL, where it stores the credentials in question, and, the PermBL, which is the official blacklist of the system.

Each ad network AdNet runs GS.Setup to generate the administration information for the group of publishers $\mathsf{G}^{\mathsf{AdNet}}$ it provides ads with: $\{\mathsf{gpk}^{\mathsf{AdNet}}, \mathsf{gsk}^{\mathsf{AdNet}}\}$. In response, each publisher collaborating with AdNet, runs GS.Join with AdNet to obtain membership in $\mathsf{G}^{\mathsf{AdNet}}$.

7.3.1.1 Registration (PPOAd.Register)

This is the case where a user U registers to UAP such that the former makes use of PPOAd's privacy services:

1. U provides the UAP with a piece of identification information. This can be a credit card, which will be used to pay U's subscription. U runs EC.UKeyGen to issue his system identity, the signature key pair (pk_{U}, sk_{U}).

2. U and UAP collaborate in a BAC.Register procedure, where U's credential $\mathsf{regtick_U}$ is issued. $\mathsf{regtick_U}$ is blind towards UAP.

3. U and UAP collaborate in a BAC.Authenticate procedure, so that UAP obtains a transcript of the $\mathsf{regtick_U}$ authentication phase, $\mathsf{mem_U}$. Note that the $\mathsf{mem_U}$ which was obtained by UAP in this way, serves blacklistability pursposes and cannot be linked to later authentications of U through $\mathsf{regtick_U}$.

4. U and UAP collaborate in two EC.Withdraw procedure, for the former to obtain two wallets $\mathsf{W_{ads}^{f,l}}$ of accountable ecash each corresponding to the two different settings of accountable ecash established in the setup phase.

5. UAP stores in its membership database the new user's entry: $\{\mathsf{U}, pk_{\mathsf{U}}, \mathsf{mem_U}\}$, and provides U with a signed proof of payment: $\mathsf{PaymRec} = \mathsf{Sig_{UAP}}(timestamp, \mathsf{U})$.

In what follows, we will assume that a user U visits a website Pub, which is in contract with a number of ad networks $\mathsf{Adv_1}, \ldots, \mathsf{Adv_m}$, who provide the website with ads.

7.3.1.2 Ad-targeting (PPOAd.Target)

This procedure involves the targeting of ads taking place when U visits Pub.

1. *User Authorization*: U interacts with UAP in a BAC.Authenticate procedure to authenticate himself as a non-blacklisted member of the PPOAd system. In this procedure U demonstrates knowledge — in a zero knowledge fashion — of his membership credential $\mathsf{regtick_U}$. Let SID be the corresponding transcript of U-UAP interactions.

2. tick *issue phase*: UAP issues a signed, dated permission tick, which will enable U to access the website requested. tick may have the form of

$$\mathsf{tick} = \mathsf{Sig_{UAP}}(timestamp, \mathsf{SID}).$$

3. *Preferences setup.* In this phase, U sets up his ad-preferences and sends them to the UAP. UAP then sends the webpage http request with U's preferences. As we will see later, the preferences-related info provided to UAP does not enable U activity-tracking neither by the UAP, nor by the requested website.

4. *Targeting.* Ad networks, who receive U's preferences as coming from UAP itself, process the ad-preferences and provide Pub with the corresponding list of ads.

5. Pub *Visit.* U provides tick to Pub and the Pub-webpage is presented to U.

7.3.1.3 Ad-clicking (PPOAd.Adclick)

This operation refers to the case, where U has already visited Pub and clicks on one of the ads an ad network $\mathsf{AdNet_i}$ provided to Pub. The series of interactions involve the following:

1. *Clicked Ad's website request.* Pub sends the ad-click information to the clicked ad's website, which is essentially one of the advertisers in contract with $\mathsf{AdNet_i}$. Let $\mathsf{Adv_j}$ be the one. The ad-click information includes $\mathsf{AdNet_i}$, Pub, SID and a timestamp. Note that this step is currently performed in ad-systems and serves billing and user-profiling purposes. Note that SID can be considered as a session identifier for the U.

2. AdID *construction.* In this phase the ad network, advertiser and clicked-product's identifier is popularized to U. The complete ad's identity would then be the following:

$$\mathsf{AdID} = \{\mathsf{Pub}||\mathsf{AdNet_i}||\text{product ID}\},$$

   where we assume that the same products of different advertisers have different identification numbers. As we will discuss in section 7.4, in addition to the AdID, an AdID-related key-pair is constructed $(pk_{\mathsf{AdID}}, sk_{\mathsf{AdID}})$.

3. adtick-*based Authorization.* Let MaxClicks be the number of times an honest user usually clicks on an ad.[1] Based on how many times U has — over all his browsing activity — clicked on that particular AdID, we have three adclick protocols of $\mathsf{U\text{-}AdNet_i}$ interaction:

   (a) If U has clicked on the same AdID fewer than MaxClicks times, he and $\mathsf{AdNet_i}$ collaborate in EC.Spend procedure, so that U spends one of his $\mathsf{W^i_{ads}}$ digital coins to the AdID related key-pair.

[1] This number varies from two to four, depending on how interesting that product is, and should be defined after suitable research.

(b) If U has clicked exactly MaxClicks times to the same AdID, he and $\mathsf{AdNet_i}$ commit in an EC.Spend procedure for one of the coins of U's $\mathsf{W^l_{ads}}$ wallet. In addition, U and $\mathsf{AdNet_i}$ collaborate in EC.UKeyGen for U to create an account $(pk_{\mathsf{U}}^{\mathsf{AdID}}, sk_{\mathsf{U}}^{\mathsf{AdID}})$ within $\mathsf{AdNet_i}$ for that particular AdID. $\mathsf{AdNet_i}$ stores

$$\{pk_{\mathsf{U}}^{\mathsf{AdID}}, \mathsf{SID}, \mathsf{AdID}\}$$

to its database.

(c) If U has clicked on the same AdID more than MaxClicks times, he has already been issued an AdID-account. Thus, he demonstrates knowledge of $sk_{\mathsf{U}}^{\mathsf{AdID}}$. In this way, his behavior towards this AdID will be traceable.

We can see that a user trying to attack an advertiser using PPOAd will eventually have his click-activity for that particular ad traced. In this way, $\mathsf{AdNet_i}$ may have all the information necessary to characterize the sequence of clicks on that AdID as malicious or benign. Different CPC rates may apply in this case.

4. If everything is fine, the the $\mathsf{Adv_j}$ website is presented to U.

If at any point of the procedure, U declines to cooperate, $\mathsf{AdNet_i}$ or Pub report SID to UAP, who can then run BACK.BLAdd on TempBL to put a temporary hold on $\mathsf{regtick_U}$ which corresponds to SID. Note that thanks to the properties of BAC system adopted, UAP does not need to know the user U or his $\mathsf{regtick_U}$. On the other hand, if U tries to click on the same AdID more MaxClicks times using his adticks, he will need to spend more than MaxClicks coins of his $\mathsf{W^i_{ads}}$ wallet or more than one coin of his $\mathsf{W^l_{ads}}$ to the same AdID. Because of the accountable ecash properties, this will result in revocation of pk_{U}, while $\mathsf{regtick_U}$ will be immediately blacklisted (through $\mathsf{mem_U}$).

7.3.1.4 Update Membership (PPOAd.UpdateMembership)

To enforce payment of its registered members' regular contribution, at the end of each prefixed period, UAP changes its credentials' parameters. To continue making use of PPOAd's services, users contact UAP by providing identification and payment (most recent PaymRec) information. Each user and UAP commit in a BAC. Authenticate protocol, for the former

to prove that his old credential is not among the blacklisted ones. If a user's $\mathsf{Uregtick_U}$ is not blacklisted, U pays his monthly contribution, issues a new $\mathsf{regtick_U}$ and receives new PaymRec. On the other hand, if $\mathsf{regtick_U}$ is blacklisted or U does not pay, his old credential will be invalid and thus will not be possible for him to use PPOAd's services.

7.4 System Considerations

In this section, we will elaborate on the *security* and *privacy* properties of our system, as well as on other practical issues.

7.4.1 Implementation Issues

System Software Components. We have already mentioned that for our system purposes, we extended the current ad system infrastructure with the UAP entity, while users of the system install a PPOAd-specific software, which involves the following:

- the User-ad-preferences software, where the users specify their public and topic-related ad-preferences which will occasionally be sent to UAP,
- the cookie-clean-up section, which removes all the cookies of users' browsing activity,
- the UAP/ad-network-communication part, which consists of the withdrawer side of the accountable ecash system, the user side of the anonymous credential system presented in chapter 4.

In addition, user-side makes use of an anonymizing network to contact UAP. UAP installs the bank and server side of the ecash and BAC systems respectively to authorize PPOAd-users to browse online. Ad networks and publishers just need to assign their products identity numbers and install the spender part of the ecash protocol.

User-ad-preferences play an important role for our system's *ad-efficiency.* As mentioned in previous section, after his registration to the PPOAd (PPOAd.Register) the user obtains and installs software to handle the PPOAd's interactions. Depending on how targeted wishes his ads to be, the user creates many partial profiles by choosing various types of products

he is interested in and the particular products in each category individually that may be of his interest. When the user visits a website and after the PPOAd.Authenticate phase — the user-software obtains the classification of the requested website and forwards to UAP the corresponding partial profile. Assuming that the lower layer information, i.e., consumer's machine's IP is hidden towards the UAP, because of the BAC system unlinkability property, the latter cannot link the partial profiles of the same user. In addition, being partial (related to the website), while prone to change at any time, the same partial profile may commonly be met across different users and will not be enough to link browsing activities of the same consumer across different websites.

AdID **key-pair construction**. To preserve security and privacy, it is essential that each AdID-combination: {Pub, AdNet, productID}, where Pub is the visited website, AdNet is the ad network providing the ad and productID the serial of the product, is assigned a different key-pair. To achieve this, AdNet constructs AdID's key-pair by contributing to the key-generation algorithm a pre-specified hash of the following quantity: $\mathsf{gpk}_{\mathsf{Pub}}||pk_{\mathsf{AdNet}}||\mathsf{productID}$, where $\mathsf{gpk}_{\mathsf{Pub}}$ is the Pub's public information in the $\mathsf{G}^{\mathsf{AdNet}}$, and pk_{AdNet} the public key of AdNet. In this way, the same key will be generated for the same AdID, without the need of precalculating it, while the probability that the same key is generated for two AdIDs is negligible.

Distributed UAP is critical if PPOAd is intended for large-scale use. This can be achieved through the use of blind group and group signatures, where UAP-related blind and plain signatures were used. In addition, depending on our privacy and computation efficiency requirements, we may group UAPs serving users of the same geographical area together. In this way, operations such as validity checks will be accelerated.

7.4.2 Privacy

Assuming that the partial profiles reveal nothing w.r.t. the consumer, and that user-cookies are successfully erased through the PPOAd-software installed, privacy in our system is guaranteed through the ecash and BAC systems' security properties (see section 4). In

particular, consumers' anonymity and activity unlinkability is provided directly via the anonymity and unlinkability properties of the blacklistable anonymous credentials used in PPOAd.Register and PPOAd.Target procedures and anonymity and unlinkability properties of the ecash schemes used in PPOAd.Adclick procedure. Note that even when a consumer clicks on the same AdID more than MaxClicks times, the former's behavior towards that particular AdID is only traceable (when and how often the consumer clicks on it) and not his overall browsing activity.

7.4.3 Security

Each part of PPOAd's security is satisfied. Correctness is guaranteed through the correctness of the schemes adopted. Mis-authentication resistance is achieved through the corresponding property of the blacklistable anonymous credential system used at the authorization phase of our protocols. Unframability is guaranteed through the combination of the mis-authentication resistance property and the ecash nature of the adticks: being unforgeable and ecash-based, only an authenticated PPOAd-user who issued the adticks can use them successfully. Fairness and accountability are achieved also through the accountable ecash security properties (see section 4): a user trying to click at the same ad many times will either have his public key revealed or his click-activity w.r.t. that ad traced. If the latter is the case, and the user is classified as malicious, he will be automatically be blacklisted and his ad-clicks ignored.

7.4.4 A Market Model

Incentivising users to use our system is critical for the latter's adoption since, if popular, both ad networks and ad agencies would be motivated to feed our system with advertisements. Users' motive is the combination of receiving ads related to their interests while maintaining their privacy.

On the other hand, it is necessary that all the participating entities behave properly for users to continue using the system. Threats towards security continue to exist. In our threat model, we assumed that UAP restricts users to a single registration to PPOAd. However, it is conceivable that in the real world, corrupted UAP entities — since they are paid by the

users — may be tempted to issue multiple accounts to the same party, which would enable the latter to forge clicks without limit. It is thus critical that we offer monetary incentive for the system entities to behave according to our threat model. In addition, since targeted advertising is already very profitable, we need to create incentives to direct it as we propose. In this section we elaborate on a market model towards for both cases.

In response to our threat model issue, we require that UAP entities are paid by both, the user — through his PPOAd-subscription — and by the ad networks who also benefit from click-fraud. Wanting to maintain their clientele, UAP will be forced to be "honest" in their functional operations towards both entities: users and ad networks.

As far as the PPOAd application is concerned, ad networks already have incentives to participate in our system: through the PPOAd click-fraud detection mechanism, ad networks' fraud detection ability — thus their credibility — will be enhanced. In addition, despite our privacy provisions, ad networks may still target ads even more effectively: the targeting procedure is now based on partial profiles provided by the user himself, while it is likely that their audience is extended with users who — strictly for privacy reasons — had so far removed ads from their browsers. Being offered better click-fraud detection rates, advertisers would also benefit from PPOAd.

7.5 Related Work

Fraudulent Click detection has been attempted in the past. In particular, Jakobsson, MacKenzie and Stern in [Jakobsson *et al.*, 1999] introduce an ad system where advertisers (in their system are called merchants) utilize e-coupons to detect malicious actions. However, in their scheme they do not deal with privacy the same way we defined it.

Combining targeted ads and privacy has been attempted in the past. Juels in [Juels, 2001] has suggested a target ad technique with the use of third parties, the negotiants, which would update a bulletin board with users' ad-preferences. Although perfectly secure in terms of privacy, they do not deal with our second security concern. Claessens and Diaz in [Claessens *et al.*, 2003] in fact suggested a more lightweight privacy preserving target ad system, where users would be grouped in terms of profiles for them to be presented

with ads. V. Toubiana et al. in [Toubiana *et al.*, 2009] have transfered the targeting mechanism to a browser extension, in a private way towards ad networks and publishers. However, though there are some suggestions, they do not consider click fraud. To privately target ads, iPrivacy [Stolfo and Smith, 2001] ecommerce system, had their clients obtain anonymous email accounts — held by the company itslelf or the banks — bound to specific advertising profiles; in this way users only receive ads of their interests.

7.6 Conclusion

In this chapter we addressed privacy and security in the area of online advertising, which is important for the privacy provided in various websites ranging from commercial and healthcare ones to banking. In particular, we provided a set of protocols providing targeted ads with similar ad-effectiveness as current systems in a privacy preserving way. As opposed to the existing work on the topic, the technique presented in this chapter takes measures against click fraud. In fact the privacy provided in our ad system is conditional, i.e., guaranteed only for honest users.

Chapter 8

Privacy in Online Transactions

Online transactions have considerably increased during the past few years. In an average online transaction we may identify four stages: *(a)* the research regarding the product to be bought, *(b)* the payment and *(c)* delivery of the online purchase as well as *(d)* the evaluation of the overall merchant user interaction. There are privacy and security issues deriving from each one of them; as we address the first two in chapters 7 and 9, dealing with the other two is the theme of this chapter.

More specifically, *product delivery* raises many privacy concerns, primarily deriving from information the delivery company acquires from the merchant. As noted, the delivery company is usually under contract to the seller. Given the (usually) long-term monetary relationship between the two, the delivery company knows the following:

1. the type of products the merchants sell
2. the name and shipping address of the person the product is for, who may or may not be the one who bought the product
3. the exact object shipped, if it is fragile or of great value.

Certainly, the courier company knows the person to whom the product is delivered, as well as the type of the product. In addition, since the same delivery company may serve a variety of other websites, the former may obtain a very good approximation of the transaction profile of consumers who often make purchases online.

Transaction rating systems raise privacy concerns as well: *privacy of a transaction is at least as strong as the privacy provided in the evaluation procedure following the payment.* On the other hand, the credibility of the overall rating system entirely depending on the credibility of the issued recommendations, complete anonymity is not desirable: an honest peer has no choice but to suffer from repeated misbehaviors (e.g. sending an infected file to others) of a malicious peer, which lead to no consequences in this perfectly anonymous world. Pseudonymity seems to offer a degree of compromise between the privacy need of the transaction parties and the credibility requirement of reputation systems. A peer, representing himself via a pseudonym, is free from the burden of revealing his real identity when carrying out transactions with others, while he may make his transactions unlinkable (i.e., hard to tell whether they come from the same peer) by using a different pseudonym in each transaction. Unfortunately, in most existing reputation systems [Kinateder and Pearson, 2003; Kinateder *et al.*, 2005] the reputation value is bound to each pseudonym, which enables malicious acts: a new pseudonym of a malicious peer will have a neutral reputation, irrespective of his past evil deeds, while honest peers may still suffer from his future misbehavior;; at the same time, as honest users won't use a new pseudonym in order to keep the reputation they have accumulated, they cannot fully enjoy anonymity and unlinkability.

To address both of these issues, in this chapter:

- we will introduce a *privacy-preserving delivery system* based on package-routing through multiple courier companies (section 8.1), where,
  - the courier company knows at most the merchant or the type of the product shipped, but not the recipient.
  - there is no way for the merchant to recover the address of the intended recipient without collaborating with more than one courier company.

  We emphasize on the fact that our system is deployable. Our threat model is based on the powers of any current real-world delivery system entities. For the purposes of our protocols, we made use of blind ([Chaum, 1981], [Camenisch and Lysyanskaya, 2002a], [Okamoto, 2006]) and group ([Camenisch and Stadler, 1997]) signatures as

well as of blind group signature schemes ([Lysyanskaya and Ramzan, 1998]).

- we introduce *identity bound reputation systems*, where reputation is bound to users' identities (section 8.2) as opposed to changeable pseudonyms. In particular, we define a privacy and security model for such a system and *construct a reputation system* which satisfies it. Our definition captures the following requirements:
  - Each peer has a reputation which he cannot lie about or shed. In particular, though each peer generates as many one time pseudonyms as he needs for his transactions, all of them must share the same reputation. Also, our system is robust against a peer's deliberate attempts to increase his own reputation.
  - Reputation values are updated and demonstrated in a way that does not compromise anonymity. In particular, the system maintains unlinkability between the identity of a peer and his pseudonyms and unlinkability among pseudonyms of the same peer.

8.1 Anonymous Delivery

In a typical online delivery system, we may identify the following entities:

- *Merchants*, who are the entities who maintain a website selling a particular product or series of products. A broader definition of merchants may include websites like Amazon or EBay, where a large variety of products is sold.
- *Customers*, who buy one more products from merchants.
- *Delivery Companies* (DCs), which are the courier companies paid by a merchant to deliver the product to an address specified by the customer. Delivery companies maintain a number of *mail stations* (MSs) on their own, while (if necessary) making use of the mail stations of other DCs. Although affiliated with DCs, in the following sections *mail station*s will constitute separate entities.

For anonymity purposes, we extend the current delivery system with a central *Anonymous Physical Object Delivery Administration* (APODA), which is the manager of our Anonymous

Physical Object Delivery (APOD) system. It authorizes the DCs and their mail stations to participate in the APOD, maintains the APOD website, etc. Merchants who need to send something anonymously may do it through any of the DCs which have contracted with APOD. As we will show in a later section, a part of the DC's payment goes to the APODA, who then distributes the payments among the rest of the nodes in the system according to the services they provided.

In the following sections we elaborate on the very specific privacy and security requirements and adversarial model of a delivery system (subsection 8.1.1.2), while we illustrate how the required properties are integrated in our protocols (subsections 8.1.2, 8.1.4). In subsection 8.1.5, we compare our scheme with other schemes from the online world.

8.1.1 Requirements - Threat Model

Our goal is to create a realizable system. In this section we see how "real world " is integrated in both, our adversarial model and requirements, while we show how similar it is to Tor anonymizing network.

8.1.1.1 Threat Model

Each *Merchant* is interested in maintaining his clientele, which implies that he is trusted to perform his functional operations correctly. However, we assume that he is "curious", namely he may try to combine information he possesses to reveal his customers' identities. A merchant may also collaborate with the DC he has paid to learn the recipient's address.

We make similar assumptions regarding *Delivery Companies*' powers . In particular, although "honest" in their functional operations, it is likely that a DC would collaborate with a merchant it has contracted with to reveal the recipient of a particular package[1]. The reason for the latter assumption is the following: the DC's primary concern is to maximize its profit and thus to get paid for the services it has provided. Because of this strong

[1]It is easy to see how this model is applied in real world if we consider the fact the employees in a DC may not trick any client directly, since they will lose their job, while they may try to combine information the company has obtained legally to draw their own conclusions.

monetary DC-dependence on the merchant, DCs are motivated — if requested — to provide the latter with all the recipient-related information its *mail stations* possess. Collusion between two DCs, however, is considered to be highly unlikely.

The *Anonymous Physical Object Delivery System(*APOD*)* consists of several independent or semi-dependent *mail stations* (MSs) which are associated with one of the DCs as well as affiliated with an administration authority (APODA). We generally assume that MSs are independent if they belong to a different DC, while there is a chance of sharing the information they possess when they are part of the same company. More specifically, each MS: *(a)* possesses its own secret authorization/identification information (digital and group membership signature keys), *(b)* forwards mail towards their destination by contacting at most the MS the package came from and the MS the mail is forwarded to, and, *(c)* may provide the information it possesses to the central authority of the same DC.

As mentioned before, for practical purposes we include in the design of the DC system an central administration station APODA, which handles payment and authorization matters. As such, it provides a valid MS with certificates (keys etc.). In our threat model, only the payment section of APODA is online and obtains no further information regarding the system unless compelled by a privileged authority such as a judge.

8.1.1.2 Requirements

Privacy is the main focus in our system. In the context of product delivery service (and assuming that no identity is revealed through the online payment procedure), privacy requires that the merchant should not be able to learn his customer's address, unless authorized by the latter. In addition, the DC should not be able to link any particular package destination address to the merchant who authorized the package's shipment.

Other requirements of our system, which basically derive from the nature of the system we want to enhance, are the following:

- *Package Delivery to Intended Recipients.* We require that the package shipped is delivered to the legal recipient of the package.

- *Package Tracing.* We require that a customer who has requested anonymous delivery

of her online purchases is able to trace her packages without any information related to her or the item shipped being leaked. In addition, we require that merchant is able to trace the status of the delivery of the product, without acquiring any information regarding the intended package recipient. Tracing the package from both merchant and customer is especially important when the package has not been delivered within the estimated time.

- *Fairness.* Delivery Companies and mail stations involved are only paid when they perform their service correctly.

- *Proof of Delivery/Accountability.* We require that there can exist an undeniable proof of receipt issued by the anonymous recipient when she receives the package. Although unforgeable, this "receipt" should carry no identification or location-related information. In addition, in case of delivery failure, there should be possible to trace the misbehaving party.

8.1.1.3 PAR vs. Tor

To better understand our delivery system's architecture, it is easy to see how the former assimilates PAR (see chapter 6), the payment system specially designed for the Tor anonymity network . In particular, APOD and PAR are similar in terms of threat models and goals.

- (Goals) In both cases the goal is accountable and fair packet/package delivery through a group of nodes/MSs with guaranteed sender/merchant - receiver/recipient unlinkability. Another similar goal is the user-anonymity w.r.t. the other communication party: PAR (Tor) requires *sender anonymity w.r.t. the receiver*, while in APOD we require *recipient anonymity w.r.t. the merchant.*

- (Adversarial Model) In both cases we deal with a local adversary, i.e. an adversary that may not control all the nodes/MSs in a user-chosen[2] delivery path. As in PAR (Tor), path nodes can only observe the traffic of their path neighbors and collaborate with other nodes which may or may not be part of the same path. Similarly, in our APOD

[2] *User* for PAR (Tor) is the sender, while for the APOD *user* is the recipient.

MSs may observe the package-flow from/to their path neighbors and collaborate only with mail stations of the same DC which may or may not be part of the path of a particular package. For APOD, we explicitly rule out "active attacks" such as attaching a GPS-based tracking device to the packages.

8.1.2 A Privacy Preserving Delivery System

In what follows, we will assume that each customer has completed the product purchase with the merchants anonymously, i.e., no identification information has leaked through product browsing or payment procedure.

As mentioned before, APOD is coordinated by an offline administration authority, the APODA. Delivery companies (DCs) which participate in the APOD obtain membership credentials from the APODA. In a similar way, APODA issues authorization credentials to the *mail stations* (MSs) that offer their services to the APOD. Therefore, the APODA is the coordinator of two groups: (a) the DC group (APODA-DC) and (b) the MS group (APODA-MS) of the participating DCs and MSs respectively. We need to emphasize that, although DC group members may own some or all of the MSs in the APODA-MS group, no package may be provided anonymous delivery unless authorized by a DC group member.

Each Merchant is in agreement with one or more DCs. In particular, each merchant is a member of the Mgroup (DC-M) of one or more DCs.

The customer chooses one among the DCs that have contracted with the merchant and are part of the APODA-DC group. Then, the merchant uses his DC membership credentials to issue a blind ticket T to the customer. The customer uses T to log in to APOD's website anonymously and to choose the MSs she wants her package to go through. She then collaborates with the APODA to issue one blind *package-coin (*pcoin*)* per MS in the path with serial numbers of her choice. Serial numbers in this case serve as package tracking numbers. The client uses the information contained in the website to encrypt triplets of

(package-coin, tracing-info, next-destination)

with each path station's public key. She then interacts with the merchant to get a proof-receipt of the final form of the label which the latter will attach to the product.

Within the delivery process, each path MS decrypts the part of the package-label corresponding to it, revealing the package coins (pcoins) as well as the MS to forward the package to. In addition, each MS uploads the tracing information to the APODA site, so that both the merchant and the client are informed of the package delivery status. We note that no piece of label-information provided to each path MS carries merchant/client identification information.

To assure that only the intended recipient of the product may receive the package, the customer and the merchant agree on a secret PIN number whose endorsed hash is added to the overall packet label. The endorsement is basically created by the DC in collaboration with the merchant in a way that it reveals no information regarding which exactly DC of the APODA-DC group has produced it.

To enforce that each station forwards the packet towards the right direction, package-coins (pcoins) are accompanied by receipts which MSs will only get from the next path station after the latter receives the package. As pcoins with their receipts will later be used for the distribution of payments among the path MSs, there is a strong motivation for MSs to do their job properly.

8.1.3 Protocol Description

The *Anonymous Delivery System's Administration* (APODA) makes the required setup (if any) for the two groups it manages (see section 4.6 for preliminaries):

- the APODA-DC group, which is instantiated through a blind group signature scheme and
- the APODA-MS group, which is realized through a plain group signature scheme.

Therefore, the APODA executes BGS.Setup and GS.Setup to obtain:

$$(\mathsf{bgpk}^{\mathsf{APODA-DC}}, \mathsf{bgsk}^{\mathsf{APODA-DC}}) \text{ and } (\mathsf{gpk}^{\mathsf{APODA-MS}}, \mathsf{gsk}^{\mathsf{APODA-MS}}).$$

In addition, for payment purposes, APODA executes BS.KeyGen to generate a blind signature key pair $(\mathsf{pk}_{\mathsf{APODA}}, \mathsf{sk}_{\mathsf{APODA}})$ and defines two hashes: a pcoin(H_{pcoin}) and a PIN (H_{PIN}) - related. The APODA publishes her public keys and the hashes:

$$\mathsf{bgpk}^{\mathsf{APODA-DC}}, \mathsf{gpk}^{\mathsf{APODA-MS}}, \mathsf{pk}_{\mathsf{APODA}}, H_{\mathsf{pcoin}} \text{ and } H_{\mathsf{PIN}}.$$

Delivery Companies (DCs) acquire membership in the group of companies participating in the APOD. More specifically, each delivery company DC_i collaborates with the APODA in a BGS.Join procedure to issue a blind group signature key-pair $(\mathsf{bgpk}^{\mathsf{APODA-DC}}_{\mathsf{DC}_i}, \mathsf{bgsk}^{\mathsf{APODA-DC}}_{\mathsf{DC}_i})$.

To manage all of its participating merchants, DC_i groups them together in a blind group signature group (see section 4.6), the $\mathsf{DC}_i - \mathsf{M}$. Therefore, DC_i performs the appropriate setup (BGS.Setup) to generate the corresponding blind group signature administration information:

$\mathsf{bgpk}^{\mathsf{DC}_i - \mathsf{M}}, \mathsf{bgsk}^{\mathsf{DC}_i - \mathsf{M}}$. DC_i publishes $\mathsf{bgpk}^{\mathsf{DC}_i - \mathsf{M}}$.

Mail stations (MSs) acquire membership in the APODA-MS group by interacting with the APODA in GS.Join protocol to issue $(\mathsf{gpk}^{\mathsf{APODA-MS}}_{\mathsf{MS}_i}, \mathsf{gsk}^{\mathsf{APODA-MS}}_{\mathsf{MS}_i})$, which enables each MS MS_i to sign a quantity on behalf of the APODA-MS group in an indistinguishable way. Each MS_i also runs EC.UKeyGen procedure to issue a public encryption key pair $(pk^e_{\mathsf{MS}_i}, sk^e_{\mathsf{MS}_i})$.

Each *Merchant* M_j is a member of the group of clients (M-group) of one or more DCs he has contracted with. Let DC_i be one of these DCs. To obtain membership, M_j collaborates with the DC_i's central authority in BGS.Join protocol to issue a blind group signature key-pair $(\mathsf{bgpk}^{\mathsf{DC}_i - \mathsf{M}}_{\mathsf{M}_j}, \mathsf{bgsk}^{\mathsf{DC}_i - \mathsf{M}}_{\mathsf{M}_j})$. M_j also runs EC.UKeyGen protocol to create a public encryption key pair $(pk^e_{\mathsf{M}}, sk^e_{\mathsf{M}})$.

Customer C has preestablished a pseudonymous account with the merchant, which we assume carries no C-identification information $(\mathsf{P_C}, \mathsf{secret}_{\mathsf{P_C}})$. Although out of the scope of this work, we may consider $\mathsf{P_C}$ as a pseudonym such as the ones introduced in anonymous credential systems [Lysyanskaya *et al.*, 1999].

In what follows we will assume that a customer C collaborates anonymously with a merchant M_j, while M_j has contracted with the Delivery Company DC_i.

8.1.3.1 Package Label Preparation Procedure

There are four main phases in preparing the label which will be attached to each package sent anonymously: merchant-client interaction, DC-client interaction, APOD-client interaction and merchant client interaction:

1. *Merchant-Client Interaction.* M_j and C agree on a number PIN , which will serve as an authentication code between the two. M_j hashes the PIN into

$$\mathsf{PIN}_h = H_{\mathsf{PIN}}(\mathsf{PIN}||date)$$

in order to use it later as part of the barcode on top of the product. Final MS will only hand out the package to a person who demonstrates knowledge of PIN . Finally, M_j interacts with C — through P_C — such that the latter obtains a blind credential from M_j, cred_b. cred_b is a blind signature of M_j on a random number N_r of C's choice

$$\mathsf{cred}_\mathsf{b} = \mathsf{BSig}^{\mathsf{DC}_i - \mathsf{M}}_{\mathsf{M}_j}(N_r),$$

where $\mathsf{DC}_i - \mathsf{M}$ denotes the M-group of DC_i. M_j does not know the final form of cred_b. However, anyone can confirm cred_b's validity as having derived by a valid DC_i's customer.

2. *Client-Delivery Company Interaction.* C uses cred_b to enter DC_i's website anonymously. DC_i's M-group administrator evaluates cred_b (BGS.Verify) and updates the statistics regarding merchant M_j. Here we need to note that according to the group signature attributes (see section 4.6) DC_i, as the M-group administrator is the only entity, who using BGS.Open procedure, can identify the merchant who produced a $\mathsf{DC}_i - \mathsf{M}$ group signature. C — through her cred_b — collaborates with DC_i to obtain a blind endorsement on PIN_h:

$$\sigma_{\mathsf{PIN}_h} = \mathsf{BSig}^{\mathsf{APODA}-\mathsf{DC}}_{\mathsf{DC}_i}(\mathsf{PIN}_h),$$

where $\mathsf{APODA} - \mathsf{DC}$ denotes the DC group of APODA. In addition, C establishes a one time use anonymous account with DC_i to enter APOD's website

$$A_\mathsf{C} = (\mathsf{BSig}^{\mathsf{APODA}-\mathsf{DC}}_{\mathsf{DC}_i}(N_A), N_A).$$

3. *Client-*APODA *Interaction.* Customer C logs in to APOD's website using A_{C}. The APODA verifies A_{C}'s validity (BGS.Verify), updates DC_i's statistics (BGS.Open) and allows C to browse in APOD's website to choose the route of her package. For each intermediate stop of the path she chooses, C:

   (a) collaborates with APOD to issue:

   $$(pc_1, r_1), (pc_2, r_2), \ldots, (pc_m, r_m),$$

   where $pc_k = \mathsf{BSig}_{\mathsf{APODA}}(H_{\mathsf{pcoin}}(r_k)), k = 1 \ldots m$ are the receipt enabled package-coins (pcoins). Receipt parts (r_k) are chosen by C and their hashes will serve as packet tracking numbers.

   (b) creates merchant-related package tracing parts: $mt_1, mt_2, \ldots, mt_m$, where

   $$mt_k = Enc_{\mathsf{M}_j}(K)||Enc_K\{1||\mathsf{Sig}_{\mathsf{P_C}}(N_k)\}, i = 1, \ldots, m.$$

   Namely mt_k are pseudonym-signed random numbers(N_k), encrypted under M_j's public key. "1" is used for merchant to realize whether an uploaded tracing number is referring to him.

   (c) combines the pcoins, their receipts and merchant package-tracing parts in groups of

   $$Msg_k = \{\mathsf{pcoin}(stop_k), receipt(stop_{k-1}), mt(stop_k), stop_{k+1}\}$$

   where

   $$receipt(stop_{k-1}) = Enc_{pk^e_{stop_{k-1}}}(K)||Enc_K(r_{k-1})$$

   is encrypted with (k-1)-stop's public key. The Msg for the last stop f, contains, additionally, $\mathsf{pcoin}(stop_f)$'s receipt in a PIN -encrypted form: $Enc_{\mathsf{PIN}}(receipt(stop_f))$. All Msg-s are encrypted with the public encryption keys each MS acquires from APOD's administration authority into $barcode_{stop_k} = Enc_{pk^e_{stop_k}}(Msg_k)$.

4. *Merchant-Client Interaction.* C, as $\mathsf{P_C}$, sends all barcodes and σ_{PIN_h} to the merchant M_j. M_j hashes and digitally signs (S.Sign) the entire barcode sequence into

   $$\sigma_{barcodes} = \mathsf{Sig}_{\mathsf{M}_j}(H_{\mathsf{proof}}(barcodes, \sigma_{\mathsf{PIN}_h}))$$

and sends it to C ($\mathsf{P_C}$) as a proof of what the former attaches to the packet to be sent out. C verifies the $\sigma_{barcodes}$'s validity and sends a verification response email with a notification of the first mail stop of the path: $\mathsf{Sig}_{\mathsf{P_C}}(stop_1, date)$.

8.1.3.2 Shipment

Merchant M_j prints out stickers for each of the barcodes as well as for the σ_{PIN}, which he attaches to the package to be sent anonymously. He then delivers the package to the first station of the path. For label integrity purposes, both parties, M_j and $stop_1$, exchange signed hashes of the encrypted route of the packet sent out:

$$\mathsf{Sig}_{\mathsf{M}_j}(H(barcodes, \sigma_{\mathsf{PIN}_h})) \text{ and } \mathsf{Sig}^{\mathsf{APODA-MS}}_{stop_1}(H(barcodes, \sigma_{\mathsf{PIN}_h})).$$

While the package moves from one MS to the other, each MS decrypts the barcode which corresponds to it. In this way, the next package destination is revealed along with the pcoin. Pcoins(pc_k-s) contained in each barcode are checked for validity (BS.Verify), while their serial is uploaded in the database of the APOD along with the merchant tracing parts (mt-s). In this way, C may track her package delivery status (by checking whether each serial number has been uploaded and thus reached its destination). At the same time, receipt parts of each barcode are sent back to the path predecessors of each station as a proof that the package was properly delivered. Merchant tracing parts (mt-s) are uploaded on APOD's website; M_j may then attempt to decrypt them using his secret decryption key. We note that M_j can only see the tracing numbers uploaded on the APOD website and not the particular MSs who uploaded them. To avoid any path recovery attacks based on the time each mt-s are uploaded, path MSs may randomize the time interval between the package arrival time and the corresponding mt-upload.

When the package reaches the final stop — where C picks her package up the last pcoin serial is uploaded. To obtain the package, C should provide the PIN agreed upon with the merchant. Non invertibility property of hash functions guarantees that only C is able to provide that number. A value different from H_{PIN} and a pre-agreed hash of the PIN ($H_{\mathsf{PIN\_received}}$) is then signed with MS's MS group signature uploaded to APOD's website:

$$\mathsf{Rec}_{\mathsf{Del}} = \mathsf{Sig}^{\mathsf{APODA-MS}}_{\mathsf{MS}_k}(H_{\mathsf{PIN\_received}}(\mathsf{PIN})).$$

M_j records $\mathsf{Rec}_{\mathsf{Del}}$ as proof that the package was properly delivered. At the same time, PIN reveals the receipt for the pcoin provided in the last stop. If no one comes to pick the package up within 10 days of its arrival at the last stop, the latter returns the packet to the MS it received it from.

8.1.3.3 Payment

The merchant charges the customer for the anonymous delivery service. The price may include the services of the upper bound of number of MSs that can be included in the anonymous path. DC_i charges the merchant in proportion to the merchant-signed endorsements the former receives from customers in the client-DC_i interaction phase. In a similar vein, the APODA charges the DC_i at each valid client-APODA interaction. The aggregated payments the APODA receives are distributed among the different MSs in proportion to the valid pcoins and receipts they present to the APODA.

8.1.4 System Considerations

In this section we will provide a brief presentation of how our requirements are satisfied.

8.1.4.1 Privacy

Privacy in our system consists of two parts: (a) *Recipient Anonymity* against the merchant and the delivery companies the latter has contracted with, and (b) *Sender-Recipient Unlinkability* against any delivery company or the APODA.

During the label preparation procedure, *Recipient Anonymity* is preserved through the combination of the anonymity provided by $\mathsf{P_C}$ and the unlinkability property guaranteed by the *Blindness* property of blind (group) signatures. In particular, a customer C uses her $\mathsf{P_C}$ pseudonym to browse the merchant's website, an (unlinkable to $\mathsf{P_C}$) anonymous account $\mathsf{cred_b}$ to browse to the DC's website and an (unlinkable to $\mathsf{cred_b}$) account A_C to visit APODA's website. The information each entity possesses at the stage of the label preparation is the following:

- the *merchant* M knows $\mathsf{P_C}$, the product $\mathsf{P_C}$ wants to have anonymously delivered, and that he provided $\mathsf{P_C}$ a blind $\mathsf{cred_b}$.

- the *delivery company* DC_i (as the manager of its M-group) knows that $\mathsf{cred_b}$ has interacted with M and that it provided $\mathsf{cred_b}$ with a blind A_C.

- the APODA knows that that A_C has interacted with DC_i and the MSs A_C has requested info for, which may finally be added to the delivery path or not. However, APODA has no information regarding M.

It is obvious that there is no recipient (customer) identification information known to any of the entities participating in the label preparation procedure. *Sender-Recipient Unlinkability* is also satisfied at this stage. Since timing is not an issue here, the merchant can not be linked to a particular A_C.

Customer Anonymity is preserved throughout the package delivery procedure. No C-identification information is contained in the label attached to the product. For the delivery of the product at the final stop, C only needs to demonstrate knowledge of PIN .

As far as the *Sender-Recipient Unlinkability* requirement is concerned, the information attached to the package, $(\sigma_{\mathsf{PIN}}||barcode_1||\ldots||barcode_m)$, has been created by the customer and cannot be linked to any of $\mathsf{cred_b}/A_\mathsf{C}$ accounts the latter used to create the label. However, each MS in the path knows both the exact form of the label attached to the package and its delivery path neighbors. In our threat model, MSs from the same DC may collaborate by comparing package labels, so they recover a package's path. Although we consider this case highly unlikely as it is not cost effective, the severity of this attack is considerably decreased by the following:

- M_j may attach the barcodes in any order. Although this would require extra computation power in each stop, as each MS will have to go through the entire label to detect the barcode which refers to it, no MS — except for the first and the last — will be able to find its place in the path.

- C is the one choosing the entire path. She can easily choose the first and final stops[3] to be from different DCs.

[3]We refer to the stops of these path positions, since they would link the sender (merchant) to a particular recipient (location wise).

Even in cases where the aforementioned scenario cannot be avoided, the most a DC may learn is the location of the final stop of a particular package without knowing the corresponding it to particular merchant or recipient. For completeness, we will refer to different types of collaborations between entities in our system. Although collaborations involving APODA or more than one DCs are not included in our threat model — since there is no direct monetary dependence between the merchant and APODA or other DCs — we refer to them as they may occur in the extreme case where a Judge has requested information about the recipient of a particular package.

1. M-DC_i (or DC_i-APODA): Because of $\mathsf{cred_b}$ (A_C) blindness, M-DC_i (DC_i-APODA) collaboration will reveal nothing more than what DC_i (APODA) knows.

2. any M-APODA collaboration: The APODA knows the MS – (mt-s/$\mathsf{Rec_{Del}}$) uploads correspondence, while M knows the (mt-s/$\mathsf{Rec_{Del}}$) – $\mathsf{P_C}$ correspondence. Thus M-APODA collusion may lead to complete package path recovery.

Depending on the privacy level we need to enforce, one way to avoid this attack scenario is via authorized-anonymous MS-logins(uploads) to APODA's website, using unlinkable-blind credentials [Syverson *et al.*, 1997]. Payments can be made through another type of blind coins, issued in response to each valid pcoin-receipt upload; these may be deposited unlinkably by MSs in person. Delivery proofs $\mathsf{Rec_{del}}$s may have the form of

$$\mathsf{Rec_{del}} = \mathsf{BSig_{APODA}}(H_{\mathsf{PIN}\_received}(\mathsf{PIN})),$$

where the signature is produced blindly by the MS-APODA collaboration and uploaded anonymously by the final path MS. In this way, M-APODA attempts at package path recovery will fail.

8.1.4.2 Package Delivery to Intended Recipients

It is satisfied through the non-invertibility attribute of hash functions. In this way, only the legal recipient of the package, i.e., the one who interacted with the merchant, is able to demonstrate knowledge of PIN . To avoid any attack on any party's behalf to link a package to a particular $\mathsf{Rec_{del}}$ upload, the final path MS uploads a pre-agreed hash of the PIN as opposed to the PIN itself.

8.1.4.3 Package Tracing

Package tracing is satisfied through the uploads of the pcoins' serials and the *mt*-s to the APOD's website. A merchant may visit that site anytime to collect the *mt*-s which refer to him. The customer may trace her package delivery status by checking on the serial numbers uploaded.

8.1.4.4 Fairness-Accountability

Fairness is satisfied in our system since, if a MS does not forward the package towards the right direction, it will never receive his pcoin receipt and will thus not be paid. pcoin receipts serve accountability as well, as they provide a proof of proper delivery of the package to the next path MS. Invalid pcoin-receipt pairs may be resolved through APODA, which will request the cooperation of all nodes to recover the full path corresponding to a package label and, thus, the misbehaving MS.

We note that we assume a customer does not deliberately provide invalid pcoin-receipt pairs, as it would only affect the payment distribution within the MSs, while she — having already paid the merchant — will have no monetary motive. On the other hand, the PIN requirement for the final package delivery guarantees that no customer can falsily claim failure of the delivery process.

8.1.5 Related Work

As mail service is not a new concept, anonymous package delivery has been addressed in the past by several companies.

iPrivacy [Smith, 2001] guarantees anonymous ecommerce activity, including anonymous delivery service. However, in iPrivacy the delivery company already knows the address of the recipient. The consumer provides the merchant with a special code number which corresponds to his address in iPrivacy's databases. iPrivacy then uses extra physical boxes, each with different address for the package to be sent to different locations prior to its final destination. Recipient anonymity in this case is physically vulnerable, while the iPrivacy company may link a merchant to a particular address.

ContinentalRelay [CO, 1999 2007] is another company guarranteing anonymous package delivery. However, in this case anonymity is guarranteed from the merchant (sender) but not from the delivery company itself: customers pay a monthly fee to maintain a fake Australian address. Every package sent to this imaginary mailbox is then forwarded to the customer's real address. However, this solution may be more expensive and inconvenient, as some mail carrier services will not deliver to a mailbox.

Kushik Chatterjee in [Chatterjee, 2008] has also suggested a patent for efficient anonymous package delivery service. In particular, Chatterjee suggested a system where the physical address of the recipient is identified within the delivery system with an identification number, which is what sender attaches to the mail sent. Thus recipient's physical address is concealed from the sender but not from the delivery company.

Tor[Dingledine *et al.*, 2004] and other onion routing protocols [Reed *et al.*, 1998] as well as PAR (see chapter 6) can also be considered as part of the related work and has been described in section 8.1.1.2.

8.2 Privacy-Preserving Evaluation Systems

As mentioned in earlier sections, in a reputation system we may classify participants in peers and the bank. A peer may interact with other peers and can be either a *user* (buyer) or a *merchant* in different transactions. Peers should be able to provide proof of their reputation record as well as evaluate the performance of each of their collaborators. The bank, on the other hand, maintains the peers' reputation logistics.

The operations supported by a reputation system are depicted in fig. 8.1. Peers who enter the system register with the bank to acquire an identity (Registration), which they can later use to generate reputation tokens. In the same picture two peers, a user U and a merchant M, who have already registered, through their pseudonyms $\mathsf{P_U}$ and $\mathsf{P_M}$ respectively, are considering to commit in a transaction. Before the transaction takes place, both parties exchange certificates of their reputation level (Reputation Demonstration). After the transaction terminates, both, $\mathsf{P_M}$ and $\mathsf{P_U}$ rate each other. The rating procedure (Reputation Award) can either be performed through direct exchange of special reputation

tokens between the two transaction-participants, or with the involvement of a semitrusted party through validated reputation reports.

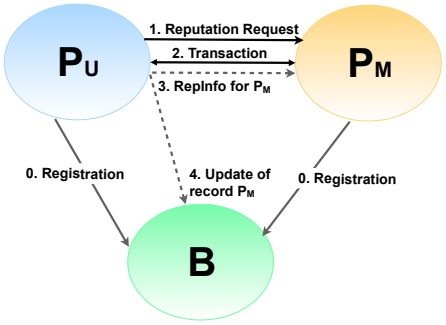

Figure 8.1: Series of operations in a transaction: 1. Registration, where both pseudonyms, the merchant's $\mathsf{P_M}$ and the user's $\mathsf{P_U}$, register to a central authority, the bank B. 2. Reputation request, where the two pseudonyms exchange information regarding the reputation each has so far acquired. 3. Actual transaction. 4. Transaction evaluation, where $\mathsf{P_M}$ and $\mathsf{P_U}$ evaluate each other. 5. Reputation record update, where both pseudonyms update their reputation records based on the evaluation they have received.

As mentioned before, a privacy preserving reputation system aims to combine the following properties under a real world attacker: *1. Anonymity:* no individual or collaboration of individuals should be able to link a pseudonym to a particular identity. *2. Transaction Unlinkability:* no individual or collaboration of individuals should be able to link two transactions having been performed by different pseudonyms as transactions of the same individual, i.e., no one should be able to connect two pseudonyms of the same user as such. *3. Fairness:* no one should be able to provide a valid proof of reputation assignment the former has not been assigned; (esp. important for the negative ratings case) no one should be able to avoid getting granted the reputation value, he has been assigned by his transaction-collaborators. *4. Reputation Record Unforgeability:* no one should be able to lie for the reputation value he owns. *5. Accountability:* misbehaving parties should be detected and punished (for more strict definitions, refer to section 8.2.1).

The required peer anonymity and transaction unlinkability posit fairness and accountability in a reputation system difficult to apply. Evidently, there are many problems and attacks which exploit the anonymity provisions of such a system to downgrade its credibility:

Pseudonymity WayOuts. In existing reputation systems, privacy is provided in the form of pseudonymity: to transact with others, each peer generates one or more pseudonyms unlinkable to each other and to his identity; apparently, parties with bad reputation may get rid of the latter without any consequence simply by deactivating the bad-reputed pseudonym.

Fairness Attacks. There are a number of attacks aiming fairness, i.e., Reputation awarding is not completed as the protocol states: the misbehaving parties may dishonestly award bad reputation with parties they have not yet interacted or avoid receiving bad reputation. More specifically:

1. Assuming that peers always know when they have misbehaved, misbehaving peers may always terminate intentionally the evaluation procedure at a point from which they can be favored i.e., before they are assigned (or providing the required information to be assigned) a bad reputation point or — even worse — after they have awarded a bad reputation point to the honest other transaction party.
2. In many existing protocols, peers having been awarded bad reputation by another peer can always avoid claiming it. This attack may work even if reputation is bound to the identity behind the pseudonym and is enabled by the way reputation update is enforced.

Cybill Attacks A user or a coalition of users may create pseudonyms, fake transactions and award a few pseudonyms which in this way unfairly acquire high reputation values.

In this chapter we deal with all of aforementioned attacks and problems. In particular, we formalize security and privacy in the context of identity-bound reputation systems, while we illustrate two sets of protocols satisfying these definitions: one supporting only positive among transaction parties (base protocol – see section 8.2.2.1) and its negative reputation

extension (complete protocol – see section 8.2.2.2). In our system we bind reputation to identities to achieve accountability, while we make use of both reputation assignment approaches: we utilize reputation tokens (repcoins) for our base protocol, while we involve the bank in the rating procedure to support negative ratings in the complete version of our system.

8.2.1 A Model for Anonymous Reputation Systems

In this section, we overview our system requirements, threat model and other underlying assumptions for our protocols. In addition, we enumerate the types operations considered in the system, followed by the security definition. We note that some of these definitions were inspired by previous work on other primitives, such as [Camenisch and Lysyanskaya, 2001; Camenisch *et al.*, 2005].

8.2.1.1 System Architecture

The entities involved in an anonymous reputation system are:

- *Peers*, who are the regular users of a P2P network. A peer interacts with other peers via pseudonyms of his choice and can be either a *user* (buyer) or a *merchant* in different transactions. Peers can award reputation points to other peers (through their pseudonyms), and can show their reputation level to other peers.

- *Bank*, who manages information regarding each peer's reputation (where the information is tied to actual identities — public keys — of peers, not to pseudonyms). Specifically, it maintains three databases: the reputation token (repcoins) quota database (denoted D_{quota}), the reputation database (denoted D_{rep}), and the history database (denoted D_{hist}). D_{quota} holds the amount of repcoins that each peer is allowed to award to other peers. When a peer withdraws a wallet of repcoins, the amount of his repcoin quota is decreased correspondingly. Bank also replenishes all the peer's account periodically, as per system parameters (for example, every day each peer can award at most 20 repcoins to others; see the discussion in Section **??**). D_{rep} contains the amount of reputation points that each peer has earned by receiving repcoins from

other peers. In order to prevent peers from double-awarding (awarding two peers with same repcoin), D_{hist} holds all the repcoins that are deposited.

8.2.1.2 Operations

The operations supported by a reputation system are depicted in fig. 8.1. To define them more strictly we will use the following notation: when an operation is an interactive procedure (or a protocol consisting of multiple procedures) between two entities A and B, we denote it by $\langle O_A, O_{\mathsf{B}} \rangle \leftarrow \mathsf{Pro}(I_C)[A(I_A), \mathsf{B}(I_{\mathsf{B}})]$, where Pro is the name of the procedure (or protocol). O_A (resp. O_{B}) is the private output of A (resp. B), I_C is the common input of both entities, and I_A (resp. I_{B}) is the private input of A (resp. B). We also note that depending on the setup, some operations may require additional global parameters (e.g., some common parameters for efficient zero-knowledge proofs, a modulus p, etc). Our system will need these additional parameters only when using underlying schemes that use such parameters, e.g., e-cash systems or anonymous credential systems. To simplify notation, we omit these potential global parameters from the inputs to all the operations.

1. $(pk_{\mathsf{B}}, sk_{\mathsf{B}}) \leftarrow \mathsf{Bkeygen}(1^k)$ is the key generation algorithm for Bank.

2. $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) \leftarrow \mathsf{Ukeygen}(1^k)$ is the key generation algorithm for peers. We call pk_{U} the (master) public key of U, and sk_{U} the master secret key of U.

3. $(\mathsf{P}, si_{\mathsf{P}}) \leftarrow \mathsf{Pnymgen}(1^k)$ is the pseudonym generation algorithm for peers. The si_{P} is the secret information used to generate the pseudonym P.

4. $\langle W, D'_{\mathsf{quota}} \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{RepCoinWithdraw}\ (pk_{\mathsf{B}}, pk_{\mathsf{U}}, n)\ [\mathsf{U}(sk_{\mathsf{U}}), \mathsf{B}(sk_{\mathsf{B}}, D_{\mathsf{quota}})]$. A peer U tries to withdraw n repcoins (in the form of a wallet W) from Bank B. Bank, using D_{quota}, checks if U is eligible for withdrawal. If so, the withdrawal is carried out and D_{quota} is changed accordingly.

5. $\langle (W', S, \pi), (S, \pi) \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{RepAward}\ (\mathsf{P}_{\mathsf{U}}, \mathsf{P}_{\mathsf{M}}, pk_{\mathsf{B}})\ [\mathsf{U}(si_{\mathsf{P}_{\mathsf{U}}}, W, pk_{\mathsf{U}}, sk_{\mathsf{U}}), \mathsf{M}(si_{\mathsf{P}_{\mathsf{M}}})]$. A peer U (via P_{U}), using his wallet W, gives a repcoin (S, π) to M(via P_{M}). Here S is a serial number and π is the proof of a valid repcoin.

6. $\langle \top, (D'_{\mathsf{rep}}, D'_{\mathsf{hist}}) \rangle / \langle \bot, \bot \rangle \leftarrow$ RepCoinDeposit $(pk_{\mathsf{B}}, S, \pi)$ [M($\mathsf{P_U}$, $si_{\mathsf{P_U}}$, pk_{U}, sk_{U}), B(sk_{B}, D_{rep}, D_{hist})]. A peer M deposits the repcoin into his reputation account. If the repcoin (S, π) is valid and not double-awarded, then the coin is stored in the history database D_{hist}, and the amount of reputation of pk_{M} in D_{rep} is increased by one.

7. $(pk_{\mathsf{U}}, \Pi_G) / \bot \leftarrow \mathsf{Identify}(S, \pi_1, \pi_2)$. If a repcoin is double-awarded with (S, π_1) and (S, π_2), Bank can find the peer who double-awarded the coin using this operation. Here, Π_G is a proof that pk_{U} double-awarded the repcoin with the serial number S.

8. $\top / \bot \leftarrow \mathsf{VerifyGuilt}(S, \Pi_G, pk_{\mathsf{U}})$ outputs $\top$ if the peer U (represented by pk_{U}) indeed double-awarded the coin with the serial number S.

9. $\langle C_U^l, \top \rangle / \langle \bot, \bot \rangle \leftarrow$ RepCredRequest $(pk_{\mathsf{B}}, pk_{\mathsf{U}}, l)$ $[\mathsf{U}(sk_{\mathsf{U}}), \mathsf{B}(sk_{\mathsf{B}}, D_{\mathsf{rep}})]$. A peer U requests a credential that will enable U to prove to another peer that he has reputation level l. Bank B refers to D_{rep}, and if U has sufficient reputation it issues a credential C_U^l. (As it will be discussed later, how exactly the reputation levels are defined is a system parameter).

10. $\langle \top, \top \rangle / \langle \bot, \bot \rangle \leftarrow$ ShowReputation $(P_{\mathsf{U}_1}, P_{\mathsf{U}_2}, pk_{\mathsf{B}}, l)$ $[\mathsf{U}_1(sk_{\mathsf{U}_1}, si_{P_{\mathsf{U}_1}}, C^l_{\mathsf{U}_1}), \mathsf{U}_2(si_{P_{\mathsf{U}_2}})]$. A peer U_1 (via P_{U_1}) proves to U_2 (via P_{U_2}) that he has reputation level l.

8.2.1.3 Threat Model.

Aiming to integrate "real world" in our protocols we make the following assumptions regarding our adversary:

- *Peers may attempt to "cheat" in any way.* Aiming to increase their credibility and profit, peers may attempt to forge their reputation record or reputation tokens (repcoins) so that they uncontrollably increase their reputation; they may also pursuit to frame other users to appear guilty for their actions; as we will see later, peers may also endeavor to reveal a pseudonym's identity.

- *Bank is assumed "honest but curious".* Namely, as its monetary interest is to increase its profit, it trusted to behave honestly in its functional transactions, which involve maintenance of reputation levels and issuing the repcoins for each peer. However, it

may be motivated to combine the information it has legally obtained (D_{quota}, D_{rep}, and D_{hist}) to compromise users' privacy, i.e., link a particular pseudonym to its owner's identity.

It is noticeable that in our threat model we assume *Unbounded Collusion.* Any number of parties on this network may collude to break anonymity of some other party. We specifically include the bank in this assumption. We assume collusion because in most real environments, it is possible for one party to open multiple accounts on the system. It may cost more money, but it does achieve the goal. Since a bank employee can do the same, we assume that the bank is colluding, too, albeit perhaps in response to a court order. Even if we assume a foolproof system for restricting accounts to one per person, two or more people could communicate via a private back channel, thus effectively creating multiple accounts under control of a single entity.

We need to emphasize on the fact that in our threat model we do not consider any type of information leaked through the lower communication layer. In particular, as in all parts of user online interaction, we assume that all communication takes place over an anonymous communication network, e.g., a Mixnet [Chaum, 1981] or an Onion Router [Reed *et al.*, 1998; Dingledine *et al.*, 2004].

8.2.1.4 Requirements.

Using the definition of the operations we presented before, in this section we present how the general definitions of security and privacy — i.e., their coefficients — are interpreted in the context of a reputation system.

Correctness requires the following:

1. If an honest peer U_1, who has enough repcoins in his repcoin quota, runs RepCoinWithdraw with an honest Bank B, then neither will output an error message; if the peer U_1, using the wallet (output of RepCoinWithdraw), runs Award with an honest peer U_2 (via his pseudonym), then U_2 accepts a repcoin (S, π); if the peer U_2 runs RepCoinDeposit with the honest Bank to deposit the repcoin (S, π) then U_2's reputation in Bank will be increased by one.

2. If an honest peer U_1 runs RepCredRequest with an honest Bank and a reputation level for which he is eligible, then U_1 gets a valid credential. For a valid credential C_U^l, its owner can always prove his reputation through ShowReputation$(l, C_U^l, \ldots)$ procedure.

For the following definitions, we say that a peer U appears consistent with a pseudonym P to an adversary A who has corrupted certain parties including the bank, if U and P's owner are uncorrupted, and if the levels for which P successfully invoked ShowReputation are a subset of the levels for which U successfully invoked RepCredRequest. We also assume that A is participating in the system for some arbitrary sequence of operations executed by honest and corrupted parties.

Privacy is defined as the combination of the following two unlinkability properties:

1. *Peer Anonymity (Peer-Pseudonym Unlinkability).* Given a pseudonym P that does not belong to a corrupted party, the adversary can learn which peer owns P no better than guessing at random among all non-corrupted peers that appear consistent with P.

2. *Pseudonym-Pseudonym Unlinkability.* Assuming that the bank is not included in the corruptable parties, this property requires that given two pseudonyms $\mathsf{P}_1, \mathsf{P}_2$ that do not belong to corrupted parties, the adversary has no advantage in telling whether $\mathsf{P}_1, \mathsf{P}_2$ belong to the same peer or not. Next, consider an adversary who corrupted some peers and Bank as well. Then the above requirement should hold as long as there are *at least two* non-corrupted peers who appear consistent with both P_1 and P_2 (because if there is only one such uncorrupted peer, clearly both pseudonyms belong to the same one).

As denoted in previous sections, *security* in our scheme consists of fairness, accountability, exculcability and reputation record unforgeability:

1. *Fairness (No Over-Awarding).*

   (a) No collection of peers should be able to award more repcoins than they withdrew. Suppose that n peers $\mathsf{U}_1, \ldots, \mathsf{U}_n$ collude together, and that the sum of the amount

of repcoins allowed to them is N. Then, the number of different serial numbers of repcoins that can be awarded to other peers is at most N.

(b) Suppose that one or more colluding peers run the Award protocol with two pseudonyms $\mathsf{P}_{\mathsf{M}_1}$ and $\mathsf{P}_{\mathsf{M}_2}$ such that $\mathsf{P}_{\mathsf{M}_1}$ gets (S, π_1) and $\mathsf{P}_{\mathsf{M}_2}$ gets (S, π_2). Then, we require that $\mathsf{Identify}(S, \pi_1, \pi_2)$ outputs a public key pk_{U} and a proof of guilt Π_G such that $\mathsf{VerifyGuilt}(pk_{\mathsf{U}}, S, \Pi_G)$ accepts.

(c) Each repcoin that is accepted but not double-awarded in the Award protocol increases exactly one reputation point in the database D_{rep} irrespective of the beneficiary of the repcoin. However, we don't regard it as a breach of security when a peer M_1 received a repcoin but passed it to M_2, who deposited it into his reputation account; in any event, this is just another form of collusion. Another justification is that the peer M_1 sacrifices one reputation point.

2. *Exculpability.* This property is to protect the honest peer from any kind of framing attack against him. No coalition of peers, even with the bank, can forge a proof Π_G that $\mathsf{VerifyGuilt}(pk_{\mathsf{U}}, S, \Pi_G)$ accepts where pk_{U} is an honest peer U's public key who did not double-award a repcoin with the serial number S.

3. *Reputation Unforgeability.*

(a) No coalition of peers, where l is the highest reputation level of any one of them, can show a reputation level higher than l for any of their pseudonyms. This implies as a special case that a single peer cannot forge his reputation.

(b) Consider a peer U with reputation level l, who owns a pseudonym P. Suppose that some coalition of peers has empowered U with the ability to prove that P has reputation level $l' > l$. Let Bad be the set of peers with reputation level at least l' among the coalition (note that by the previous requirement, there must be at least one peer in Bad). Then, it must be that U can learn the master secret key of a peer $\mathsf{U}' \in \mathsf{Bad}$.

8.2.2 Anonymous Identity-Bound Reputation System

We will now highlight the main concepts of the operation of both schemes:

- the *base scheme,* which refers to the offline reputation system supporting exclusively positive ratings, and
- the *complete scheme,* which constitutes an extension of the "base scheme" to support fair negative ratings.

In both cases: To restrict peers' registrations to the bank to at most one per peer, to enter the system the latter provides the bank with strong identification information. The peer obtains system membership information and issues digital cash-based, reputation-credentials, the repcoins. In response, the bank creates an entry in its *reputation database* D_{rep}, where it maintains the data regarding each peer's reputation. To handle the repcoin data, the bank also maintains the *repcoin quota database* D_{quota}.

Base Scheme. After the transaction between two peers U and M takes place, the *reputation award* operation takes place. As shown on fig. 8.2, a peer U(via his pseudonym $\mathsf{P_U}$) can increase the reputation of a pseudonym $\mathsf{P_M}$ by giving a *repcoin*,[4] which is basically a digital cash coin.

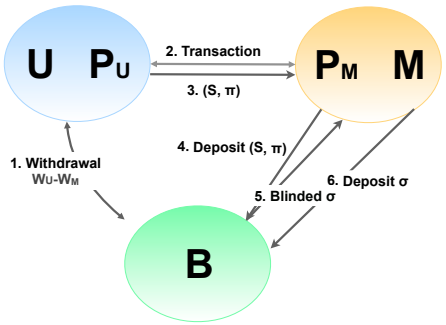

Figure 8.2: Reputation granting process: (1) U withdraws a wallet W (i.e., repcoins) from the Bank B. (2) U, via $\mathsf{P_U}$, awards (i.e., spends) a repcoin (S, π) to M. (3) M, via $\mathsf{P_M}$, deposits the repcoin (S, π). (4) If the deposit is successful, $\mathsf{P_M}$ obtains from B a blind permission σ. Note that σ is blind to B and only visible to M. (5) M deposits σ, and B increases M's reputation point.

[4]If M wants to increase of reputation of $\mathsf{P_U}$, they can carry out the same protocol with their roles reversed.

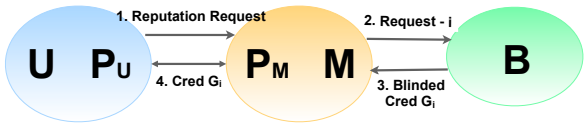

Figure 8.3: Reputation demonstration process: (1) M requests a credential for the group G_i. (2) If M has enough reputation count for G_i, B issues a credential cred to M. (3) By using cred, $\mathsf{P_M}$ proves its membership of G_i to $\mathsf{P_U}$.

At their convenience, peers deposit the repcoins they have received from their collaborators, as shown in fig. 8.3. Note that M does not deposit the repcoin using his identity. This is for the sake of maintaining unlinkability between a pseudonym and a peer: *if* M *directly deposited the repcoin, collusion of Bank and* U *would reveal that* M *and* $\mathsf{P_M}$ *are linked.* Thus, to preserve unlinkability, we use a level of indirection. When $\mathsf{P_M}$ successfully deposits the repcoin, it gets a blind permission from the bank, which — realized as a blind signature — does not contain any information about $\mathsf{P_M}$. So, M can safely deposit the permission.

The reputation demonstration procedure is illustrated in fig. 8.3. It is noteworthy that $\mathsf{P_M}$ does not reveal its exact reputation value, but shows its membership to the reputation group G_i. In particular, demonstration of exact reputation value could allow an attacker who continuously queries for the reputation of many pseudonyms — without even needing to transact with them — to infer whether two pseudonyms correspond to the same user. To make matters worse, with bank's collaboration, pseudonyms can be linked to a limited number of identities that have the exact same reputation value with the pseudonym. Grouping together identities which belong to the same reputation level, makes small changes in reputation accounts invisible to other pseudonyms. Bank can still see the changes that take

place in peers' reputations, but cannot link them to specific pseudonyms any more.

We chose to employ an anonymous credential system (see section 4.3). to realize the reputation groups. The anonymous credential enables M, via his pseudonym $\mathsf{P_M}$, to prove his membership in group G_i anonymously. Thus, unlinkability between M and $\mathsf{P_M}$ is maintained. The reputation levels (i.e., groups G_i) are defined as a system parameter. Reputation levels are not necessarily required to be disjoint. One example would be that G_i contains peers who has more than 2^i different reputation values.

Negative Reputation Extension Scheme. To support negative ratings, the reputation database contains two types of entries for each identity, one for the later's positive reputation value and one for his negative reputation. In fact, there are two types of repcoins, the merchant ones, that can be used only when the peer acts as merchant in a transaction and the user ones, when the peer participates as buyer.

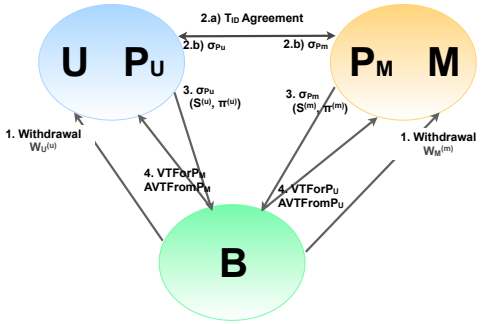

Figure 8.4: Transaction Preparation Procedure: (1) U(M) withdraws $\mathsf{W_U^{(u)}}$($\mathsf{W_M^{(m)}}$) wallet (i.e., repcoins) from the Bank B. (2) $\mathsf{P_U}$ and $\mathsf{P_M}$ agree on $\mathsf{T_{id}}$ and produce $\sigma_{\mathsf{P_U}}$ and $\sigma_{\mathsf{P_M}}$ respectively. (3) $\mathsf{P_U}$ and $\mathsf{P_M}$ deposit $\sigma_{\mathsf{P_U}}$ and $\sigma_{\mathsf{P_M}}$ resp. Both commit to $\mathsf{T_{id}}$ by spending to B a user repcoin $(S^{(\mathsf{u})}, \pi^{(\mathsf{u})})$ ($\mathsf{P_U}$) and a merchant one $(S^{(\mathsf{u})}, \pi^{(\mathsf{u})})$ ($\mathsf{P_M}$). (4) B issues nominal VoteTicks (VTFor) and AcceptVoteTicks (AVTFrom) for the two pseudonyms to rate each other.

The reputation award procedure is depicted in figures 8.4 and 8.5. As before, we assume that a merchant M and a user U participate in a transaction. As opposed to the base scheme,

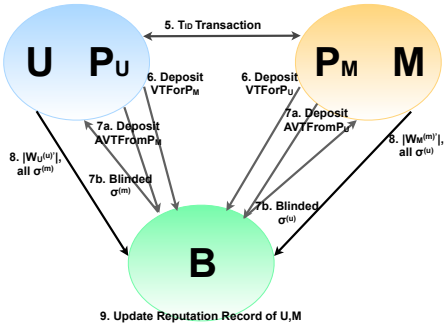

Figure 8.5: Reputation Update process: (5) The actual Transaction takes place. (6) $\mathsf{P_U}$ and $\mathsf{P_M}$ deposit their VoteTicks with the reputation value they want to award their transaction partners, while they use their AcceptVoteTick to accept the reputation value they have been awarded. (5) If the deposit is successful, i.e., the tickets are fresh and valid, $\mathsf{P_M}$ and $\mathsf{P_U}$ obtain from B the corresponding type of blind permission σ. Note that σ is blind to B and only visible to the pseudonym which receives it. (6) U and M deposits σs and the unspent part of their user or merchant wallets. B checks whether the deposited elements match its logs and updates U's and M's entries in D_{rep} accordingly.

the bank acts as a semitrusted mediator, who before the transaction starts, guarantees that both, $\mathsf{P_M}$ and $\mathsf{P_U}$ have committed in the transaction by using one merchant-repcoin and user-repcoin respectively. After both parties have committed in the transaction, the bank issues pseudonym-specific reputation tickets to both pseudonyms which can be used by them after the transaction takes place to do two tasks: to rate each other and to receive the reputation they have been granted by their collaborator. The repcoin deposit procedure is similar to the one of the base scheme. Pseudonyms deposit one part of their reputation tickets to receive their rating that they have been awarded throughout their transactions and — depending on the type of rating (positive or negative merchant or user) — the bank issues blind reputation permissions (as the ones issued in the base scheme), which are finally deposited by the peers as identities. To ensure fairness, for each repcoin type, the amount of the repcoins withdrawn by each peer should be equivalent to the sum of the unused repcoins and the deposited permissions.

The reputation demonstration procedure for both types of reputation takes place the same way as in the base scheme, while the same primitives are used for and permissions.

In both schemes we make some assumptions, which are critical for the accountability and fairness provided in our system.

- *Accountability Incentive Mechanism.* As opposed to the rest of the components of system security, whose goals we aim to achieve by rendering a break of the security computationally infeasible (modulo some standard cryptographic assumptions), in terms of accountability there are some breaches which are impossible to completely prevent. For example, as long as there is no central party involved on-line in each transaction, a user can always award the same repoint twice to different parties. As another example, if anonymity and unlinkability is to be preserved, a peer with a high reputation level can always give away all his data and secret keys to another peer, allowing the latter to claim and prove the high reputation as his own. To achieve accountability, we build into our model an incentive structure (similar to previous work, e.g., [Lysyanskaya *et al.*, 1999]), whereby such security breaches would hurt the offender. In particular, for the first case above, we require that a double

awarding of a repcoin would reveal the identity of the offender (which can then lead to consequences outside of our model). For the second case, we require that in order for Alice to empower Bob, who has a lower reputation level, to prove a reputation level as high as Alice's, Alice would have to effectively give Bob her master private key. This information may be quite sensitive, especially if the private key used within the reputation system is the same one used for a public-key infrastructure outside the system.

- *Restricted Awarding.* There is some limit to the number of reputation points any party can hand out per unit time. While we don't specify how this limit is set, we tentatively assume that it costs real money to obtain such points to hand out. This might, for example, be the daily membership fee for participation in the P2P network. Note that the assumption corresponds quite well to the best-known existing reputation system, Ebay. One can only dispense reputation points there after making a purchase; that in turn requires payment of a fee to the auction site. Bhattacharjee and Goel have derived a model for what this fee should be [Bhattacharjee and Goel, 2005]; they call the necessary property "inflation resistance".

We proceed with the details of each scheme. We emphasize on the fact that our schemes will work with any implementation of the underlying primitives (anonymous credential systems, blind signatures, digital cash) as long as the master public and secret keys for peers in our system are of the same form as those in the underlying e-cash scheme and anonymous credential system. That is, the key generation algorithms Ukeygen, EC.UKeyGen, and PS.Ukeygen are all the same.[5]

Our scheme will also require a zero knowledge proof of knowledge of both the master secret key corresponding to a master public key, and the secret information of a nym's owner (which is given as an output of the AC.FormNym operation). Thus, when instantiating our scheme with specific primitives, it is useful to choose underlying primitives that admit

[5]As discussed in the high level summary before, an important part our system setup is the assumption that peers are motivated to keep their master private key secret. For this reason, it is beneficial to have the master public and private keys be part of an external PKI which is used for other purposes (e.g., signing documents) outside our system.

efficient proofs of this form.

8.2.2.1 Base Scheme

For the **setup** of our reputation system, the bank B does the following:

- executes $\mathsf{EC.BKeyGen}$ procedure of e-cash scheme to create a digital signature key-pair (pk_{B}, sk_{B}). This is the key-pair that will be used for creating the repcoins. Bank publishes pk_{B}.
- executes $\mathsf{BS.BkeyGen}$ procedure of blind signatures scheme to create a blind signature key pair to be used in the Reputation Deposit procedure (pk^b_{B}, sk^b_{B}). Bank publishes pk^b_{B}.
- defines fixed reputation levels l_i, represented by a group G_i. These "reputation" groups — although managed by Bank — play a role similar to the one organizations play in anonymous credential systems. For each one of these groups, Bank runs $\mathsf{AC.OKeyGen}$ protocol to generate public-secret key pairs (pk_{G_i},sk_{G_i}). Bank also publishes pk_{G_i}s.
- does the appropriate setup (if any) for the pseudonym generation. For example, this may involve selecting an appropriate algebraic group G_p.

On the peers' side, each peer U_i invokes $\mathsf{EC.UKeyGen}$ to create a master public-secret key-pair ($pk_{\mathsf{U}_i}, sk_{\mathsf{U}_i}$).

Generation of Pseudonyms. Each peer generates his own pseudonyms. There is no particular structure imposed on the pseudonyms, and they need not be certified or registered with Bank (or any other entity). The only requirement is that the pseudonym generation leaves the owner with some secret information (e.g., the random string used for the generation procedure), such that possession of this information proves ownership of the pseudonym. We will also need such a proof to be executed. Thus, in principle, we can simply use a random string r as the secret information and $\mathsf{P} = f(r)$ as the pseudonym, where f is some one-way function, with an associated zero-knowledge proof of knowledge

of the inverse of P. However, a more efficient solution is to let the pseudonym generation procedure to be a digital signature key generation, keeping the signing key as the secret information and the verification key as the pseudonym. Here, being able to produce valid signatures will prove ownership of the pseudonym, without a need for a zero-knowledge proof.

RepCoin Withdrawal. RepCoin Withdrawal takes place between Bank B and a peer U. Both U and B engage in $\mathsf{EC.Withdraw}$ procedure of a e-cash scheme. For simplicity purposes, we assume that a wallet W of n repcoins has been withdrawn. Since the only properties related to repcoins are anonymity of an honest withdrawer and repudiation of any double spender, the wallet can be like the one suggested in [Camenisch *et al.*, 2005], or n separate digital coins withdrawn through any known e-cash scheme.

Reputation Award. This procedure is executed between two pseudonyms, one (i.e., $\mathsf{P_U}$) belonging to a peer U and one (i.e., $\mathsf{P_M}$) belonging to a peer M. Both engage in $\mathsf{EC.Spend}$ protocol of a e-cash scheme. However, this protocol takes place strictly between the two pseudonyms $\mathsf{P_U}$ and $\mathsf{P_M}$ instead of involving the actual identities U and M. Thus, $\mathsf{P_U}$ gives a repcoin to $\mathsf{P_M}$, where no information about identities of the parties involved is revealed.

Reputation Update. This protocol is invoked when a peer M wants to increase his reputation based on the repcoins that his pseudonyms have received since the last time he updated his reputation record. As previously discussed, maintaining unlinkability between a pseudonym and its owner is a crucial feature of our system. Towards this end, a single interaction for update (with a merchant presenting himself to Bank either as a peer or as a pseudonym) will not work, as we explain below.

Assume peer M wants to deposit a repcoin he received as $\mathsf{P_M}$ from pseudonym $\mathsf{P_U}$ of User U. Note that no one except M knows who is the owner of $\mathsf{P_M}$. Given the fact that U knows the exact form of the repcoin he gave to M, if M tried to deposit the repcoin by presenting himself as M to Bank, a collusion of Bank and U would reveal that M is the owner of $\mathsf{P_M}$. Trying to solve this by letting M "rerandomize" the repcoin in some way

before depositing it presents problems for enforcing the no over-awarding requirement. On the other hand, if Reputation Update procedure was done by the pseudonym $\mathsf{P_M}$ of M, there would be a problem in persuading the Bank to update M's record without revealing that M is the owner of $\mathsf{P_M}$.

Therefore, our Reputation Update protocol has two stages. First, $\mathsf{P_M}$ contacts Bank and gets a blind permission from it that shows a repcoin has been deposited and is valid. Second, M deposits that blind permission. In particular, the following procedure takes place:

1. *Obtaining Blind Permission.* Peer M executes $\mathsf{EC.Deposit}$ procedure of e-cash scheme using his pseudonym P_M, but here the actual deposit does not happen. Rather, if Bank B accepts the repcoin, M gets from B a blind signature on a random message. That is, P_M sends to B a repcoin that it has received. If B accepts the coin as valid, $\mathsf{P_M}$ chooses a random message C and gets a blind signature of C: σ^b_{B}. We call $(C, \sigma^b_{\mathsf{B}})$ a *blind permission.*

2. *Deposit of the Blind Permission.* M sends B the permission $(C, \sigma^b_{\mathsf{B}})$. Then, B checks if the tuple is fresh and increases the reputation of M.

Reputation Demonstration. This protocol is invoked when one peer wants to demonstrate his reputation to another peer, both interacting strictly through their pseudonyms. We will utilize predefined groups G_i corresponding to reputation levels l_i, which are managed by Bank. For a peer U who wants, via $\mathsf{P_U}$, to prove his reputation level l_i to a pseudonym $\mathsf{P_V}$ of a peer-verifier V, the protocol proceeds as follows:

1. If he has not done it before, U contacts the bank to register in the group G_i that corresponds to the desired reputation level l_i. U interacts with G_i (Bank) by invoking $\mathsf{PS.FormNym}$ protocol of a anonymous credential system, in order to generate a nym $N^{l_i}_{\mathsf{U}}$ for U under that group.[6] (U can generate as many nyms as he wants.)

[6]Recall that there is a big difference between pseudonyms and nyms. As discussed before, Pseudonyms are public-secret key-pairs, used as means to preserve peers' anonymity when involved in transactions. A nym of a peer will be associated with a particular reputation group. Bank, as the manager of the reputation

2. U contacts G_i, providing its master public pk_{U} key and a zero knowledge proof of knowledge π that he possesses the corresponding master secret key sk_{U}. U also presents $N_{\mathsf{U}}^{l_i}$ and a zero-knowledge proof π_N that it has been created correctly and he is the owner.

3. G_i checks that U is valid and that his reputation is indeed in that group (or higher), and executes $\mathsf{AC.GrantCred}$ to generate a credential $C_N^{l_i}$ for $N_{\mathsf{U}}^{l_i}$.

4. U interacts with the verifier $\mathsf{P_V}$ under his pseudonym $\mathsf{P_U}$. $\mathsf{P_U}$ proves by executing $\mathsf{AC.VerifyCred}$ that he possesses a credential from group G_i. Specifically, $\mathsf{P_U}$ proves that its owner has registered under a nym to G_i and has acquired — through that nym — a credential of membership.

8.2.2.2 Supporting Negative Reputation

The negative reputation extension of the base-scheme will work as well with the same general underlying primitives and requirements as the former. Briefly, we make the following changes w.r.t. the base scheme:

1. *In bank involvement.* In this version of our reputation protocol, the bank acts as the semi-trusted mediator in the rating procedure. The distribution of its functionality becomes — thus — critical for our system's efficiency, which is the reason we make use of blind group signatures (see, section 4.6 instead of plain blind signatures used in the base-scheme.

2. *In types of repcoins.* To support negativity in ratings, there are at least two types of repcoins: positive and negative, while for fairness purposes, users and merchants are required to use different types of repcoins. We, thus, result in a system with four types of reputation: (positive and negative) merchant reputation and (positive and negative) user reputation.

---

groups, will be able to link the nyms with the peer identities (master public key). In contrast, unlinkability of peers and pseudonyms is maintained, as per our security definitions.

3. *In types of repcoin permissions.* We make use of more types of them, used at the repcoin deposit procedure: $\mathsf{rperm}_{+/-}^{(\mathsf{u})/(\mathsf{m})}$, each corresponding to each of the four different types of reputation (repcoins) mentioned in the previous point.

4. *In the organization of Reputation Database.* To enforce the deposit of all the repcoins a peer receives, the bank, except for the reputation database D_{rep}, maintains $D_{rep-Wallets}$, which contains peers' repcoin withdrawal information.

For the **setup** of the system, the bank B does the following

- executes $\mathsf{EC.BKeyGen}$ procedure of e-cash scheme twice to create digital signature keypairs ($pk_{\mathsf{B}}^{(\mathsf{u})/(\mathsf{m})}$, $sk_{\mathsf{B}}^{(\mathsf{u})/(\mathsf{m})}$). The two e-cash settings will be used for the realization of the user and merchant type of repcoins. B publishes $pk_{\mathsf{B}}^{(\mathsf{u})/(\mathsf{m})}$.

- defines the maximum number of transactions each peer can participate in as merchant, $n^{(\mathsf{m})}$, and as user, $n^{(\mathsf{u})}$.

- executes $\mathsf{GBS.BkeyGen}$ procedure of group blind signatures scheme four times to create the blind group signature administration key pairs ($bgpk_{\mathsf{B}}^{(+/-,(\mathsf{u})/(\mathsf{m}))}$, $bgsk_{\mathsf{B}}^{(+/-,(\mathsf{u})/(\mathsf{m}))}$), one for each permission type. In addition, the Bank administration runs the $\mathsf{GBS.Join}$ on behalf of each Bank-member B_i to issue the individual group membership key pairs ($pk_{\mathsf{B}_i}^b$, $sk_{\mathsf{B}_i}^b$), respectively. These key pairs will be used in the reputation deposit procedure. Bank publishes $bgpk_{\mathsf{B}}^{(+/-,(\mathsf{u})/(\mathsf{m}))}$.

- defines fixed reputation levels l_i, represented by a group G_i. These "reputation" groups — although managed by Bank — play a role similar to the one organizations play in anonymous credential systems. For each one of these groups, Bank runs $\mathsf{AC.OKeyGen}$ protocol to generate public-secret key pairs (pk_{G_i},sk_{G_i}). Bank also publishes pk_{G_i}s.

- does the appropriate setup (if any) for the pseudonym generation. For example, this may involve selecting an appropriate algebraic group G_p.

On the peers' side, each peer U_i invokes $\mathsf{EC.UKeyGen}$ to create a master public-secret keypair ($pk_{\mathsf{U}_i}, sk_{\mathsf{U}_i}$).

Generation of Pseudonyms. The same procedure is followed as in the base-scheme case.

RepCoin Withdrawal. RepCoin Withdrawal takes place between the bank B and a peer U. Both U and B engage twice in $\mathsf{EC.Withdraw}$ procedure of the e-cash scheme, to issue two wallets of RepCoins: one for the merchant repcoins ($\mathsf{W}_{\mathsf{U}}^{(\mathsf{m})}$) and one for the user ones $\mathsf{W}_{\mathsf{U}}^{(\mathsf{u})}$. At the end of this operation, the Bank charges U's entry in $D_{rep-Wallets}$ with $|\mathsf{W}_{\mathsf{U}}^{(\mathsf{m})}| \leq n^{(\mathsf{m})}$ and $|\mathsf{W}_{\mathsf{U}}^{(\mathsf{u})}| \leq n^{(\mathsf{u})}$ repcoins, where $n^{(\mathsf{m})}$ and $n^{(\mathsf{u})}$ are the maximum merchant and user repcoins allowed to be issued to a particular peer per time period. As in the base-scheme case, the wallets may be like the one suggested in [Camenisch *et al.*, 2005], or separate digital coins withdrawn through any known e-cash scheme.

Reputation Award. This procedure is executed between a bank member B[7] and two peers, the user U, who plays the role of the consumer, and the merchant M. U and M interact through one of their pseudonyms, i.e., P_{U} and P_{M} respectively. B is chosen by both P_{U} and P_{M}. The Reputation Award consists of two phases, which are illustrated in figures 8.4 and 8.5:

1. *Transaction Preparation*, where P_{U} and P_{M} interact with each other, and the B so that they both commit in the transaction. In particular, P_{M} and P_{U} agree on a transaction ID and their roles in the transaction. To commit in transaction T_{id}, P_{U} spends to the bank one repcoin from his $\mathsf{W}_{\mathsf{U}}^{(\mathsf{u})}$ wallet in reference to T_{id}, while P_{M} does the same using $\mathsf{W}_{\mathsf{M}}^{(\mathsf{m})}$. Both parties receive back from B nominal reputation tickets to evaluate each other after the transaction takes place. More specifically

   (a) P_{U} and P_{M} agree on the transaction ID number, T_{id}, while with their pseudonym keys, they sign the following message:

   $$\{\mathsf{T}_{\mathsf{id}}, (\text{merchant: } \mathsf{P}_{\mathsf{M}}), (\text{user: } \mathsf{P}_{\mathsf{U}})\} \mid\mid \textit{date}$$

   into the $\sigma_{\mathsf{P}_{\mathsf{U}}}^{\mathsf{T}_{\mathsf{id}}}$ and $\sigma_{\mathsf{P}_{\mathsf{M}}}^{\mathsf{T}_{\mathsf{id}}}$ respectively.

[7] We denote the bank member as Binstead of B_i for simplicity.

(b) $\mathsf{P_U}$ deposits $\sigma^{\mathsf{T_{id}}}_{\mathsf{P_U}}$ to B, which checks for the latter's freshness. If $\sigma^{\mathsf{T_{id}}}_{\mathsf{P_U}}$ is valid, $\mathsf{P_U}$ and B commit in a $\mathsf{EC.Spend}$ procedure for the former to spend a repcoin from his $\mathsf{W}^{(\mathsf{u})}_{\mathsf{U}}$ wallet.

(c) $\mathsf{P_M}$ deposits $\sigma^{\mathsf{T_{id}}}_{\mathsf{P_M}}$ to B, who checks the validity of the latter, i.e., whether the same $\mathsf{T_{id}}$ has been deposited more than once and/or by any other pseudonym than $\mathsf{P_U}$, in which case, the $\mathsf{T_{id}}$ is rejected. If everything is valid, both, $\mathsf{P_U}$ and B commit in a $\mathsf{EC.Spend}$ procedure for the former to spend a repcoin from his $\mathsf{W}^{(\mathsf{m})}_{\mathsf{M}}$ wallet.

(d) B verifies that both, $\mathsf{P_U}$ and $\mathsf{P_M}$ have spent a valid repcoin from their corresponding wallets by executing $\mathsf{EC.Deposit}$ with itself and provides $\mathsf{P_U}$ with two non-blind tickets: (a) One to enable $\mathsf{P_U}$ to vote for $\mathsf{P_M}$ within a day after the transaction $\mathsf{T_{id}}$:

$$\mathsf{VoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_U}\rightarrow\mathsf{P_M}} = \mathsf{Sig_B}(\mathsf{P_U} \text{ vote for } \mathsf{P_M} \text{ on } \mathsf{T_{id}}),$$

and (b) one to enable $\mathsf{P_U}$ to collect the reputation vote $\mathsf{P_M}$ has assigned to the former:

$$\mathsf{AcceptVoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_U}\leftarrow\mathsf{P_M}} = \mathsf{Sig_B}(\mathsf{P_U} \text{ accepts } \mathsf{T_{id}}\text{-vote from } \mathsf{P_M}).$$

In a similar way, $\mathsf{P_M}$, obtains $\mathsf{VoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_M}\rightarrow\mathsf{P_U}}$ and $\mathsf{AcceptVoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_M}\leftarrow\mathsf{P_U}}$. $\mathsf{VoteTicks}$ should be deposited within a day after the transaction took place, while $\mathsf{AcceptVoteTicks}$ can only be deposited after a day the transaction $\mathsf{T_{id}}$ took place. If any participant ($\mathsf{P_M}$/ $\mathsf{P_U}$) of the transaction fails to commit to it, while the other party has already spend one of its repcoins, B issues special blind tokens of neutral value, which acts as a verification that the corresponding pseudonym accidentally used one of its repcoins. We may call them $\mathsf{VoidTicks}$.

2. *Actual Transaction,* where the actual transaction takes place.

3. *Reputation Assignment*, where both parties $\mathsf{P_U}$ and $\mathsf{P_M}$, rate each other. In particular, $\mathsf{P_U}$ deposits

$$\mathsf{VoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_U}\rightarrow\mathsf{P_M}}, \text{Reputation for } \mathsf{P_M}, \text{date}$$

to the bank. The banks checks whether the $\mathsf{VoteTick}$ deposited is fresh and matches its database information, i.e., if $\mathsf{T_{id}}$ is indeed an at most one day old transaction between $\mathsf{P_U}$ and $\mathsf{P_M}$, and logs $\mathsf{P_U}$'s rating for $\mathsf{P_M}$ in D_{rep}. $\mathsf{P_M}$ does the same with $\mathsf{VoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_M}\rightarrow\mathsf{P_U}}$.

Reputation Update Procedure, where each user updates his reputation record. This procedure, which is depicted in fig. 8.5, involves two steps:

1. $\mathsf{P_U}(\mathsf{P_M})$ deposits the $\mathsf{AcceptVoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_U}\leftarrow\mathsf{P_M}}$ ($\mathsf{AcceptVoteTick}^{\mathsf{T_{id}}}_{\mathsf{P_M}\leftarrow\mathsf{P_U}}$) to the bank B to receive its reputation.

2. B verifies the deposited $\mathsf{AcceptVoteTicket}$'s validity, i.e., whether they refer to a valid transaction between $\mathsf{P_U}$ and $\mathsf{P_M}$ having taken before more than a day. The bank then retrieves the reputation value $\mathsf{P_M}(\mathsf{P_U})$ has assigned to $\mathsf{P_U}$ ($\mathsf{P_M}$) through the corresponding $\mathsf{VoteTicket}$ and commits with the former in a $\mathsf{BGS.Sign}$ procedure to provide $\mathsf{P_U}(\mathsf{P_M})$ with the corresponding type of blind permissions($\mathsf{rperm}^{(\mathsf{u})/(\mathsf{m})}_{+/-}$).

3. Each peer P gathers all of his blind rperms and deposits them to B, who then updates P's logs. In any case, the following should hold:

   (a) $|\mathsf{W}^{(\mathsf{u})\prime}_{\mathsf{P}}| + (\#\mathsf{rperm}^{(\mathsf{u})}) + \#\mathsf{VoidTicket}^{(\mathsf{u})} = |\mathsf{W}^{(\mathsf{u})}_{\mathsf{P}}|$ and

   (b) $|\mathsf{W}^{(\mathsf{m})\prime}_{\mathsf{P}}| + (\#\mathsf{rperm}^{(\mathsf{m})})\#\mathsf{VoidTicket}^{(\mathsf{m})} = |\mathsf{W}^{(\mathsf{m})}_{\mathsf{P}}|$.

   If these equations do not hold, then P will be assigned a negative reputation point for each coin missing (default negative reputation).

Reputation Demonstration. The same procedure is followed as in the base-scheme case.

8.2.3 Discussion

In this section we demonstrate how security and privacy requirements are achieved in our scheme. As far as deployability is concerned, in the absence of a concrete implementation, it is hard to make concrete statements about practical issues. Worse yet, our main result is a framework which can accommodate different algorithms. That said, there are at least two areas that deserve further attention, performance and system security.

8.2.3.1 Security - Privacy

The following theorem states the correctness, privacy and security of our general scheme.

Theorem 1 *If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure, then our scheme satisfies* correctness*,* peer-pseudonym unlinkability*,* pseudonym-pseudonym unlinkability*,* no over-awarding*,* exculpability*, and* reputation unforgeability*.*

We prove the above theorem by showing the following lemmas hold.

Lemma 2 *If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure, then our scheme satisfies* correctness*.*

Proof. From the correctness of the secure e-cash scheme and the secure blind signature scheme, our scheme satisfies the first condition of the correctness. The correctness of the secure anonymous credential system guarantees the second condition.

Lemma 3 *If the underlying primitives (anonymous credential system, E-Cash system, and blind signatures) are secure, then our scheme satisfies* peer-pseudonym unlinkability*.*

Proof. In our scheme, pseudonyms are random element. Therefore, there is no link between pseudonyms and public keys. Now, as in the definition of peer-pseudonym unlinkability, consider a sequence of arbitrary operations, a target pseudonym P, and the set H of uncorrupted Peers that appear consistent with P in terms of their reputations. Since we are assuming anonymous and secure channels, the adversary's view includes all the operations involving corrupted parties (including Bank). We claim that this view is consistent with the target pseudonym P belonging to any of the Peers in H. Indeed, from the anonymity of the secure e-cash scheme, RepAward protocols executed in the sequence between a corrupt and an uncorrupt pseudonym (whether it is P or another pseudonym), do not leak any information about the pseudonym's owner. From the blindness of the secure blind signature scheme, RepCoinDeposit protocols excuted in the sequence do not leak any information

about a link between a pseudonym and the owner of the pseudonym. From the unlinkability of the secure anonymous credential system, ShowReputation protocols executed in the sequence do not leak any information about the owner of a pseudonym.

Moreover, even upon seeing the changes in D_{rep}, the adversary cannot guess the owner of the target pseudonym, since the owner may not have deposited his repcoin(s) or blind permission(s) yet.

Lemma 4 *If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure, then our scheme satisfies* pseudonym-pseudonym unlinkability*.*

Proof. As in the definition of pseudonym-pseudonym unlinkability, consider a sequence of arbitrary operations, the target pseudonyms $\mathsf{P}_1, \mathsf{P}_2$, and the set H of uncorrupted peers that appear consistent with both P_1 and P_2 in terms of their reputations. Since we are assuming anonymous and secure channels, the adversary's view includes all the operations involving corrupted parties.

First, consider the case where Bank is not corrupt. From the anonymity of the secure E-cash scheme, Award protocols executed in the sequence between a corrupt and an uncorrupt pseudonym (whether it is P_1 or P_2 another pseudonym), do not leak any information about the pseudonym's owner. From the blindness of the secure blind signature scheme, RepCoinDeposit protocols executed in the sequence do not leak any information about a link between a pseudonym and the owner of the pseudonym. From the unlinkability of the secure anonymous credential system, ShowReputation protocols executed in the sequence do not leak any information about the owner of a pseudonym. Moreover, pseudonyms are random element so that there is no link between pseudonyms. Therefore, the adversary cannot tell whether P_1 and P_2 belong to the same peer.

Second, consider the case when Bank is also corrupt. As long as $|H| > 2$, the adversary cannot tell whether P_1 and P_2 have the same owner. Using a similar argument to that in the proof of Lemma 3, the adversary's view is consistent with the target pseudonym P_1 belonging to any of the Peers in H. Likewise, the view is also consistent with P_2 belonging to any peer in H. Therefore, the adversary cannot tell whether P_1 and P_2 belong to the same peer.

Lemma 5 *If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure, then our scheme satisfies* no over-awarding.

Proof. For the first condition of no over-awarding, from the property of the identification of double-spenders of the e-cash scheme, if a peer U has double-awarded a repcoin, then Bank can identify him.

For the second condition, consider any coalition of peers where the maximum number of repcoins allowed to them is N. Then, the number of different serial numbers of repcoins that they can generate is at most N from the balance property of the e-cash scheme.

For the final condition, the honest-but-curious behavior of Bank guarantees one valid blind permission per any repcoin that is accepted but not double-awarded. Also, from the unforgeability of the blind signature scheme, no other valid blind permission can be forged. Upon receiving the perm, the honest-but-curious Bank will eventually increase exactly one reputation point.

Lemma 6 *If the underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure, then our scheme satisfies* exculpability.

Proof. Simply from the exculpability of the secure e-cash scheme, our scheme satisfies the exculpability.

Lemma 7 *If underlying primitives (anonymous credential system, e-cash system, and blind signatures) are secure, then our scheme satisfies* reputation unforgeability.

Proof. From the unforgeability and consistency of credentials of the secure anonymous credential system, our scheme satisfies the first condition of reputation unforgeability. In addition, non-transferability of the secure anonymous credential system guarantees the second condition in our scheme.

8.2.3.2 Performance

In general, our protocol is neither real-time nor high-performance. We are not proposing per-packet operations; most of what we do is per-user or per-purchase. As such, performance is not critical. Even so, it does not appear to be a problem. A full performance analysis of our base scheme is given here.

In this section we give a specific instantiation of our base scheme, where we make use of the anonymous credential system by Camenisch and Lysyanskaya [Camenisch and Lysyanskaya, 2001] (denoted by CL), the e-cash scheme by Camenisch et al. [Camenisch *et al.*, 2005] (denoted by CHL), and the blind signature scheme by Okamoto [Okamoto, 2006] (denoted by Ok). We do so in order to present a concrete and efficient construction (we include the efficiency analysis, relying on that of the underlying primitives, with each of the operations).

For the *Setup*(1^k) of the system, Bank B does the following:

- executes $\mathsf{CHL.BKeygen}(1^k)$ to generate an e-cash key pair $(pk_{\mathsf{B}}^{\mathsf{ec}}, sk_{\mathsf{B}}^{\mathsf{ec}})$, and publishes $pk_{\mathsf{B}}^{\mathsf{ec}} = (g_{\mathsf{ec}}, \hat{g}_{\mathsf{ec}}, \tilde{g}_{\mathsf{ec}})$.
- executes $\mathsf{Ok.KeyGen}(1^k)$ to generate a blind signature key pair $(pk_{\mathsf{B}}^{\mathsf{bs}}, sk_{\mathsf{B}}^{\mathsf{bs}})$ and publishes $pk_{\mathsf{B}}^{\mathsf{bs}}$.
- executes $\mathsf{CL.OKeyGen}(1^k)$ for each reputation group G_i $(1 \leq i \leq k)$ to generate the anonymous credential system key pair $(pk_{\mathsf{B}}^{\mathsf{ac}_i}, sk_{\mathsf{B}}^{\mathsf{ac}_i})$ for G_i, and publishes $pk_{\mathsf{B}}^{\mathsf{ac}_i} = (n_{\mathsf{ac}_i}, a_{\mathsf{ac}_i}, b_{\mathsf{ac}_i}, d_{\mathsf{ac}_i}, g_{\mathsf{ac}_i}, h_{\mathsf{ac}_i})$.
- creates a cyclic group $G_p = \langle g_p \rangle$ of order $p = \Theta(2^k)$ where the DDH assumption holds. This algebraic group is used for pseudonym generation on the peer's side.

On the peers' side, each peer U executes $\mathsf{CHL.UKeyGen}(1^k)$ to obtain $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) = (g_{\mathsf{ec}}^{x_{\mathsf{U}}}, x_{\mathsf{U}})$, and publishes pk_{U}. Note that x_{U} will be used as the master secret key of Uin the anonymous credential system (and this discrete-log based key is a reasonable choice for a more general PKI key as well).

Operations.

1. *Generation of Pseudonyms.* Each peer generates his pseudonyms locally using $\mathbb{G}_p$. Specifically, he chooses a random number $r_i \in \mathbb{Z}_p$ and compute $g_p^{r_i}$. The value $g_p^{r_i}$ is considered a pseudonym $\mathsf{P_U}^i$ of peer U.

2. *RepCoin Withdrawal.* A peer U executes $\mathsf{CHL.Withdraw}$ with Bank, and obtains a wallet W of 2^w repcoins. This procedure takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

3. *Reputation Award.* A pseudonym $\mathsf{P_U}$ gives a repcoin to $\mathsf{P_M}$ by executing $\mathsf{CHL.Spend}$ with $\mathsf{P_M}$. This procedure also takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

4. *Reputation Update.*

   (a) *Obtaining Blind Permission.* A pseudonym $\mathsf{P_M}$ and Bank B participate in $\mathsf{CHL.Deposit}$ protocol, which takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds. If $\mathsf{CHL.Deposit}$ accepts, $\mathsf{P_M}$ acquires the blind permission $\sigma_{\mathsf{B}}^{\mathsf{bs}} = \mathsf{Ok.Sign}(sk_{\mathsf{B}}^{\mathsf{bs}}, r_{perm})$ where r_{perm} is a random message. Obtaining the blind permission takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

   (b) *Deposit of the Blind Permission.* M (the owner of $\mathsf{P_M}$) sends $\sigma_{\mathsf{B}}^{\mathsf{bs}}$ to B. B checks if the permission $(r_{perm}, \sigma_{\mathsf{B}}^{\mathsf{bs}})$ is fresh; if so, it increases M's reputation value. This procedure takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

5. *Reputation Demonstration.* Suppose that a pseudonym $\mathsf{P_U}$ asks $\mathsf{P_M}$ to demonstrate its reputation level, and that M (the owner of $\mathsf{P_M}$) wants to show to $\mathsf{P_U}$ that it belongs to G_i, i.e., his reputation is at least at level l_i.

   (a) *Obtaining a nym under G_i.* M contacts Bank B and executes $\mathsf{CL.FormNym}$ with respect to G_i[8]. Let $N_{\mathsf{M}}^{l_i}$ be the nym that M obtained from this procedure. Note that $N_{\mathsf{M}}^{l_i}$ is of the form: $g_{\mathsf{ac}_i}^{x_{\mathsf{U}}} \cdot h_{\mathsf{ac}_i}^{r}$. This takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

   (b) *Obtaining a credential for G_i.* M contacts B, and he sends B the message

[8] We use both protocol1 and protocol6 of [Camenisch and Lysyanskaya, 2001] instead of just protocol1 to ensure the non-transferability of credentials.

$(pk_{\mathsf{M}}, N_{\mathsf{M}}^{l_i})$. Then, M executes with B a zero-knowledge proof of knowledge

$$pk\{(\alpha, \beta) : pk_{\mathsf{M}} = g_{\mathsf{ec}}^{\alpha}, N_{\mathsf{M}}^{l_i} = g_{\mathsf{ac}_i}^{\alpha} \cdot h_{\mathsf{h}_i}^{\beta}\}.^{9}$$

This takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

Now, B verifies the proof. If the proof is verified so that M is eligible for a credential of the group G_i, B executes the $\mathsf{CL.GrantCred}$ ($\mathsf{protocol4}$) with respect to G_i. Let C_{l_i} be the output credential. This takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

(c) *Showing reputation using the credential.* $\mathsf{P_M}$ contacts $\mathsf{P_U}$ and executes $\mathsf{CL.VerifyCred}$ ($\mathsf{protocol3}$) with respect to G_i to prove that owner of $\mathsf{P_M}$ has a credential for the group G_i. This takes $\mathrm{O}(1)$ exponentiations and $\mathrm{O}(1)$ rounds.

8.2.3.3 Other Security Issues

In addition to the anonymous peer-to-peer communication necessary for the underlying application, there is now a new communications path: from each party to the bank. Parties who are engaging in our protocol will need to contact the bank. This provides another channel that might be detected by, say, the attacks described in [Kesdogan *et al.*, 2006b]. Indeed, there may exists a sort of "meta-intersection attack" [Danezis and Serjantov, 2004]: the peer-to-peer traffic alone may not be suspicious, but it coupled with conversations with the bank might be sufficient for identification.

A second area for security concern is CPU consumption. Our scheme (see previous section) requires public key operations; these are CPU-intensive. An attacker who has a identified a candidate participant in real-time might be able to connect to it — we are, after all, talking about peer-to-peer systems — and measure how its own communications take. The obvious defense is to make sure that any given operation takes constant time; in turn, this likely means preconfiguring each peer node with a maximum number of concurrent connections supported.

---

[9] This proof can be parsed as "I know the exponent α and β that was used in generating pk_{M} and $N_{\mathsf{M}}^{l_i}$". See [Camenisch and Stadler, 1997; Camenisch and Lysyanskaya, 2001] for more detail. The proof can be regarded as an authentication procedure.

8.2.4 Related Work

A number of papers have addressed the issue of reputation and privacy.

There are many papers on reputation systems for peer-to-peer networks. Most focus on building distributed reputation systems, rather than worrying about privacy; [Gupta *et al.*, 2003] is typical.

The difficulty of building systems like this is outlined by Dingledine, Mathewson, and Syverson [Dingledine *et al.*, 2003]. They present a number of similar systems and show why bolting on reputation is hard.

A typical approach is typified by [Voss *et al.*, 2005], who incorporate privacy into their scheme. However, their system does not provide unlinkability. It also requires a trusted "observer" module for full functionality.

The work by Kinateder et al. [Kinateder and Pearson, 2003; Kinateder *et al.*, 2005] is close to ours. The system in [Kinateder and Pearson, 2003] differs from ours in two notable ways. First, its reputations are linkable. Indeed, they see this as a virtue, in that recommendations can be weighted depending on the reputation of the recommender. Second, they assume a trusted hardware module (i.e., a TPM chip) on every endpoint. In [Kinateder *et al.*, 2005], they describe a more general system based on UniTEC [Kinateder and Rothermel, 2003]. Reputation statements are signed by a pseudonym's private key. Unlinkability is achieved by switching public keys. Apparently, the UniTEC layer can share reputations between different pseudonyms, but the authors do not explain how this is done. Presumably, this is handled by bookkeeping at that layer. More seriously, although they assert that a trusted module is desirable but not necessary, they do not explain how that could work, and in particular how they can prevent cheating.

Pavlov et al. [Pavlov *et al.*, 2004] present a system, based on secret-sharing, which has many of the same properties as ours. However, it depends on locating "witnesses", other parties with knowledge of the target's reputation. In a sufficiently-large community with a low density of interaction, this may be difficult. Furthermore, it does not provide unlinkability; witness testify about a known party's past behavior.

Another work related to ours is Voss [Voss, 2004] and Steinbrecher [Steinbrecher, 2006]. In both of the systems, users interact with each other through pseudonyms, and reputation

is strongly connected to identities. In fact, in [Voss, 2004] reputation points are implemented as coins, which may have positive or negative value. However, in both cases, Trusted Third Parties[10] are required to ensure unlinkability between identities and pseudonyms.

Approaches other than reputation systems have also been presented to deal with misbehaving users in anonymous or pseudonymous systems. Belenkiy et al. [Belenkiy *et al.*, 2007] make use of endorsed e-cash to achieve fair and anonymous two-party protocol wherein parties buy or barter blocks of data. Whereas e-cash stands for reputation in our scheme, e-cash stands for actual money in their scheme; a peer uses e-cash to buy data from other peers. Johnson et al. [Johnson *et al.*, 2007] focus on protecting a service in Tor from a malicious user without blocking all the exit Tor nodes. In particular, they present a protocol where misbehaving anonymous users are blacklisted by servers.

8.3 Conclusion

In this chapter, we dealt with privacy and security issues related to individuals' online transactions: purchase profiling performed through the product delivery by the merchant or the delivery companies and the lack of credibility in existing pseudonymous rating systems. In particular, we presented:

- a real-world applicable delivery service protocol for online purchases with guaranteed merchant-customer unlinkability and recipient anonymity w.r.t. the merchant and/or the delivery companies involved; our protocols utilize similar techniques to the Tor[Dingledine *et al.*, 2004] anonymity network and support package tracing and mail delivery proof. As opposed to currently deployed anonymous delivery techniques, recipient's address is concealed even from the company paid to perform the delivery.

- a identity-bound reputation system, which as opposed to existing pseudonym-based reputation systems, reputation is bound to the identities of peers, ratings are fair, i.e., no party may avoid being rated, all with guaranteed users' transaction unlinkability and user anonymity; we define a security model for such systems and build a set of protocols that satisfy its requirements.

[10] In [Steinbrecher, 2006] TTP appear in the form of designated identity providers.

Online payments is the only part of online transactions we haven't dealt yet. However, it being part of the online banking activities of a peer, we integrated it in the following section.

Chapter 9

Online Banking

One of the hardest realms in which to achieve privacy is finance. Credit card purchases, bank account management and the corresponding credit risk activities constitute the most popular components of it. It is conspicuous how important the security of financial data becomes, as it involves sensitive individual's information, while the corresponding security mechanisms augment current online privacy threats.

A very representative example is *credit cards*. Credit cards have many useful properties which is the reason they have become very popular. Apart permitting delayed payment, they provide users with logs of their own transactions, receipts, and the opportunity to challenge and correct erroneous charges. However, being closely related to their owners' identities, credit cards' extended use constitutes a serious threat to consumers' privacy: *Frequent occurrences of credit card losses, credit card number based-impersonation attacks as well as human nature errors, i.e. overcharge of a client, make it necessary for cardholders to be able to monitor their own transaction activity and for merchants to provide banks with detailed description of each credit card transaction. Under the umbrella of the need of immediate charge justification/correction, each bank, which is no more trusted than the people operating it, acquires a global view of its customers' transaction activity.*

Constituting an easy way to remotely use the funds of a bank account, debit cards raise similar issues with the credit cards. To authorize a debit-card purchase and to provide immediate justification of each amount subtracted, banks maintain detailed log of all the merchants contacted by the debit-cardholder.

Acting towards the bank protection, *credit score* mechanism is also justified to acquire a large amount of information regarding individuals' transactional behavior: *Banks claim that the more accurate the user-profiling is, the more the credit score reflects individuals' credibility, and the better protected banks are against non-eligible individuals.*

Unfortunately, banks are no more trusted than the people operating it; they can use the same information to build and sell profiles of their customers. None of the currently deployed credit card systems offer consumer's privacy towards banks.

Anonymous bank accounts with untraceable activity would be a solution, which is partly adopted in this chapter. In particular we adopt pseudonymity as means for achieving transactional privacy w.r.t. bank accounts. In such a system, an individual has a multitude of separate, unlinkable identities each used for each account, that can be used as desired. A separate pseudonym can be used for each account, thus preventing linkage between different sorts of activities. In many cases, there are also security benefits to having multiple pseudonymous accounts. Often, knowledge of a "routing number" (effectively, the bank's identity) and an account number are sufficient to *withdraw* money from an account as well as deposit money into it. Having multiple pseudonymous accounts, that can be opened for a special purpose and closed when no longer needed, could prevent such incidents.

The challenge in this case would be the realization of supportive functionalities, i.e., credit card service and risk management mechanisms. In addition, society often requires that other information about bank accounts be disclosed. For example, in the U.S., banks and other financial institutions are required to report interest or dividend payments, since they are generally considered to be taxable income. Some jurisdictions require that a portion of the interest be paid directly to the government; other jurisdictions impose taxes on actual balances. At the same time the U.S. credit score is a bank generated, public variable which characterizes each individual's payment credibility. As there is a strong need for it to be updated regularly, these requirements conflict with a desire for privacy.

We show that for tax purposes, neither banks nor the government need to know who owns a particular bank account. Rather, banks need only know that transactions are limited to authorized parties; governments need only ensure that balances and income are properly reported and taxed. An ideal system would be one where an individual could open a bank

account without disclosing his or her real identity, while still ensuring that the relevant tax authorities would receive accurate reports.

We present a solution that accomplishes these goals. In particular, we present a system, where individuals having obtained membership in a bank, may (if eligible):

1. open privately an arbitrary number of anonymous accounts, i.e., without anyone being able to link those accounts to their owner, or group them together as owned by the same individual (see section 9.1).

2. be withheld taxes in proportion to their aggregated balance to the bank, without endangering their accounts' privacy (see section 9.1).

3. show how the taxation mechanism mentioned before can be applied for the privacy preserving calculation of the credit score of the individuals (section 9.3).

4. issue anonymous credit cards, which they can use in online and offline purchases to achieve transaction anonymity and unlinkability even towards the banks; our credit card mechanism supports all current credit card functionalities, error correction and expense reporting etc. (section 9.2).

In the case of taxation and credit card mechanism we develop and discuss the corresponding security and privacy models.

9.1 Anonymous but Taxable Bank Accounts

In this section we will present our bank account mechanism. In our system, an honest individual is able to own and handle multiple bank accounts, not entirely connected to his identity, while being taxed in proportion to his balance in a privacy preserving fashion.

In particular, assuming an architecture consisting of banks, users (individuals) and a taxation authority, our system supports the management and taxation of two types of bank accounts:

1. "Nominal" accounts where the identity of account owners are known to the bank and accounts' activity can be fully traced.

2. "Anonymous" accounts, where the identity of the owner is concealed from the bank. Anonymous accounts' activity traceability varies according to their owners' privacy requirements.

Our protocols a couple of interactions. A user registers at a bank once (User Registration) and requests to open accounts. Based on his credentials, the bank may approve or reject his request. If the requested accounts are anonymous, and if the user is eligible, the bank issues anonymous account permissions which can later be used by the user anonymously, for the latter to be authorized to open anonymous accounts. The user then creates extra account-related management credentials (Account Opening), i.e., account management authorization information. On an annual basis, the user collaborates with the bank to issue tax reports for each of his accounts (Tax Report Issue), which he later deposits to the bank or taxation authority for his overall withheld tax to be calculated (Tax Report Deposit).

In this architecture we combine privacy and security. Privacy requires that the owner of each account is hidden among the owners of anonymous accounts (account – account-owner unlinkability), while separate accounts of the same user are unlinkable one to the other (account–account unlinkability). To achieve privacy, we extend the aforementioned operations with an extra stage intervening the last two operations (Tax Report Issue and Tax Report Deposit), the Tax Report Transformation stage, where the account owner plays the role of mixer and transforms his tax reports into a form unlinkable to their initial one. Security, on the other hand, requires that all users are fairly taxed, i.e., tax reports are unforgeable, of a single use and non-transferable between users (tax report unforgeability, non-transferability), while misbehaving individuals are traced and punished (accountability). We achieve security by extending the existing instantiations of system operations with various privacy preserving primitives.

In the following section we will present the model for our system formally, i.e., we will present the entities and operations of our system while we define security and privacy in this context. In section 9.1.2 we present our protocols, while in section 9.1.4 we discuss how our requirements are satisfied. In section 9.1.5 we present previous work on similar issues.

9.1.1 A model for Anonymous but Taxable Bank Accounts

In this section, we present an overview of our system requirements, threat model and other underlying assumptions for our protocols. In addition, we enumerate the operations considered in the system, followed by the security definition. We note that some of these definitions were inspired by previous work on other primitives, such as [Camenisch and Lysyanskaya, 2001; Camenisch *et al.*, 2005].

9.1.1.1 System Entities

The entities involved are exactly the ones mentioned in chapter 3:

- *Users*, who open bank accounts and must pay taxes.
- *Banks*, who allow users to open accounts for the purpose of storing cash and handling financial transactions. They are responsible for reporting interest for income tax purposes. In accordance to the two types of accounts they support, each bank B maintains two databases:
  1. the $\mathsf{D}_{\mathsf{reg}}$, which contains the contact and credential information of its clients, as well as the nominal accounts' history information, and
  2. the D_{α}, which contains all the anonymous accounts' information, i.e., authorization information, account balance, etc.
- *Tax Authority* TA, which is responsible for ensuring that correct income taxes are paid by all users. Tax Authority corresponds to the U.S.'s Internal Revenue Service (IRS), the Canada Revenue Agency, the U.K.'s HM Revenue & Customs, etc.

We also assume that each individual owns an identity card and an IDC (see chapter 5 for more details), which is ultimately bound to a specific person. Banks use these identity cards to gain strong assurance of the identity of the person who registers to them.

9.1.1.2 Operations

To define the operations supported in our taxation mechanism more strictly we will use the following notation: when an operation is an interactive procedure (or a protocol con-

sisting of multiple procedures) between two entities A and B, we denote it by $\langle O_A, O_B \rangle \leftarrow \mathsf{Pro}(I_C)[A(I_A), B(I_B)]$, where Pro is the name of the procedure (or protocol). O_A (resp. O_B) is the private output of A (resp. B), I_C is the common input of both entities, and I_A (resp. I_B) is the private input of A (resp. B). We also note that depending on the setup, some operations may require additional global parameters (e.g., some common parameters for efficient zero-knowledge proofs, a modulus p, etc). Our system will need these additional parameters only when using underlying schemes that use such parameters, e.g., eCash systems or anonymous credential systems. To simplify notation, we omit these potential global parameters from the inputs to all the operations.

- $(pk_{\mathsf{B}}, sk_{\mathsf{B}}) \leftarrow \mathsf{Bkeygen}(1^k)$ is the key generation algorithm for the banks. We call pk_{B} the (master) public key of a bank B, and sk_{B} the master secret key of B.

- $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) \leftarrow \mathsf{Ukeygen}(1^k)$ is the key generation algorithm for the users. We call pk_{U} the (master) public key of a user U, and sk_{U} the master secret key of U.

- $\langle (\mathsf{P}, sec_{\mathsf{P}}), (\mathsf{P}, \mathsf{D_{reg}}') \rangle \leftarrow \mathsf{UserRegistration}(1^k, pk_{\mathsf{U}}, pk_{\mathsf{B}})[\mathsf{U}(sk_{\mathsf{U}}), \mathsf{B}(sk_{\mathsf{B}}, \mathsf{D_{reg}})]$ is the user registration procedure at the bank, which involves the pseudonym generation algorithm for users. The sec_{P} is the secret information used to generate the pseudonym P.

- $\langle \mathsf{perm}_\alpha, \mathsf{D_{reg}}' \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{AnonymousAccountRequest}\ (pk_{\mathsf{B}})\ [\mathsf{U}(sk_{\mathsf{U}}), \mathsf{B}(sk_{\mathsf{B}}, \mathsf{D_{reg}})]$. A user U requests to open one or more anonymous bank accounts to a bank B. B consults $\mathsf{D_{reg}}$ to check if U is eligible for it and if so, it provides U with a wallet of credentials perm_α.

- $\langle sec_\alpha, \mathsf{D}_\alpha' \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{AccountOpen}\ (pk_{\mathsf{B}})\ [\mathsf{U}(sk_{\mathsf{U}}, \mathsf{perm}_\alpha), \mathsf{B}(sk_{\mathsf{B}}, \mathsf{D}_\alpha)]$. U having requested successfully for an anonymous account, using his $cred_\alpha$ generates sec_α, i.e., the secret information which will authorize U to make use of his account. D_α is changed to reflect the updated user account information.

- $\langle T^i, \mathsf{D}_\alpha' \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{TaxReportIssue}\ (\mathsf{P}^{\mathsf{i}}, pk_{\mathsf{B}})\ [\mathsf{U}(sec_\alpha^i, sk_{\mathsf{U}}), \mathsf{B}(sk_{\mathsf{B}})]$. A user U owning account α_i (secret information sec_α^i) and the bank B collaborate for the former to issue a tax report for α_i. U demonstrates ownership of α_i and contributes his secret

information to issue tax report T^i. B notes in D_α that a tax report for α_i has been issued.

- $TT^i \ / \ \perp \leftarrow \ \mathsf{TaxReportTransform}(T^i, sk_{\mathsf{U}})$. A user U having issued a tax report T^i, transforms it in the unlinkable form TT^i, for which he can still demonstrate ownership.

- $\langle \top, \mathsf{D}_\alpha{}', \mathsf{D}_{\mathsf{reg}}{}' \rangle / \langle \perp, \perp \rangle \leftarrow \ \mathsf{TaxReportDeposit}\ (pk_{\mathsf{B}}, pk_{\mathsf{U}}, TT^1, \ldots, TT^N)\ [\mathsf{U}(sk_{\mathsf{U}}), \mathsf{TA}(sk_{\mathsf{TA}})]$. User U who has already issued the tax reports $T^1, \ldots, T^N$ and transformed them to $TT^1, \ldots, TT^N$ respectively, deposits them to the tax authority TA. If the tax reports are valid, fresh and correspond to U, TA notifies B to update $\mathsf{D}_{\mathsf{reg}}$.

- $TotalTax / \perp \leftarrow \ \mathsf{TotalTaxCalculation}\ [\mathsf{U}(sk_{\mathsf{U}}, T^1, \ldots, T^N), \mathsf{TA}(sk_{\mathsf{TA}})]$. The user U and the TA collaborate for the latter to calculate the overall tax amount withheld by U's accounts in bank B. The secret input of the user is his secret information and the initial tax reports $T^1, \ldots, T^N$.

9.1.1.3 Threat Model

We make the following assumptions:

- *Users may try to cheat.* A user trying to avoid paying taxes may attempt to lie regarding the tax he has been withheld and is motivated enough to attempt any type of forgery. We also assume that malicious users may collaborate in order to change their reported balance to their benefit, as long as through this collaboration they do not endanger their funds.

- *Banks are "honest but curious".* Aiming to maintain their clientele, banks are trusted to perform all their functional operations correctly, i.e., they issue credentials, open and update accounts as instructed by their customers. However, they may use the information they possess for other reasons, i.e., to sell credit card based profiles to advertising companies, while they may collaborate with tax authority to reveal the identity behind an anonymous account.

- *Tax Authority is considered to be "honest but curious".* Although we assume that it is operated by the government who wants to protect honest users, the tax authority

officials, however, are not assumed to protect privacy; indeed, there have been a number of incidents in the U.S. of privacy violations by tax authorities or by unscrupulous individuals employed by the tax authorities.

9.1.1.4 Requirements

Correctness, user *privacy* and *security* of the operations of the system are our core requirements. We will strictly define each of them.

Correctness requires that if an honest user U, who is eligible for opening anonymous accounts with an honest bank B, runs AnonymousAccountRequest and AccountOpen with B, then none will output an error message. Also, if honest U, has opened accounts $\alpha_1, \ldots, \alpha_N$ with honest B, and runs TaxReportIssue, then no one will output an error message, while when the user tries to deposit them and thus runs with TA TaxReportDeposit and TotalTaxCalculation no entity will output error message and they will output the aggregated tax withheld by honest U's accounts.

Privacy — generally equivalent to honest users' activity untraceability — in the context of our bank system is interpreted to the following:

1. *Account-Account-owner Unlinkability.* There should be no way for any entity or collaboration of entities, including the bank and tax authority, to link accounts to a particular user identity.

2. *Account-Account Unlinkability.* There should be no way for any entity or collaboration of entities, including the bank and tax authority, to link two or more accounts of the same user.

In both cases we consider an adversary who, having corrupted some parties including bank B, is participating in the system for some arbitrary sequence of operations executed by honest and by corrupted parties.

Taking in consideration that each anonymous account is taxable and managed through a pseudonym, we should consider both unlinkability properties w.r.t. pseudonyms and w.r.t.

tax reports. More specifically, *account – account-owner unlinkability* can be inducted to account-pseudonym – account-owner unlinkability and account – account tax-report (final form) unlinkability. The first requires that given a pseudonym $\mathsf{P_U}$ that does not belong to a corrupted party, the adversary can learn which user owns $\mathsf{P_U}$ no better than guessing at random among all non-corrupted peers that appear consistent with $\mathsf{P_U}$. Account – account tax-report unlinkability requires that given a first version of a tax report T that does not belong to a corrupted party, the adversary can learn which user owns T no better than guessing it at random among all non-corrupted users in D_α. In addition, given a transformed (final form) tax report TT, that does not belong to a corrupted party, the adversary can learn which pseudonym (and thus which account) it corresponds to no better than guessing at random among all pseudonyms (accounts) of non-corrupted users in D_α.

In a similar way, *account – account unlinkability* can be interpreted in account-pseudonym – account-pseudonym unlinkability and account-tax report – account tax-report unlinkabilty. *Account-pseudonym – account-pseudonym unlinkability* requires that given two pseudonyms $\mathsf{P}^1_\mathsf{U}, \mathsf{P}^2_\mathsf{U}$ that do not belong to corrupted parties, the adversary has no advantage in telling whether $\mathsf{P}^1_\mathsf{U}, \mathsf{P}^2_\mathsf{U}$ belong to the same user or not. Next, consider an adversary who corrupted some users and the bank as well. Tax report related unlinkability requires that, given two tax reports T^1, T^2 that do not belong to corrupted parties, the adversary has no advantage in telling whether they belong to the same user or not.

Security consists of the following properties:

1. *Fairness.* An accurate statement of the contents or tax liability of all accounts belonging to a given individual is reported to the tax authority per normal practice (i.e., quarterly or annually). More strictly: suppose that n users $\mathsf{U}_1, \ldots, \mathsf{U}_n$ collude together. Let the sum of the tax withheld by all of them together is

$$\mathsf{SumTax} = \sum_{i=1\ldots n} TotalTax^i,$$

where $TotalTax^i$ is U_i's tax amount withheld. Then, fairness requires that the group of users may report in total a minimum of SumTax amount withheld. By fairness, we also require that the following hold:

(a) *Tax Report non-Transferability.* No user should be able to exchange tax report(s) with any other user or use the tax report of another. Assuming two corrupted users U_1 and U_2, where U_1 has issued T^1. Tax Report non-Transferability requires that there is no valid transformation TT^1 of T^1 (through TaxReportTransform) for which the following happens with non negligible probability: if U_2 attempts to deposit TT^1 in honest B through TaxReportDeposit, B accepts.

(b) *Tax Report Unforgeability.* No user or coalition of users should be able to construct a valid tax report for his accounts, i.e., a tax report for which TaxReportDeposit is accepted by the tax authority or the bank.

2. *Privacy preserving Tax Reporting* (covered by privacy property).Tax reports, whether viewed individually or in aggregate, should not reveal any information about the owner or their activities beyond what is necessary for accurate taxation.

3. *Accountability.* Users who attempt to avoid paying taxes for their accounts are traced and punished. We may tentatively assume that this requirement falls in the margin of fairness property, as users who try to avoid paying taxes or attempt to report a higher amount than the amount withheld are caught.

9.1.2 Anonymous but Taxable Account System

In accordance to the two types of accounts it supports, each bank B maintains two databases:

1. the $\mathsf{D}_{\mathsf{reg}}$, which contains the contact and credential information of its clients, as well as the nominal accounts' history information, and

2. the D_{α}, which contains all the anonymous accounts' information, i.e., authorization information, account balance, etc.

As accountability imposes a "privacy-preserving" *centralization* critical, inside the bank, each user can be privately authenticated by demonstrating knowledge of a single master secret, ms_{U}, which he generates at the registration procedure. Users are highly motivated not to share their secret, which they use to open and manage their anonymous accounts. More, specifically the user utilizes his ms_{U} to issue, bank (blindly) authorized permissions

of single use, perm_α, which he later deposits anonymously. To manage his accounts, the anonymous user generate account pseudonyms, which are secretly, but provably, connected to their owner's $\mathsf{ms_U}$. As privacy requires, pseudonyms of the same user are totally unlinkable one with the other (account-account unlinkability), while pseudonyms, reveal nothing regarding the owner of the $\mathsf{ms_U}$(account owner anonymity) without their owner collaboration. However, when a user misbehaves, i.e., he attempts to open more accounts than he is authorized for, he risks that his $\mathsf{ms_U}$ is revealed and his identity and activity completely traced. Users are annually required to deposit to the tax authority an equal amount of tax reports to the number of accounts they own. For tax reporting, we may identify the following user-bank phases:

1. *Tax Report Generation.* It involves three stages:

   (a) *Tax-Report-number Acquisition*, where account owners obtain one valid tax-report-number (TRN) per account. It is important to note that TRNs are not linked to the accounts they are used for.

   (b) *Actual Report Generation*, where the account owner, contacts the bank through his account pseudonym, proves that he is the owner of the account — by demonstrating knowledge of the $\mathsf{ms_U}$ connected to the account pseudonym — and provides a verifiable commitment to both his $\mathsf{ms_U}$ and TRN. The bank then produces the prime version of the account's tax report:

   $$(T^\sigma, T^M) \;=\; (\mathsf{Sig}^x_{\mathsf{B}}(\text{TaxInfo}),\ \text{TaxInfo}),$$

   where

   $$\text{TaxInfo} \;= Tax \;||\; \text{Commitment}(\mathsf{TRN},\ \text{Master-Secret}),$$

   Tax is the tax withheld from the user's account and by $\mathsf{Sig}^x_{\mathsf{B}}(M)$, we denote a complicated procedure which involves bank's (x-multiple) signature on M. The exact number of bank signatures applied on M is not revealed to the user. However, the bank provides the user with a randomized token $\mathsf{SigInfo}$ which contains that information, in a form only readable by the taxation authority, along with re-randomization information $\mathsf{SITranform}$ for the user to make $\mathsf{SigInfo}$ unlinkable to its initial form.

(c) *Tax Report Transformation*, where the account owner, applies a transformation function F to both, the bank-signed tax report T^{σ}, and the corresponding unsigned message, ending up to the depositables

$$TT^{\sigma} = F(T^{\sigma}), \text{ and } TT^{M} = F(T^{M}).$$

The account owner also transforms SigInfo through SITranform.

The tax report consists of the pair: (TT^{σ}, TT^{M}).

2. *Tax Report Deposit.* Each user deposits all of his tax reports to the tax authority. The deposit of tax reports includes three stages:

(a) *Deposit of all the unused* perm_{α}. In this way, the tax ayuthority can accurately compute the number of anonymous accounts of each user.

(b) *Deposit of the depositable tax report pairs*, $(TT^{\sigma,i}, TT^{M,i})$ corresponding to each account owned by the user. The user proves that each tax report pair is valid, i.e., that it corresponds to bank signature(s) (according to the transformed version of SigInfo), that is fresh and was constructed using the same master secret as the rest of depositable tax reports of the same user (i.e., that all tax reports correspond to the same user).

(c) *Tax Amount Calculation procedure*, where, the tax authority collaborates with the user to process the individual tax reports and calculate the overall tax that user's accounts have been withheld.

9.1.3 Detailed Protocol Description

As mentioned before, the bank manages two different registries: one handling users' non-anonymous information (reg-setting) and accounts and another one handling anonymous accounts (α-setting). As each setting is realized as organizations in pseudonymous systems (see section 4.3 or [Camenisch and Lysyanskaya, 2001] for more details), the bank runs PS.OKeyGen twice, once for the reg-setting and once for the α setting. In particular, the bank:

- generates all the common system parameters (see [Camenisch and Lysyanskaya, 2001]): the length of the RSA moduli ℓ_n, the integer intervals $\Gamma =]-2^{\ell_\Gamma}, 2^{\ell_\Gamma}[$, which is basically the interval master-secrets belong to, $\Delta =]-2^{\ell_\Delta}, 2^{\ell_\Delta}[$, $\Lambda =]-2^{\ell_\Lambda}, 2^{\ell_\Lambda+\ell_\Sigma}[$, such that $\ell_\Delta = \epsilon(\ell_n + \ell_\Lambda) + 1$, where ϵ is a security parameter, and $\ell_\Lambda > \ell_\Sigma + \ell_\Delta + 4$.

- chooses two pairs (one for each setting) of random $\ell_n/2$-bit primes: $p'_{\mathsf{reg}}, q'_{\mathsf{reg}}$ and p'_α, q'_α, such that $p_x = 2p'_x + 1$ and $q_x = 2q'_x + 1$ are prime and sets modulus $n_x = p_x \cdot q_x$, where $x = \mathsf{reg}, \alpha$.

- chooses random elements $a_x, b_x, d_x, g_x, h_x \in QR_{n_x}$, where $x = \mathsf{reg}, \alpha$. In addition to the standard organization setup procedure of [Camenisch and Lysyanskaya, 2001], the bank also chooses random $k_\alpha, l_\alpha, m_\alpha, s_\alpha, z_\alpha \in QR_{n_\alpha}$.

Thus, the Bank's public-secret information for the two settings are

- $\{(n_{\mathsf{reg}}, a_{\mathsf{reg}}, b_{\mathsf{reg}}, d_{\mathsf{reg}}, g_{\mathsf{reg}}, h_{\mathsf{reg}}), (p_{\mathsf{reg}}, q_{\mathsf{reg}})\}$, for the reg-setting, and

- $\{(n_\alpha, a_\alpha, b_\alpha, d_\alpha, g_\alpha, h_\alpha, k_\alpha, l_\alpha, m_\alpha, s_\alpha, z_\alpha), (p_\alpha, q_\alpha)\}$, for the α-setting.

In addition to the aforementioned parameters, the bank generates a blind signature key pair $(pk^b_{\mathsf{B}}, sk^b_{\mathsf{B}})$ and an RSA signature key pair,

$$\{sk_{\mathsf{B}}, pk_{\mathsf{B}}\} = \{(d, p_\alpha, q_\alpha), (e, n_\alpha)\},$$

based on the α RSA-parameters and $1 < e < \phi(p_\alpha q_\alpha)$ and $de = 1(mod(\phi(p_\alpha q_\alpha)))$. e is given to the taxation authority (TA).

On the other hand, TA generates an encryption key pair (pk_{TA}, sk_{TA}) of a known randomized homomorphic encryption scheme (Paillier etc) and provides the bank with the encryption key (see section 9.1.4 or [Paillier and Pointcheval, 1999] for more details).

In what follows, we will assume that a user U collaborates with a bank B to open anonymous accounts, handle them and be taxed for.

User Registration. In this phase, U contacts B *in person* to create an entry in the latter's $\mathsf{D}_{\mathsf{reg}}$ registry. This is a prerequisite for users to open (anonymous) accounts with B.

1. U provides identification credentials to B(i.e. passport, etc.)

2. U runs PS.UKeyGen to obtain a bank-oriented master secret $\mathsf{ms_U}$ and a public/secret key pair $\{pk_{\mathsf{U}}^{\mathsf{B}}, sk_{\mathsf{U}}^{\mathsf{B}}\}$ connected to his $\mathsf{ms_U}$.

3. U runs PS.FormNym using the reg-parameters to generate a bank pseudonym $\mathsf{P}^{\mathsf{reg}}$ ([Camenisch and Lysyanskaya, 2001]), connected to $\mathsf{ms_U}$ in zero knowledge fashion.

4. U $\leftrightarrow$ B: execute EC.Withdraw procedure for U (see 4.1 or [Camenisch *et al.*, 2005] for more details) to withdraw a wallet $\mathsf{WAcc}_{\mathsf{U}}^{\mathsf{B}}$ of perm_α (ecoins). $\mathsf{WAcc}_{\mathsf{U}}^{\mathsf{B}}$ will later authorize U to open anonymous accounts in B. Consequently, the size of the wallet withdrawn depends on the maximum number of anonymous accounts U is eligible for.

5. U $\leftrightarrow$ B: execute PS.GrantCred procedure so that U obtains a registration credential $\mathsf{cred}_{\mathsf{U}}^{\mathsf{B}}$ for having registered in $\mathsf{D_{reg}}$, which is provably connected to $\mathsf{ms_U}$.

6. U stores in his database his secret key ($sk_{\mathsf{U}}^{\mathsf{B}}$), the information related to his pseudonym($\mathsf{pubP}^{\mathsf{reg}}$, $\mathsf{secP}^{\mathsf{reg}}$) and credentials ($\mathsf{pubcred}_{\mathsf{U}}^{\mathsf{B}}$, $\mathsf{seccred}_{\mathsf{U}}^{\mathsf{B}}$), while B stores only the public information.

Account Opening. To open an anonymous account, user U contacts B initially anonymously. Both, B and U make use of the α-parameter group. The following interactions take place:

1. U(anonymous) $\leftrightarrow$ B: run EC.Spend for U to spend an ecoin (S, π) (perm_α) from his $\mathsf{WAcc}_{\mathsf{U}}^{\mathsf{B}}$ wallet. If the ecoin used has been spent before, B runs the EC.Identify and EC.Trace procedures to recover U's identity($pk_{\mathsf{U}}^{\mathsf{B}}$) and activity ($sk_{\mathsf{U}}^{\mathsf{B}}$).

2. U: runs PS.FormNym, to generate a pseudonym P^{i} for managing his new account α^i. The pseudonym created has the form of

$$\mathsf{P} = a_\alpha^{\mathsf{ms_U}} b_\alpha^s,$$

where s is a U-B-generated value, whose final form is only known to U (see [Camenisch and Lysyanskaya, 2001]).

3. U(anonymous) $\leftrightarrow$ B: run $\mathsf{PS.VerifyCredOnNym}$, where U demonstrates ownership of $\mathsf{cred}_{\mathsf{U}}^{\mathsf{B}}$ and B verifies both, that $\mathsf{cred}_{\mathsf{U}}^{\mathsf{B}}$ and P^{i} are bound to the same ms_{U} (user) and that their owner has registered to the bank with a reg-pseudonym which is bound to the same ms_{U} as P^{i}.

4. U stores in his database the public/secret information related to his account-pseudonym ($\mathsf{pubP}^{\mathsf{i}}$, $\mathsf{secP}^{\mathsf{i}}$). B stores ($\mathsf{pubP}^{\mathsf{i}}$, S, π).

Tax Report Issue. This is a procedure taking place between the owner U of an account α^i, who participates through his pseudonym P^{i} and the bank B. It consists of three stages:

1. *Tax Report Number Acquisition.* The account pseudonym P^{i} collaborates with B in a $\mathsf{BS.Sign}$ procedure, for the former to obtain a (blind towards B) TRN related ticket trtick^i. U deposits *in person* to B the trtick^i to receive a tax-report-number TRN^i. B sends to TA the tuple (U, TRN^i) and stores it in its $\mathsf{D}_{\mathsf{reg}}$. Tax report numbers are chosen from a range $\mathsf{Range}_{\mathsf{T}}$, such that the sum of any two tax report numbers will result in a number out of the valid TRN-range ($\mathsf{Range}_{\mathsf{T}}$), i.e.,:

$$\forall \mathsf{TRN}^i, \mathsf{TRN}^j \in \mathsf{Range}_{\mathsf{T}} : \mathsf{TRN}^i + \mathsf{TRN}^j \notin \mathsf{Range}_{\mathsf{T}}.$$

2. *Tax Report Generation.* The following take place:

   (a) P^{i}: using $\mathsf{secP}^{\mathsf{i}}$ proves that he is the owner of P^{i}, by engaging in the ZKPoK:

$$PK\{(\beta, \gamma) : (\mathsf{P}^{\mathsf{i}})^2 = (a_\alpha^2)^\beta \cdot (b_\alpha^2)^\gamma\}.$$

   (b) $\mathsf{P}^{\mathsf{i}} \rightarrow \mathsf{B}$: $C = \mathsf{Com}(\mathsf{ms}_{\mathsf{U}}, \mathsf{TRN}^i, r^i) = k_\alpha^{\mathsf{ms}_{\mathsf{U}}} \cdot l_\alpha^{\mathsf{TRN}^i} \cdot m_\alpha^{r^i}$,
   where Com is a tax report related commitment scheme, ms_{U} U's master-secret, TRN^i, the single-use tax-report-number, which U acquired anonymously, and r^i is a U-generated randomness.

   (c) $\mathsf{P}^{\mathsf{i}} \leftrightarrow \mathsf{B}$: execute the following ZKPoK protocol for P^{i} to show in zero knowledge fashion that C was computed correctly, i.e., that the committed master secret matches the master secret used in the construction of P^{i} (ms_{U}) and that the exponent of l_α (TRN^i) is among the specified range:

$$PK\{(\gamma,\delta,\varepsilon,\eta) : (\mathsf{P^i})^2 = (a_\alpha^2)^\gamma (b_\alpha^2)^\delta \ \wedge \ C^2 = (k_\alpha^2)^\gamma (l_\alpha^2)^\eta (m_\alpha^2)^\varepsilon \ \wedge$$

$$\gamma \in \Gamma \ \wedge \ \delta \in \Delta \ \wedge \ \eta \in \mathsf{Range_T}\}.$$

(d) $\mathsf{P^i} \rightarrow \mathsf{B}$: a random r_x; if B has received r_x before, the procedure is repeated.

(e) B: decides x based on r_x. It then computes $h_\alpha^{\mathsf{tax}_i}$ and uses his RSA signature key to *sign* $T^{M,i} = h_\alpha^{\mathsf{tax}_i} \cdot C$ x times into $T^{\sigma,i}$. B provides U with an x-related piece of information encrypted to $\mathsf{SigInfo} = \mathsf{Enc_{TA}}(x)$, where $x \in Range_x$. $T^{\sigma,i}$ is then:

$$T^{\sigma,i} = h_\alpha^{d^x \mathsf{tax}_i} \cdot k_\alpha^{d^x \mathsf{ms_U}} \cdot l_\alpha^{d^x \mathsf{TRN}^i} \cdot m_\alpha^{d^x r^i} (mod n_\alpha).$$

(f) $\mathsf{B} \rightarrow \mathsf{U}$: $T^{\sigma,i}$, $\mathsf{SigInfo}$ and $\mathsf{SigInfo}$ re-randomization information $\mathsf{SITranform}$.

3. *Tax Report Transformation.* In this case, after having obtained his signed tax reports, U applies the transformation function F, so that — although provably valid — the modified tax reports are unlinkable to their initial form. In our scheme $F(M)$ is instantiated by adding an extra factor to M. In particular, U

(a) transforms both $T^{\sigma,i}$ and $T^{M,i}$ using $F(M,r) = M \cdot s_\alpha^{r_1} \cdot z_\alpha^{r_2}$, where M is the message to be transformed and $r = r_1 || r_2$ is a U-specified randomness. Thus, we get the following for the signed tax report and the corresponding message, respectively,
$TT^{\sigma,i} \leftarrow F(T^{\sigma,i}, r^{\sigma,i}) \leftarrow h_\alpha^{d^x \mathsf{tax}_i} \cdot k_\alpha^{d^x \mathsf{ms_U}} \cdot l_\alpha^{d^x \mathsf{TRN}^i} \cdot m_\alpha^{d^x r^i} \cdot s_\alpha^{r_1^{\sigma,i}} \cdot z_\alpha^{r_2^{\sigma,i}}$
$TT^{M,i} \leftarrow F(T^{M,i}, r^{M,i}) \leftarrow h_\alpha^{\mathsf{tax}_i} \cdot k_\alpha^{\mathsf{ms_U}} \cdot l_\alpha^{\mathsf{TRN}^i} \cdot m_\alpha^{r^i} \cdot s_\alpha^{r_1^{M,i}} \cdot z_\alpha^{r_2^{M,i}}$.

(b) re-randomizes the encryption of $\mathsf{SigInfo}$ according to $\mathsf{SITranform}$

Tax Report Deposit. Each user U(using a real identity) sends to the TA all the tax reports he has acquired, $(TT^{\sigma,1}, TT^{M,1}), \ldots, (TT^{\sigma,N}, TT^{M,N})$, where N is the number of U's accounts. U then proves that each one of these pairs were constructed in a correct way and that they correspond to *his* accounts. The tax report validation consists of two steps:

1. *Signature Validation*, where U shows that $(TT^{\sigma,i}, TT^{M,i})$, for all $i = 1 \ldots N$, correspond to transformations of bank-signatures:

(a) TA: decrypts SigInfo, reads x and raises all $TT^{\sigma,i}$s to B's signature verification key e, x times using $(mod n_\alpha)$:
$TT^{M',i} \leftarrow (TT^{\sigma,i})^{e^x} \leftarrow h_\alpha^{\mathsf{tax}_i} \cdot k_\alpha^{\mathsf{ms_U}} \cdot l_\alpha^{\mathsf{TRN}^i} \cdot m_\alpha^{r^i} \cdot s_\alpha^{e^x r_1^{\sigma,i}} \cdot z_\alpha^{e^x r_2^{\sigma,i}}$.

(b) U $\leftrightarrow$ TA: interact in the following ZKPoK protocol to prove that in each pair, $TT^{M,i}$ and $TT^{M',i}$ correspond to the same $TaxInfo$, i.e., that in both cases the exponents of $h_\alpha, k_\alpha, l_\alpha, m_\alpha$ are the same, or that $\frac{TT^{M,i}}{TT^{M',i}}$ is a factor of powers of s_α and z_α:

$$PK\{(\theta,\eta)\} : (\frac{TT^{M,i}}{TT^{M',i}})^2 = (s_\alpha^2)^\theta (z_\alpha^2)^\eta.$$

2. *Tax Report Ownership and non-Repetition Proof.* where U proves to the tax authority TA that each one of the tax reports he deposits had been created through his collaboration with B and that he has not deposit the same tax report twice. The latter is achieved through the one-time-use TRN s. For each one of $TT^{M,i}$s (or $TT^{M',i}$), U reveals the TRN^i to the TA, while he engages to a ZKPoK protocol for the TA to verify that the exponent of k_α in $TT^{M,i}$ (and thus,$TT^{M',i}$) matches the $\mathsf{ms_U}$ used in P^{B}, i.e.,

$$PK\{(\gamma,\delta,\tau,\varepsilon,\theta,\eta) : (\mathsf{P}^{\mathsf{reg}})^2 = (a_\alpha^2)^\gamma (b_\alpha^2)^\delta \wedge$$

$$\wedge \frac{TT^{M,i}}{l_\alpha^{\mathsf{TRN}^i}} = h_\alpha^\tau \cdot k_\alpha^\gamma \cdot m_\alpha^\varepsilon \cdot s_\alpha^\theta \cdot z_\alpha^\eta \ \wedge \gamma \in \Gamma \ \wedge \ \delta \in \Delta\}.$$

Total Tax Calculation. In this operation, TA confirms that U has deposited tax reports for all of his accounts and then uses them to extract the overall tax amount withheld by U's accounts. In particular, TA and U collaborate in an EC.Spend procedure for the latter to spend his unused ecoins from $\mathsf{WAcc}_{\mathsf{U}}^{\mathsf{B}}$ wallet. TA then estimates the exact number of U's accounts and computes the overall tax withheld as follows:

1. TA: computes the product of all $TT^{M,i}$ ($TT^{M,all}$), which because of the homomorphism of the commitment scheme used, equals to

$$\prod_{i=1,\ldots,N} TT^{M,i} = \prod_{i=1,\ldots,N} (h_\alpha^{\mathsf{tax}_i} \cdot k_\alpha^{\mathsf{ms_U}} \cdot l_\alpha^{trni} \cdot m_\alpha^{r^i} \cdot s_\alpha^{r_1^{M,i}} \cdot z_\alpha^{r_2^{M,i}}) =$$

$$h_\alpha^{TotalTax} \cdot k_\alpha^{N\mathsf{ms_U}} \cdot l_\alpha^{R_t} \cdot m_\alpha^{\sum_{i=1,\ldots,N} r^i} \cdot s_\alpha^{\sum_{i=1,\ldots,N} r_1^{M,i}} \cdot z_\alpha^{\sum_{i=1,\ldots,N} r_2^{M,i}}.$$

2. U reveals $TotalTax = \sum_{i=1...N} \mathsf{tax}_i$, which is the overall tax withheld.

3. U and TA collaborate in a ZKPoK protocol to prove that $\frac{TT^{M,all}}{h_\alpha^{TotalTax} \cdot l_\alpha^{R_t}}$ is correctly created and thus prove that $TotalTax$ is the required amount (note that TA knows R_t):

$$PK\{(\beta, \gamma, \delta, \zeta, \eta) \mathsf{P}^{\mathsf{reg}2} = (a_\alpha^2)^\gamma (b_\alpha^2)^\delta \wedge \frac{TT^{M,all}}{h_\alpha^{TotalTax} l_\alpha^{R_t}} = (k_\alpha^N)^\gamma m_\alpha^\varepsilon \cdot s_\alpha^\zeta \cdot z_\alpha^\eta \wedge \gamma \in \Gamma \wedge \delta \in \Delta\}.$$

9.1.4 Discussion

In this section, we will discuss how each one of our privacy and security requirements are satisfied. We also discuss deployability.

9.1.4.1 Security - Privacy

The following theorem states the correctness, privacy and security of our general scheme:

Theorem. if the underlying primitives (anonymous credential system, e-cash system, blind signatures, commitments and ZKPoK) are secure, then our scheme satisfies *correctness*, *account–account unlinkability*, *account–account-owner unlinkability*, *fairness in tax reporting*, *tax report non transferabiliy*, *tax report unforgeability*, and *accountability*.

We use prove this theorem with the following lemmas.

Lemma 1. If the underlying primitives (anonymous credential system, e-cash system, commitments and ZKPoK) are secure, then our scheme satisfies *Correctness*.

Proof. The first condition of correctness is satisfied directly through the *correctness* of the underlying schemes of ecash and anonymous credentials and according to which if U is honest neither EC.Spend procedure of perm_α nor PS.VerifyCredOnNym (which take place at the Account Open will output an error message. The correctness and verifiability of the RSA signature scheme, its homomorphism and the correctness of the used ZKPoK protocols used to confirm that U is the owner of all tax reports and guarantee that TaxReportDeposit will not output an error message.

Lemma 2. If the underlying primitives (anonymous credential system, ecash system, and ZKPoK) are secure, then our scheme satisfies *account-account unlinkability.*

Proof. Account-account unlinkability is maintained in the Account Open procedure through the *unlinkability* property of the ecash scheme used for perm_α and the *unlinkability of pseudonyms* property of the underlying anonymous credential system. Account-account unlinkability is also maintained through the tax reporting: Let α^1 and α^2 two accounts of U for which he obtains tax reports T^1, T^2 respectively. Then T^1 and T^2 are unlinkable one to the other because of the *hiding* property of the commitment scheme used to generate them and the *zero knowledge* property of the ZKPoK scheme used to prove their correct construction.

Lemma 3. If the underlying primitives (anonymous credential system, ecash system, blind signatures, commitment and ZKPoK, transformation function F, Paillier cryptosystem) are secure, then our scheme satisfies *account–account-owner unlinkability.*

Proof. Let α^i an anonymous account of user U managed by pseudonym P^i. Let T and TT be the tax report for α^i and its transformed version. Unlinkability of α^i and U at the AnonymousAccountRequest-AccountOpen procedures is achieved through the *anonymity* property of the ecash scheme realizing perm_αs and of and pseudonym system used for the generation of P^i as well as through the *blindness* of the blind signature scheme used for the acquisition of TRNs. T is unlinkable to U through the *hiding* property of the commitment scheme, which "hides" the ms_U committed in T and the security (*zero knowledge*) of the ZKPoK protocol used to validate the construction of T: no information is leaked neither TRN nor for ms_U contained in T. TT on the other hand, does not reveal anything regarding T or the account because of the *hiding* property of transformation function F used for its construction, the zero knowledge property of the ZKPoK protocol used at its validation and the re-randomization property of the Paillier cryptosystem used for blinding SigInfo.

Lemma 4. If the underlying primitives (anonymous credential system, digital signatures, commitment) are secure, then our scheme satisfies *Tax Report Unforgeability.*

Proof. Let that user U manages an account α_U through a pseudonym P and generates tax

report $T^{\sigma/M}$, which is later transformed to $TT^{\sigma/M}$, through $F()$. We need to prove that the tax report remains unforgeable at all stages. T is an RSA-signature-based function on a commitment on TRN, tax_i and $\mathsf{ms_U}$. To avoid B-signature forgeries exploiting RSA homomorphism, apply the signature scheme on T^M x number of times, while the RSA-signature verification key and x are kept secret to U. x is only revealed to TA only at the TT deposit procedure through $\mathsf{SigInfo}$. We assume that the granularity of different x-es is very small w.r.t. the total number of tax reports so that linkability attacks do not apply. U has no incentive to alter $\mathsf{SigInfo}$. To avoid such a forgery using the same tax report, we make use of TRN s, B-chosen numbers of a pre-specified range such that summations of two numbers in $\mathsf{Range_T}$ result in an invalid number. *Bindness* property of the commitment scheme used in T generation guarantees that as long as the RSA signature is unforgeable, U cannot dispute the $TaxInfo$ he has committed to in T^M.

Lemma 5. If the underlying primitives (anonymous credential system, digital signatures, commitment and ZKPoK) are secure, then our scheme satisfies *Tax Report non transferability.*

Proof. In our system users are highly motivated not to share their $\mathsf{ms_U}$. Thus, assuming that they are not doing so, Tax-Report non transferability is achieved through the need to prove knowledge of the $\mathsf{ms_U}$ at each step of the tax reporting. More specifically, account pseudonyms are required to show that their $\mathsf{ms_U}$ matches the one committed in T, which is then signed and -thus- cannot change (unforgeability of the signature scheme). The *proof of knowledge* property of the ZKPoK scheme used when depositing TT, guarantees that user depositing TT knows the corresponding $\mathsf{ms_U}$, which should match the $\mathsf{ms_U}$ used in all tax reports deposited by the same user, as well as his registration pseudonym.

Lemma 6. If the underlying primitives (anonymous credential system, ecash, digital signatures, commitment and ZKPoK) are secure, then our scheme satisfies *Fairness.*

Proof. Because of Tax Report Unforgeability and non-transferability, users cannot change the tax reported in each report or use other users' tax reports. Because of the $\mathsf{Identification\ of\ Violators}$ and $\mathsf{Violators'\ Traceability}$ property of ecash implementing perm_αs, users cannot

lie to the bank regarding the number of the accounts they have opened: if they try to prove they opened fewer accounts, some of the perm_αs in WAcc^B will be doublespent. At the same time, because of the TRNs, users cannot avoid a tax report, by depositing another one twice.

Lemma 7. If the underlying primitives (anonymous credential system, ecash, digital signatures, commitment and ZKPoK) are secure, then our scheme satisfies *Accountability.*

Proof. Because of the *Identification of Violators* and Violators' Traceability property of ecash implementing perm_αs, users who lie regarding the anonymous accounts they opened are identified. Because of the *proof of knowledge* property of the ZKPoK protocols, the *non-transferability of credentials* property of the underlying pseudonym system and the *non-transferability* property of tax reports, users trying to use other users' tax reports are detected.

Lemma 8. The transformation function F, defined on $D_M x \mathbb{Z}$, where:

- $F(M,r) = M \cdot s_\alpha^{r_1} \cdot z_\alpha^{r_2} (mod n_\alpha)$, $n_\alpha = p_\alpha \cdot q_\alpha$, p_α, q_α safe primes, $s_\alpha, z_\alpha \in QRn_\alpha$, $r = r_1 || r_2$ is a random number and M the message to be blinded;
- $D_M = \{x : \exists y,z,w,j : x = h_\alpha^y \cdot k_\alpha^z \cdot l_\alpha^w \cdot m_\alpha^j (mod n_\alpha)\}$, where $h_\alpha, k_\alpha, l_\alpha, m_\alpha \in QRn_\alpha$ are system parameters;

is computationally non-invertible and provides output indistinguishability w.r.t. M-inputs. More specifically, we claim that F supports:

- *Non Invertibility* Given an output f of $F()$ it is computationally impossible to compute $M \in D_M$ and r such that F(M,r) = f.
- *Input-Output Unlinkability* Given two messages M_1 and M_2 and an output f of $F()$ which corresponds to one of the messages, it is computationally hard to decide which message corresponds to f with a better probability than $1/2$.

Proof. Both properties derive directly from the discrete log assumption modulo a safe prime product and strong RSA assumption.

Lemma 9. The function Com used, defined on $(\mathbb{Z}x\mathbb{Z})x\mathbb{Z}$, where

$$\mathsf{Com}(x, y; r) = k_\alpha^x \cdot l_\alpha^y \cdot m_\alpha^r (mod n_\alpha),$$

$n_\alpha = p_\alpha \cdot q_\alpha$, p_α, q_α safe primes, $k_\alpha, l_\alpha, m_\alpha \in QR_{n_\alpha}$ is a commitment scheme on x, y with randomness r.

Proof. Function Com satisfies both properties *bindness* and *hiding* which derives from the discrete log assumption modulo a product of safe primes and factoring assumption.

9.1.4.2 Pailier Encryption

The Paillier cryptosystem is s a probabilistic asymmetric algorithm for public key cryptography and bases its security on the decisional composite residuosity assumption (see [Paillier and Pointcheval, 1999] for details). Assuming the system is meant for a user U to be able to receive messages confidentially, the operations supported are as in every cryptosystem the following:

- $(pk_\mathsf{U}, sk_\mathsf{U}) \leftarrow \mathsf{Pail.KeyGen}(1^k)$, where U generates his encryption key pair. In particular, U chooses two safe large prime numbers p and q, such that $gcd(p-1, q-1) = 2$, computes $n = pq$ and chooses $g \in \mathbb{Z}_{n^2}^*$, such that n divides the order of g. $pk_\mathsf{U} = (n, g), sk_\mathsf{U} = (p, q)$.

- $\langle\ \mathsf{C}/\perp\ \rangle \leftarrow \mathsf{Pail.Encrypt}(pk_\mathsf{U}, m)$, where anyone may use pk_U to generate ciphertext C on a mesage m: $\mathsf{C} = g^m \cdot r^n (mod n^2)$, where r is randomly chosen.

- $\langle\ m/\perp\ \rangle \leftarrow \mathsf{Pail.Decrypt}(sk_\mathsf{U}, \mathsf{C})$, where U uses his secret key to receive the plaintext.

It is apparent that a particular plaintext may have many ciphertexts, depending on r. We make use of this property in the encryption of x in two ways: (a) two users will not be able to distinguish whether they have the same x or not, and are thus unable to know whether they are able to exploit RSA homomorphism; (b) for re-randomization of SigInfo: users who know n can simply compute $\mathsf{C} \cdot (r')^n (mod n^2)$ and generate another ciphertext of x unlinkable to C. Thus in this case of encryption algorithm, SITranform is n.

Security Properties: *Semantic security against chosen plaintext attacks (IND-CPA)*, i.e. given pk, two messages m_1, m_2 and a ciphertext corresponding c to one of them, it is impossible to guess which of the messages corresponds to c with a better probability than $1/2$.

9.1.4.3 Deployability

Any real-world deployment of this scheme must be affordable, and must interface correctly with the existing worldwide banking system. Although protocols' implementation is not part of this work, we believe that these requirements are satisfied. In fact, in this section we show that it is easy to generate International Bank Account Numbers (IBANs) from our accounts and that these accounts can thus be used to send and receive payments. In the second paragraph, rough calculations suggest that by using modern hardware designs, the hardware necessary for timely tax reporting and verification is affordable by the organizations concerned, while the expense for banks is proportional to the number of customers they have.

Bank Account Number Generation. Although we have bank subaccounts which are identified by public keys, our system must be built on top of an existing bank account system. In this system, each sub account will correspond to a real world bank account that holds funds in the normal way, and will transact according to signed commands given by the account owner.

Because it is a real bank account, these sub accounts must also be associated with a bank account number. We cannot use a public key, which could be hundreds of decimal digits, as a bank account number. A bank account number is often handled by humans, it would not be practical for normal bank paperwork and operations.

For the sake of generality, we will assume that we will conform to the standard for International Bank Account Numbers (IBANs). An IBAN is prefixed by a 2-letter country code, followed by 2 check digits (for error correction), and up to thirty alphanumeric characters for domestic bank account numbers. Since the country code and check digits are immutable, our aim is to map a public key to the remaining 30 alphanumeric characters.

To address this, we will use an idea based on Cryptographically Generated Addresses [Aura, 2005]. This protocol is intended for mapping cryptographic public keys to the last 64 bits of an IPv6 address. This is a similar problem; a server is associated with a public key, but must be identified by an identifier in a much smaller space than that which would be necessary to keep public keys secure.

The thirty alphanumeric characters of an IBAN are equivalent to slightly more than 141 bits of information. CGA is based on the SHA-1 hash function, and we will adapt it as follows.

1. Choose a security parameter l (an integer from 0-7).
2. Choose a random modifier.
3. Concatenate the modifier with the public key and take the SHA-1 hash.
4. Repeat step 3, incrementing the modifier until the leftmost $16l$ bits of the result are zero.
5. Set the collision count to zero.
6. Concatenate the final modifier, the collision count, the country code, and the public key.
7. Take the SHA-1 hash of this value, and reinterpret the leftmost 141 bits as a 30-alphanumeric character bank account number.
8. Check for collisions, if there exists one increment the collision count and repeat from step 7.

In many cases, the IBAN will actually be structured: the first several digits will identify a bank, while the remainder identifies an account within the bank. The modifications to accommodate fewer digits are straightforward and will not be discussed further here.

Performance Issues. The protocols we describe are somewhat expensive, because they require a fair number of exponentiations. That said, we believe the cost is affordable.

Food	5
Car loan	2
Phone bils	2
Credit cards	6
Insurance	2
Cable TV	1
Rent/mortgage	1
Internet	1
Heat	1
Electricity	1
Water	1
Garbage	1
Lawn care	1
Total	25

Table 9.1: A conservative estimate of the maximum number of subaccounts, and hence checks, a typical individual will write each month.

Since opening accounts and subaccounts are uncommon operations, we ignore them. The real cost is in preparing and processing tax reports. The cost there is borne by all three parties: the individual, the bank, and the tax authority. We now analyze the cost to the latter two; the cost to the former is almost certainly minimal, since no one person will have that many accounts. Besides, the individual reaps the privacy benefits of the protocol, and hence is motivated to pay for it.

The total cost is determined by the number of subaccounts per person, and the number of individuals who use this protocol. We bound the latter by using the IRS's figure of about 150,000,000 individual tax return filers. Obviously, if not everyone is using these mechanisms, the total cost will be proportionally less.

The former is much more difficult to ascertain. Instead, we attempt to estimate the maximum number of subaccounts a typical individual would have, by assuming that a

separate subaccount is used for each monthly check written; see Table 9.1. To set an upper bound, we double that. Assuming there is a tax report once per quarter, we estimate that no more than 22,500,000,000 ($2.3 \cdot 10^{10}$) accounts exist. Reporting for each account requires one exponentiation and two zero knowledge proofs of knowledge. We estimate that the latter require twenty exponentiations apiece; each quarter, the tax authority thus must perform about $4.8 \cdot 10^{11}$ exponentiations.

A reasonably modern CPU can do about 150 2048-bit exponentiations/second. However, we can use dedicated exponentiation chips, which can do about 25,000 operations a second, or we can use the graphics processor (GPU) to do calculations. According to Szerwinski and Guïeysu [Szerwinski and Guñeysu, 2008], a GPU can do about 100 2048-bit exponentiations/second. Newer GPUs are considerably faster, and have considerably more parallelisms; in particular, the forthcoming Nvidia GeForce 300 Series will have 512 cores and should operate considerably faster. It seems reasonable to assume that we can reach speeds of 1000 exponentiations/second on a GPU-equipped computer. We estimate that the fully loaded cost of such a 1U server to be about $5,000, counting the computer itself, the rack, power, cooling, and real estate. A data center with 10,000 such machines would therefore cost about $50M, with the computers amortized over a three-year lifetime; the annual cost is thus about $17M. While not cheap, the cost is low compared to the current cost of running, say, the IRS, whose fiscal year 2008 budget was $11B. Such a complex could do $8.6 \cdot 10^{11}$ exponentiations/day; it would therefore take little more than half a day to process the tax reports, a value that is clearly acceptable. Our estimates could easily be off by a factor of 30 or more without changing the basic result: this protocol will not impose an undue processing burden on the tax authority.

It is rather more difficult to do the same calculation for banks, since figures on the number of depositors do not seem to be readily available. We can, nevertheless, perform some approximations. Preparing the tax reports requires a single zero knowledge proof of knowledge for each account; the cost per depositor (and hence the size of the data center) is thus roughly half of the tax authority's cost per filer. We approximate the number of customers who would desire such a service as the number of customers who use online banking today. According to their 2008 annual reports, Wells Fargo has 11,000,000 such

customers; Bank of America, another large bank, had about 30,000,000 online customers. For the latter, then, the cost of the data center is about 10% of the tax authority's cost, or $5,000,000, an amount easily affordable for such a large institution.

9.1.5 Related Work - Our Contribution - Future Directions

[Jakobsson and Yung, 1996; Low *et al.*, 1996] are cases of protocols providing conditionally anonymous payments from user issued bank accounts. However, their work is different from ours as there is either a third trusted party involved for anonymity revocation purposes, or they do not offer privacy against coalitions of banks.

Taxation has been addressed in the past in the stock market. In [Shouhuai Xu and Zhang, 2000], the authors propose a scheme addressing a similar problem to ours: anonymous and taxable stock market trading accounts. As in our system, users are using a generated anonymous credential from a public credential to validate anonymous stock-transaction. However, their system differs from our own in two major ways. First, they only allow for each user to own one anonymous account, because of the extra complications to tax reporting the multiple accounts would cause. Addressing these complications is one of our major contributions. Secondly, they do not aim to prevent the Tax Authority from learning which accounts the reports are coming from. Thus if the TA were to collaborate with the Stock Exchange Center, they could re-link the users with their anonymous accounts. Preventing this is another contribution of our system.

9.2 Privacy-Preserving Credit Cards

Aiming to deal with privacy issues raised in online transactions — online payments in particular — in this section we introduce a privacy-preserving credit card mechanism which can be applied in current credit card systems. In particular, we present a realizable protocol to support "credit card"-based online and offline transactions, in which banks, unless authorized by the cardholder, do not acquire any knowledge of their customers' transactions. At the same time, cardholders are provided with detailed reports regarding their purchases and may participate on any type of merchant credit card offers. As we will see in section 9.2.4,

such a privacy-preseving card-payment technique, operating in a credit fashion while offering the same functionalities as existing credit systems, has not been suggested in the past. For the purposes of our system, we made use of a combination of two types of compact ecash [Camenisch *et al.*, 2005] scheme for payments, and a combination of blind [Camenisch and Lysyanskaya, 2002a] and plain digital signatures for the rest of our system's operations.

In the following section we will briefly present our system's architecture and requirements, while we will elaborate on how our "real world" principle is adopted in our threat model. In section 9.2.2 we present our protocols, whose privacy and security we discuss in section 9.2.3. Finally, we comment on existing work on similar topic and emphasize on what ways they differ from our model.

9.2.1 System Architecture

A typical credit card mechanism consists of cardholders (consumers), merchants (sellers), Card Issuing Banks, Acquiring Banks and Credit Card Associations.

When eligible to receive a credit card, a consumer applies to a *Card Issuing Bank* she maintains an account with. The *Card Issuing Bank* bills the *cardholders* for payment and bears the risk of fraudulent use of the card. On the other hand, *Merchants*, who are eligible to receive credit card payments, are in agreement with an *Acquiring Bank* authorized to receive payments on their behalf. Banks are organized in *Credit Card Associations* (such as Visa or Master Card) that set the transaction rules between *Card Issuing* and *Acquiring Banks*.

The interactions between the entities can be seen in fig. 9.1. For convenience, in the following sections, we will refer to a merchant as 'he', to a client as 'she' and to any type of Bank as 'it'. In addition, we will use the following acronyms: CIB for Card Issuing Bank, AB for Acquiring Bank, ACCA for our Anonymous Credit Card Association and ACC for Anonymous Credit Card.

In what follows we present our security and privacy model. Because of its complicated nature, we restrict this presentation into an informal definition of the security and privacy properties and the corresponding threat model. For more details of our security model, see Appendix A.

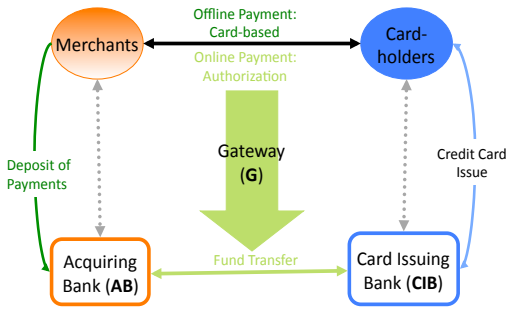

Figure 9.1: Architecture of a typical credit card system.

9.2.1.1 Threat Model.

As we aim to create a realizable system, we assume that our adversary has all the powers and motives real Banks/merchants/cardholders or groups of them would have. In addition, we assume that all parties have as main objective to increase their profit and would not act against it. More specifically:

- *Banks are "honest but curious"*. They are trusted to do their functional operations correctly, but they may collude with each other or with merchants and combine the information they possess to track their customers' activities.

- *Merchants are not considered honest.* Mostly interested in receiving their payment, merchants they may try to "deceive" a cardholder into making her pay more. For advertising purposes, merchants may be motivated — if cost effective — to collaborate with banks to profile their customers. In offline transactions the merchant knows the customer's face. However, any attempt to identify a customer manually (i.e. by comparing pictures online) is not cost effective and is thus highly unlikely.

- *Cardholders may try to "cheat"*, i.e. not to pay a merchant, to forge an ACC, or

perhaps to frame another customer by accusing her of having cheated.

9.2.1.2 Requirements.

Privacy, *security* and a notion of *deployability* are the requirements here. For simplicity (and since we assumed collaboration between banks), we will refer to all of them as a united organization, a general Bank.

As in the previous section, *Privacy* can be analyzed to the following two properties:

1. *Customer Anonymity w.r.t bank and the merchant.* Given a simple (online) cardholder-merchant transaction, no unauthorized third party — including the bank — should be able to gain any information regarding that particular transaction or link it to a particular cardholder even with merchant's collaboration.

2. *Transaction Unlinkability.* Linking different ACC-based transactions as having been done by the same cardholder should be hard without the cardholder's consent.

Security consists of

1. *Strong Cardholder Authentication*, as we require that all cardholders are strongly — but privately — authenticated when requesting a credit card related service, i.e., payment, expense report, etc.

2. *Accountability* which requires that misbehaving parties are detected. More specifically, we require that individuals who try to use their card for making purchases over their credit limit are identified, the ones who are trying to use other people's cards are detected.

3. *Credit Card Unforgeability*, i.e., no party should be able to create a credit card without the bank's collaboration, and make valid purchases with it.

4. *Credit Card non-Transferability*, i.e., no individual should be possible to use another person's card without obtaining critical secret information which the owner is not motivated to share.

Deployability addresses the applicability of our scheme. In particular, we require that our protocols

1. are usable with the current credit card systems' architecture as described in the previous section.

2. support many current credit card services, while maintaining privacy:

   (a) *Expense Report Service:* cardholders should be able to track their transactions.

   (b) *Error Correction Service:* cardholders should be able to provide an undeniable proof of any mischarge

   (c) *Card Loss Recovery:* cardholders should be able to invalidate and substitute the card when the latter is lost.

   (d) *Special Payment-rate Offers* merchants usually offer in collaboration with particular credit card associations.

9.2.2 Anonymous Credit Card System

Our credit mechanism can be viewed as a long term loan. The cardholder is credited the amount she borrows to transact, while credit limit $\mathsf{L}_{\mathsf{credit}}$ is the highest price that loan can get. The amount borrowed is outpaid in a predefined interest rate as in current credit card systems. To avoid charges for a bigger amount than the one she has spent, the cardholder is required — at the end of each predefined time period, which is usually a month — to provide undeniable proof of the amount of money she has spent.

Identities-Master Secrets. Entities in our system (cardholders, banks, merchants) are identified by the signature keys they issue after entering the system. However, the transaction activity of each cardholder is entirely enabled by a master secret, ms , which is part of her secret signature-related information. As impossible to be certified without a bank's collaboration, proof of possession of a valid ms denotes a valid cardholder and any attempt to cheat would result in recovery of the identity and ms of the misbehaving entity.

Bank Data Management. CIBs maintain a large database consisting of their customers' account information. In particular, CIBs manage D_{debit} for customers' debit accounts, D_{credit}

for the credit accounts, D_{anon} for the temporary anonymous accounts[1] used only in online ACC transactions and D_{hist} which is used as a log of D_{credit} and D_{anon}. ABs — which may/may not be CIBs — are connected to merchants' debit accounts.

System Operation. ACCs are issued with the collaboration of a CIB and the intended cardholder, who has already opened an account with the former. ACC's functionality is based on two types of ecash wallets, which are withdrawn at the ACC issuing procedure: the payment wallet W_p and the identity one W_{id}. The two wallets have the same number of ecoins , which is in proportion to the credit limit of the cardholder. However, they differ to their traceability mechanism in case of double-spending. If double-spent, W_p only reveals the identity of the double-spender. On the contrary, if part of W_{id} is double-spent, ms of the double-spender and all the ecoins withdrawn by the latter are revealed. W_p's traceability attributes are used in the loss recovery protocol, while W_{id}'s are used in identifying malicious parties.

Payment procedure basically consists of a merchant authentication phase followed by two ecash spending procedures. In offline purchases, the cardholder uses merchant's machine to spend the same number of ecoins from both wallets, W_p and W_{id}. Spent ecoins ' value corresponds to the price of the product. Transaction details are signed by merchant and stored (as receipt) in the card. Merchant is paid when he deposits the ecoins he has obtained through his customers to his AB. Apparently, any attempt on cardholder's side to cheat, i.e. by using two copies of the same card will reveal — because of the W_{id} double-spending defense mechanism — the cheater's ms . To enable online purchases, customer sets up an anonymous account with her CIB. In particular, she makes use of an ATM machine where she spends the amount of ecash she would like to use for the online purchases from both wallets. Both parties agree on secret information which would enable a cardholder to authorize the intended merchant's AB to subtract safely money from that anonymous account. Secret information is renewed so that replay attacks can not succeed. At the end of each month, the cardholder deposits in person the unspent part of her wallets to her CIB,

[1]Depending on the bank's setting this database may or may not be the same as the D_α used in section 9.1 for anonymous accounts.

so that the former is charged accordingly.

Each cardholder is encouraged each fixed time interval to back up the content of her ACC. In case of loss, the backed up ACC content is simply copied to another card. Faking ACC's loss and making use of both ACCs (the new and the old) will reveal customer's identity, who is immediately charged[2]. In all cases, the content of the card is encrypted so that only its legal owner can read its content. Expense Report Service is achieved through a decryption of the transaction-part of the card which can take place locally, at client's machine.

ACC promotion offers are realized through coupons obtained by each merchant from the ACCA. Coupons are stored in the ACC at the end of the transaction and deposited by the cardholder to her CIB, for the former to obtain better interest rates. To preserve cardholder anonymity coupons' deposit procedure is done in two phases.

In what follows we will use $[\mathsf{B}]\mathsf{Sig}_{\mathsf{C}}(\mathsf{Msg})$ ($[\mathsf{B}]\mathsf{Sig}^{\mathsf{H}}_{\mathsf{C}}(\mathsf{Msg})$) to denote the [blind] signature of C on Msg ($\mathsf{H}(\mathsf{Msg})$) and $\{\mathsf{Msg}\}_K$ to denote encryption of Msg under key K. For efficiency, every asymmetric encryption is inducted to a symmetric one: $\{\mathsf{Msg}\}_{PK}$ denotes $\{K\}_{PK}||\{\mathsf{Msg}\}_K$ for a random K.

The details of our schemes are demonstrated bellow.

9.2.2.1 Setup.

Every Bank B participating in the ADS generates a signature key pair: $(pk^s_{\mathsf{B}}, sk^s_{\mathsf{B}})$, which identifies it. To facilitate the withdrawal of the ecash wallets mentioned before, B makes the appropriate setup to support the two compact ecash schemes in [Camenisch *et al.*, 2005] and runs $\mathsf{EC.BKeyGen}$ to generate $(pk^w_{\mathsf{B}}, sk^w_{\mathsf{B}})$. For the purposes of authentication, efficiency and accountability in online transactions, each CIB publishes an array of hashes $\underline{\mathsf{Hash}}_{ot}$ and a gateway communication hash H_{GB}.

Every **merchant** M collaborates with his AB, AB, to issue a signature key pair $(pk^s_{\mathsf{M}}, sk^s_{\mathsf{M}})$ with the corresponding certificate $\mathsf{Cred}_{\mathsf{M}} = \mathsf{Sig}_{\mathsf{B}}(pk^s_{\mathsf{M}})$.

[2] Anonymity revocation in this case is limited: only the identity of the person having double-spent is revealed and not her master secret.

On the other hand, each **customer** C who opens an account with a CIB CIB collaborates with the latter in $\mathsf{EC.UKeyGen}$ procedure of both compact ecash schemes [Camenisch *et al.*, 2005]), mentioned in subsection 4, to issue two signature key pairs:

$$(pk^p_{\mathsf{C}}, sk^p_{\mathsf{C}}), (pk^{id}_{\mathsf{C}}, sk^{id}_{\mathsf{C}}).$$

Although each one of the pairs above individually identifies C, for convenience, we will refer to both pairs as $(pk^s_{\mathsf{C}}, sk^s_{\mathsf{C}})$.

The **ACCA** chooses and publishes the transaction related hashes H_{ot}, H_t and H_r.

9.2.2.2 ACC Issue.

Let that customer C is eligible for a credit limit $\mathsf{L_{credit}}$ with a CIB CIB. CIB and C collaborate in $\mathsf{EC.Withdraw_p}(sk^p_{\mathsf{C}}, \mathsf{L_{credit}})$ and $\mathsf{EC.Withdraw_{id}}(sk^{id}_{\mathsf{C}}, \mathsf{L_{credit}})$ for the latter to withdraw the payment W_p and identity W_{id} wallets. In both withdrawal procedures, C provides a sk^s_{C} related password, $pass_{pin}$. In the issued ACC is also stored public information regarding the Banks participating in ACCA (params).

Apart from $pass_{pin}$, the cardholder chooses a set of passwords: a backup $pass_e$ password — from which her backup encryption key pair $(pk^e_{\mathsf{C}}, sk^e_{\mathsf{C}})$ — is derived and $pass^t_e$, $pass^w_e$, $pass^c_e$ — which correspond to three encryption key-pairs

$$(pk^{e_t}_{\mathsf{C}}, sk^{e_t}_{\mathsf{C}}), (pk^{e_w}_{\mathsf{C}}, sk^{e_w}_{\mathsf{C}}) \text{ and } (pk^{e_c}_{\mathsf{C}}, sk^{e_c}_{\mathsf{C}}),$$

that serve for encryption of transaction, wallet and coupon part of the card as we will describe later on. Each cardholder also agrees on two hashes with Bank H_K and H_{CB}.

It is noticeable that no C*-related identification information is included in the ACC, other than the information one can infer from the credit limit of a cardholder (which equals the value in the wallets). On the other hand, because of ecash properties, only the individual who issued the ACC, i.e. withdrew the wallets, is able to use it, i.e. spend part of the wallets consisting it.*

9.2.2.3 Payment

Offline Payment. It takes place between the cardholder C, who participates through her ACC (anonymously), and a merchant M.

1. M provides $\mathsf{Cred}_{\mathsf{M}}$ to C, who checks its validity using params.
2. M signs a hash of the transaction details, $T_{det} = \{date,\ time,\ product,\ price\}$, and provides C with
$$\mathsf{Cred}_{\mathsf{M}},\ T_{det},\ \mathsf{Sig}_{\mathsf{M}}^{H_t}(T_{det}),$$
where *product* is all the product related info stored in its bar code.
3. C verifies the *product* information and inserts her $pass_e^w$ to have her W_p and W_{id} wallets decrypted,
4. C enters her $pass_{pin}$ to run $\mathsf{EC.Spend_p}(sk_{\mathsf{C}}^p,\ price)$ - $\mathsf{EC.Spend_{id}}(sk_{\mathsf{C}}^{id},\ price)$ and spends *price* value from both wallets. Let W_p' and W_{id}' be the remaining wallets.
5. M calculates $\mathsf{Rec_T} = \mathsf{Sig}_{\mathsf{M}}^{H_r}(T_{det} - finished)$ and sends it to C, while he provides C with a printed transaction record.
6. C encrypts $\mathsf{Rec_T}$ and $W_p - W_{id}$ using her $pass_e^t$, $pass_e^w$ respectively into:
$$E_{T_{det}} = \{T_{det}||Rec_T\}_{pk_{\mathsf{C}}^{e_t}} \text{ and } E_{W_{p,id}} = \{W_p'||W_{id}'\}_{pk_{\mathsf{C}}^{e_w}}.$$

To receive his payment, M simply deposits to his AB, AB, the ecoins he has received from his customers. AB contacts each customer's CIB[3], CIB, and collaborates with the latter to run $\mathsf{EC.Deposit_p}$ and $\mathsf{EC.Deposit_{id}}$ (see section 4). Depending on whether double-spending has occurred, M may receive credit for each pair of payment-identity ecoin deposited. If no double-spending is detected, CIB and AB make the required transfer to M's bank account. If there a double-spending is detected, CIB runs $\mathsf{EC.Identify_{id}}$ on the double-spent identity ecoins to reveal the double-spenders' pk_{C}^{id}. In addition, CIB may run $\mathsf{EC.Trace_{id}}$ to trace all the identity ecoins pk_{C}^{id} double-spender has withdrawn.

[3]CIBs may be identified by the form of the ecash deposited.

Online Payment. It is performed in two stages: initially the cardholder C collaborates with her CIB, CIB, to set up one or more Anonymous Accounts (*Anonymous Account Setup*), which C addresses to make her online purchases (*Transaction Payment*).

Anonymous Account Setup. The two parties involved, C and CIB collaborate through her ACC and an ATM respectively. Let that C has decided to open an anonymous account of M_{ot} value for her online transactions.

1. C interacts with CIB into

$$\mathsf{EC.Spend_p}(sk^p_{\mathsf{C}}, M_{ot}) \text{ and } \mathsf{EC.Spend_{id}}(sk^{id}_{\mathsf{C}}, M_{ot})$$

   to spend M_{ot} from her W_p, W_{id} wallets respectively.

2. C chooses m, h_{ot}, a random R_{ot} number and acknowledges CIB.

3. Chashes R_{ot} m times using hash $H_{ot} = \underline{\mathsf{Hash}}_{ot}[h_{ot}]$: $(H^{(i)}_{ot}(R_{ot}), i = 1 \ldots, m)$. Let

$$A^x_{\mathsf{C}} \;=\; H^{(x)}_{ot}(R_{anon}).$$

   A^m_{C} will be the anonymous account number for the first transaction and in general the $A^{m-(i-1)}_{\mathsf{C}}$ will be the account number for the i^{th} online transaction. C sends $A^m_{\mathsf{C}},\ m, h_{ot}$ to CIB.

4. C establishes a pseudonym P^C_α, i.e. the public part of a key-pair (pk_P, sk_P), for which she gets the corresponding certificate:

$$Cert_{P^{\mathsf{C}}_\alpha} = \mathsf{Sig}_{\mathsf{CIB}_{ot}}(pk^{\mathsf{C}}_\alpha),$$

   where CIB_{ot} is the online transaction section of CIB.

5. CIB stores in D_{anon}

$$\alpha_{\mathsf{C}}(m) = \{A^m_{\mathsf{C}},\ hi_{ot},\ M^m_{ot},\ m,\ \mathsf{P}^{\mathsf{C}}_\alpha\},$$

   where for consistency, we use $M^m_{ot} \;=\; M_{ot}$ to express the balance of the A^m_{C} account.

6. CIB confirms account's validity with

$$\mathsf{Rec}_{\alpha_{\mathsf{C}}(m)} = \mathsf{Sig}_{\mathsf{CIB}_{ot}}(H^r(\alpha_{\mathsf{C}}(m)),\ Cert_{\mathsf{P}^{\mathsf{C}}_\alpha},\ time),$$

which serves as a proof of ownership of $\alpha_{\mathsf{C}}(m)$ and of the corresponding pseudonym $\mathsf{P}_{\alpha}^{\mathsf{C}}$.

7. C using $pass_e^t$ encrypts $\alpha_{\mathsf{C}}(m)$ and $\mathsf{Rec}_{\alpha_{\mathsf{C}}(m)}$ and stores them in her ACC.

Transaction Payment. Let that the cardholder C wants to purchase a product of value V_p from a merchant M. We assume that C has logged in to M's website anonymously.

1. C provides M's website — or the gateway G behind it — with the current number of her anonymous account (A_{C}^x), while she uses A_{C}^{x-1} to calculate the following:

$$\mathsf{TInfo}_{\mathsf{CIB}} = \{\{\mathsf{Cred}_{\mathsf{M}},\ T_{det}, A_{\mathsf{C}}^{x-1}\}_{K_\alpha},\ A_{\mathsf{C}}^x\}_{pk_{\mathsf{CIB}}^e},$$

where $K_\alpha = H_K(A_{\mathsf{C}}^x)$ a key derived from A_{C}^x and $H_K = \underline{\mathsf{Hash}}_{ot}[h_k]$ used for efficiency purposes.

2. G sends $\mathsf{TInfo}_{\mathsf{CIB}}$ to C's CIB(CIB) as follows:

$$\mathsf{Sig}_{\mathsf{G}}(H_{GB}(T_{det}, \mathsf{Cred}_{\mathsf{M}}), time).$$

3. CIB decrypts $\mathsf{TInfo}_{\mathsf{CIB}}$ and checks whether it matches the information provided by G. CIB verifies that A_{C}^x is an active anonymous account number in its D_{anon} and that the current account balance is enough for the payment included in T_{det}. If there is an error, CIB sends G a transaction rejection message:

$$\mathsf{Rej}_{\mathsf{CIB}} = \mathsf{Sig}_{\mathsf{CIB}_{ot}}(\mathsf{Cred}_{\mathsf{M}},\ T_{det},\ A_{\mathsf{C}}^x,\ reject),$$

which is then forwarded to C. To avoid any replay attacks, CIB updates the $\alpha_{\mathsf{C}}(x)$ entry of D_{anon} to

$$\alpha_{\mathsf{C}}(x-1) = \{A_{\mathsf{C}}^x,\ h_{ot},\ h_k,\ M_{ot}^{x-1},\ x-1,\ Cert_{P_\alpha^{\mathsf{C}}}\},$$

where $M_{ot}^{x-1} = M_{ot}^x$ since no purchase took place.

4. CIB sends to G a signed endorsement on $\mathsf{Cred}_{\mathsf{M}},\ T_{det}$ with check of value V_p addressed to M's AB, AB:

$$\mathsf{Paym}_{\mathsf{CIB}\to\mathsf{M}} = \mathsf{Sig}_{\mathsf{CIB}_{ot}}(H_{MT}(\mathsf{Cred}_{\mathsf{M}},\ T_{det}),\ A_{\mathsf{C}}^x,\ \mathsf{AB},\ time),\ Cred_M,\ T_{det}$$

where H_{MT} is a hash agreed between Banks, used to reduce the signed text. CIB also updates ($\alpha_{\mathsf{C}}(x)$ entry in its D_{anon} to

$$\alpha_{\mathsf{C}}(x-1) = \{(A_{\mathsf{C}}^{x-1},\ h_{ot},\ h_k,\ M_{ot}^{x-1},\ x-1\}$$

where $M_{ot}^{x-1} = M_{ot}^{x} - V_p$.

5. G sends to CIB a confirmation of the valid transaction

$$\mathsf{Rec_T} = \mathsf{Sig}_{\mathsf{G}}^{H_r}(\mathsf{Cred_M}, T_{det} - finished)$$

and forwards $\mathsf{Paym_{CIB \to M}}$ message to M's AB for M's account to be credited accordingly. As G is paid in proportion to V_p, it has no motivation not to submit $\mathsf{Paym_{CIB \to M}}$ to AB.

6. CIB stores $\mathsf{Rec_T}$, T_{det} at its D_{hist} database.

We need to note that here A_{C}^{x-1} is used for authentication purposes, since C who generated R_{ot} is the only one, who knows the pre-image of A_{C}^{x} on H_{ot}. M sends a receipt of the purchase to one of C's email addresses as well as to CIB, which stores everything under the anonymous account entry. When closing the entry in D_{anon}, C, through her ACC, demonstrates knowledge of $\mathsf{P}_{\alpha}^{\mathsf{C}}$, m and R_{ot} to CIB and the latter updates the content of C's ACC accordingly (with $\mathsf{Rec_T}$, T_{det} and any additional wallets).

9.2.2.4 ACC BackUp.

It takes place between a cardholder C, who participates in person, i.e. after having identified herself to her CIB, and her CIB, CIB.

1. C generates a random number N_b, creates the $K = H_K(N_b,\ pass_{pin})$ and sends

$$BackUp = \{N_b\}_{pk_{\mathsf{C}}^e}, \{ACCcontent||date-time\}_K$$

to CIB. *K is used for efficiency purposes, since symmetric encryption is considered to be much faster. We can see that only the valid owner of the card knows how to create K given N_b.*

2. Both, C and CIB, hash and sign the $BackUp$ into

$$BackUp_x = \mathsf{Sig}_x^{H_{CB}}(BackUp),\ x = \{\mathsf{C}, \mathsf{B}\},$$

where $ACContent$ is the content of the anonymous credit card and $date-time$ is the timestamp of the backup.

3. CIB updates her D_{hist}.

9.2.2.5 Loss Recovery.

It takes place between a cardholder C, who participates as an identity and her CIB, CIB.

1. C declares the loss of her ACC and is provided with the most recent BackUp of her ACC, $BackUp$.

2. C verifies that $BackUp$ matches the most recent $BackUp_B$ of her and decrypts it.

3. C and CIB collaborate in $\mathsf{EC.Spend_p}(sk_{\mathsf{C}}^p, |W_p'|)$ to spend the remaining payment wallet of the $BackUp$ (W_p').

4. CIB credits C's credit account for the amount spent till the $BackUp$ had been taken ($\mathsf{L_{credit}}$ - $|W_p'|$) and waits till the merchants' deposit time passes. For each double-spent digital coin detected, CIB runs $\mathsf{EC.Identify_p(S)}$ to confirm its owner and credits C's credit account accordingly. In this way, we avoid overall expose of C if the last $BackUp$ kept is not up-to-date with the C's most recent transactions.

5. CIB infers C's overall spending amount and collaborates with her in $\mathsf{EC.Withdraw_p}$ and $\mathsf{EC.Withdraw_{id}}$ procedures for the latter to issue new payment and identity wallets W_p and W_{id}.

It is apparent that in this case there is a small breach in anonymity provided in our system: the Bank will be able to see the merchants the customer interacted with to spend the double-spent part of her payment wallet. In the following section, we will elaborate on this privacy breach.

9.2.2.6 Monthly Payment Calculation

At the end of every month, each cardholder C proves to her CIB, CIB, the amount of money she has spent throughout that month. To calculate C's monthly payment, CIB applies the formula used in current Credit Card Systems on C's overall credits.

1. C enters her $pass_e^w$ to decrypt the remaining of her W_{id} wallet (W'_{id}) and interacts with CIB into $\mathsf{EC.Spend_{id}}(sk_{\mathsf{C}}^s, |W'_{id}|)$ to spend it entirely.
2. CIB updates C's entry in D_{credit} with the amount of money spent by C: $\mathsf{L_{credit}}$ - $|W'_{id}|$.
3. CIB bills C based on the latter's D_{credit} entry.
4. If C is still eligible for an ACC, she interacts with CIB for issuing a new W_{id} and additional W_p according to C's new credit limit.

It is obvious that any attempt on C*'s part to lie for the remaining* W_{id} *wallet, i.e. by presenting a former version of the card, a part of* W_{id} *would be double-spent (to* CIB *and to a merchant) and* $EC.Trace_{id}$ *would reveal* sk_{C}^s.

9.2.2.7 Expense Report-Error Correction.

Expense Report. The cardholder C enters her $pass_e^t$ to decrypt the transaction related part of her ACC to obtain the detailed chain of transactions. C may request for an expense report of her active anonymous accounts by providing her CIB, CIB, anonymously with the following:

$$\{expense - report, time, A_{\mathsf{C}}^{x-1}\}_{K_\alpha},\ A_{\mathsf{C}}^x,$$

where A_{C}^x is the current anonymous account number and A_{C}^{x-1} is used for authentication purposes. CIB provides the report of A_{C}^x's transaction activity and — for security purposes — updates D_{anon} by replacing $\alpha_{\mathsf{C}}(x)$ with $\alpha_{\mathsf{C}}(x-1)$.

Error Correction. It takes place between a cardholder C, who participates as an identity, and the customer service section of ACCA or of the merchant M involved. There are

two error cases we examine here: (a) C detects a mischarge at her expense report, in which case she contacts the ACCA in person, and (b) C requests to cancel a transaction with a merchant M, in which case she contacts M in person. In both cases C uses $\mathsf{Rec_T}$ as a proof of purchase.

For now we will refer to case (a). C contacts the ACCA and the ACCA contacts M. M can either accept or reject the refund-request. If he rejects, his customer service department, (M-CS), sends to the ACCA:

$$\mathsf{Rej_M} = \mathsf{Sig_{M-CS}}(Merchant,\ T_{det},\ refund - reject,\ time).$$

ACCA, depending on its policy, may provide C with the refund:

$$\mathsf{RefCoup_{ACCA \to C}} = \mathsf{Sig_{ACCA-ref}}(\mathsf{C},\ \mathsf{Rej_M},\ T_{det},\ price,\ time),$$

where ACCA-ref is the refund section of ACCA. C interacts with her CIB, CIB, to deposit $\mathsf{RefCoup_{ACCA \to C}}$ and to withdraw additional payment and identity wallets of equivalent value. *It is noticeable that in this case, we do not care whether* C*'s identity is revealed, since no purchase was actually done.* If M accepts the return/mischarge:

1. M provides the ACCA with signed endorsements of the amount to be removed from his account and added to C's card/account. Signatures of this type may be special type of refund signatures issued by M's CS:

   $$\mathsf{Ref_M} = \mathsf{Sig_{M-CS}}(Merchant,\ T_{det},\ refund - accept,\ refund,\ time),$$

   where $Merchant$ has all the merchant related account and certificate information.

2. The ACCA sends $\mathsf{Ref_M}$ to M's AB, AB, for validation.

3. AB sends back to ACCA a confirmation:

   $$\mathsf{RefConf_{AB}} = \mathsf{Sig_{AB}}^{H_{\mathsf{AB}-ref}}(Ref_M - accept),$$

   where $H_{\mathsf{AB}-ref}$ is a refund-specific hash of AB.

4. For C to collect the payment, ACCA issues a digital check to C:

   $$\mathsf{RefCoup_{ACCA \to C}} = \mathsf{Sig_{ACCA-ref}}(\mathsf{RefConf_{AB}},\ \mathsf{Ref_M},\ T_{det},\ refund,\ time, \mathsf{C}).$$

5. C then deposits $\mathsf{RefCoup}_{\mathsf{ACCA}\rightarrow\mathsf{C}}$ to CIB, which contacts AB to make the appropriate transfers. C interacts with CIB to issue $W_{id} - W_p$ wallets of $refund$ value.

In case (b), as mentioned before, C contacts M directly. If M accepts the return of the product, it provides C with $\mathsf{Ref}_{\mathsf{M}}$, which C deposits to CIB. CIB makes the appropriate interbank communications to verify Ref_M's validity and updates C's account accordingly.

Timestamps are used for replay attacks to be avoided. It is noticeable that all messages exchanged here are signed/timestamped. In this way replay and intersection attacks are avoided. Also, we make use of different digital signature key-pairs than in the rest of the ACC protocols. This is done to avoid any type of reflection attacks involving different protocols which use the same signature key-pairs. Another important point in the expense report protocol is that privacy is not a concern and thus, no encryption is used.

9.2.2.8 ACC Promotion Offers.

This is the case where the Anonymous Credit Card Association (ACCA), some CIBs and merchants have made an agreement, so that the CIBs provide better payment interest rates to cardholders when they make many purchases from the participating merchants, i.e., the cardholder may be eligible of paying out an amount V_p in N_{M} parts without any interests applied on it. To support this mechanism in our system we introduce coupons instantiated through the use of blind group signatures.

The Anonymous Credit Card Association (ACCA), as the ACCA manager, makes the appropriate setup for the blind group signature scheme [Lysyanskaya and Ramzan, 1998], which will be used to instantiate the group of the promotion participating merchants. In particular ACCA runs BGS.Setup mentioned in subsection 4 to generate $pk^{bg}_{\mathsf{M}}, sk^{bg}_{\mathsf{M}}$, i.e., the public and secret administration information of the merchants-group. Merchants, who participate in the promotion offers, interact with the ACCA in BGS.Join to obtain a membership blind group signature secret information sk^{bg}_{M}. In addition, participating CIBs

issue a plain blind signature key pair [Okamoto, 2006; Camenisch and Lysyanskaya, 2002a]: $(pk^b_{\mathsf{B}}, sk^b_{\mathsf{B}})$.

Assuming coupons of value v_p^M and after having interacted with a merchant M in a transaction of value V_p, a cardholder C contributes randomness $r_1, \ldots, r_N$ to obtain from M $N = \lfloor \frac{V_p}{v_p^M} \rfloor$, blind signatures, the *credit coupons*:

$$\mathsf{GBSig}Mr_1, \ldots, \mathsf{GBSig}_M(r_N).$$

Credit coupons are separately encrypted and stored in the ACC with $pass_c$ and should be deposited within a month after the transaction has taken place. This may be enforced by changing group's administration information every month.

When she decides to deposit her coupons, C contacts her CIB through her ACC. If participating in the ACC offers, C's CIB, CIB checks coupons' validity by contacting the ACCA. The ACCA runs $\mathsf{BGS.Open}$ procedure to recover merchant's name, updates M-related statistics and checks M agreement details ($N_{merchant}$). CIB, using the technique in 9.2.3.4 calculates the overall amount of money the C has to be favored in general, $CreditReduction$, and issues a number of *debit coupons* of equivalent value. Debit Coupons are plain CIB blind signatures on quantities chosen by C $(r'_1, \ldots, r'_N)$ and are – thus – unlinkable to particular credit coupons:

$$\mathsf{BSig}^{r'_1}_{\mathsf{CIB}}, \ldots, \mathsf{BSig}^{r'_N}_{\mathsf{CIB}}.$$

C deposits the debit coupons at her convenience to CIB to reduce her credit amount by $CreditReduction$.

9.2.3 System Considerations

In this section, we will emphasize on particular system issues. For detailed Security Definitions and the corresponding proofs, see Appendix **??**. Here, we will only sketch how the most important of our system's requirements are satisfied.

9.2.3.1 Privacy

As mentioned before, it consists of Cardholder Anonymity w.r.t Bank and the merchant and Cardholder non-Traceability. Both of these properties are satisfied through ecash anonymity and unlinkability properties mentioned in 4. Each payment procedure is a typical ecash EC.Spend procedure from W_p and W_{id} wallets and can thus not be linked to the cardholder C who issued the ACC or to any other spending (transaction) from the same wallets. On the other hand, the anonymity provided is conditional: if C tries to spend more money than his credit-limit, i.e. more ecoins from the initial amount in the two wallets, or lie at the monthly payment calculation procedure — by providing a non-updated version of her ACC —, a part of W_{id} will inevitably be double-spent: to her CIB (CIB) at the CreditUpdate procedure and to a merchant M in a Payment protocol. After M's deposit procedure, CIB detects the double-spending and runs $\mathsf{EC.Identify}_{\mathsf{id}}$ on the double-spent identity ecoins to reveal each double-spender's pk_{C}^{id}. In addition, CIB may run $\mathsf{EC.Trace}_{\mathsf{id}}$ on the same ecoins to trace all the identity ecoins withdrawn by each double-spender.

In the case of ACC promotion offers, the two aforementioned properties are satisfied through the blindness property of blind (group) signatures. When deposited, *credit coupons* are linked to the merchant M who issued them since C's CIB, CIB, contacts the ACCA which is the manager of the group and can identify the coupons' creator (signer). However, because of the blindness property, the ACCA or/and CIB(if colluding with the ACCA) does not know the exact transaction the cardholder participated in, even when colluding with M. In any case, *Credit coupons* are deposited anonymously and the *debit coupons*, which are issued in response to the valid *credit* ones — and deposited by C in person — are blind to the CIB who issued them and thus unlinkable to any particular *credit coupon*.

There are two cases, where we accept a small breach in a cardholder's anonymity/transaction unlinkability: (a) in the *Loss Recovery* and (b) in the *Online Payment* scenarios. At Loss Recovery process, when the most recent $BackUp$ is not up-to-date, C inevitably double-spends a part of her W_p wallet: to the merchants she interacted with and to her CIB. pk_{C}^{s} is then revealed and Bank knows who C interacted with. However, this anonymity breach becomes less important if we require that backups are taken regularly. In the *Online Payment* case, Bank can obviously trace what type of transactions a particular anonymous account

is involved in through D_{hist}. However, thanks to the unlinkability property of the ecash spent at the anonymous account setup phase, linking that profile to a particular identity is impossible. In any case, the cardholder may open as many anonymous accounts she wishes, in order to avoid transaction linkability.

We also address the special case where a cardholder C, is a suspect of an offense and the Judge requests a detailed description of that C's ACC related transactions. In our system, this can be achieved only with C's consent and in a way such that the later cannot lie for her transactions:

a. Cis asked to provide sk^s_{C} for all her transactions to be revealed, which we want to avoid.

b. Cis asked to enter her $pass^t_e$ to decrypt the transaction related part of her ACC and "spend" the rest of her W_p to CIB. Transaction details of each transaction are signed by a merchant or C's CIB (CIB) — in the case of Anonymous Account Setup — and are, thus, impossible to be forged. To check for any deceptive deletion of a transaction on cardholder's side, CIB may use D_{hist} to check whether the overall amount spent matches the aggregated amount in the backed-up transaction details.

9.2.3.2 Security

In this subsection we provide an intuition of how our most important security properties are achieved. For more details, see Appendix A.

ACC Unforgeability — Non Transferability. ACC Unforgeability is satisfied through the Correctness and Unforgeability properties of the underlying ecash schemes. **ACC non-Transferability** is also satisfied, since sk^s_{C} is required for the card to be used in both offline and online purchases.

Security in ACC Transactions. We can study it in two phases: (a) the security of the anonymous account setup and (b) the security of the management of the anonymous account, which includes the transaction payments and the expense reports issued.

In case (a) cardholder C authentication is achieved through the $pass_{pin}$,C is required to enter to create the $\alpha_{\mathsf{C}}(m)$ account. As in the offline payment procedure, only the owner of the ACC may spend ecoins from the wallets in it. Double-spending tracing mechanism restricts C from using the spent part of her wallets elsewhere.

Security in case (b) is achieved through the non-invertibility property of hash functions and the unforgeability of the digital signature schemes. More specifically, it is not possible for an unauthorized party to use the balance of an account even if she knows the current anonymous account number (A^x_{C}); knowledge of H_{ot} and A^x_{C}'s pre-image w.r.t. H_{ot} is required. Signed endorsement $Rec_{\alpha_{\mathsf{C}}(m)}$ of CIB on the initial account's value, prevent CIB from cheating. In addition, as T_{det} and *credm* are part of $\mathsf{Info}_{\mathsf{CIB}}$, i.e. encrypted with a key only CIB and C may derive, G cannot lie for the *price* or the merchant M the payment is for. As timestamps are included in every message and account numbers change in every authorized request, replay attacks or offline account guessing attacks cannot succeed. Expense Report authorization is achieved in an exact similar way.

9.2.3.3 Error Correction vs. Privacy

As mentioned before, in order to request a refund for an charging error having occurred in the expense report, the cardholder C is required to present $\mathsf{Rec}_{\mathsf{T}}$ to the ACCA in person. In offline transactions, the latter does not constitute a problem since $\mathsf{Rec}_{\mathsf{T}}$ is completely unlinkable to other transactions. However, if $\mathsf{Rec}_{\mathsf{T}}$ refers to an online transaction, then — since CIB has all the information regarding the (online) transaction activity of the anonymous account — C is automatically linked to all the transactions of that particular anonymous account. We address this problem in two ways:

- We reduce the amount of transaction information linked to each account by encouraging cardholders to open several — as opposed to one — anonymous accounts for their online purchases. This security measure becomes even more attractive if we consider the fact that *Anonymous Account Setup* is similar in terms of computation to an offline payment procedure,.

- We reduce the likelihood of an error taking place by introducing additional security measures at the online transaction payment procedure. Apart from the current C authentication procedure in *Transaction Payment*, we require that the CIB obtains an email-based transaction confirmation by the owner of the anonymous account. Therefore, we make the following changes:

  - At the *Anonymous Account Setup*. In addition to $\mathsf{P}_{\alpha}^{\mathsf{C}}$, C, provides the CIB with an email address, $\mathsf{email}_{P_{\alpha}^{\mathsf{C}}}$, which is added to $\alpha_{\mathsf{C}}(m)$. It is critical that $\mathsf{email}_{P_{\alpha}^{\mathsf{C}}}$ contains no C-identification information.
  - At the *Transaction Payment*. At step 1, a nonce *nonce* is added to the $\mathsf{TInfo}_{\mathsf{CIB}}$ send by C to CIB:

    $$\mathsf{TInfo}_{\mathsf{CIB}} \;=\; \{\{\mathsf{Cred}_{\mathsf{M}},\; T_{det},\; nonce\}_{A_C^{x-1}},\; A_C^x\}_{pk_{CIB}^e},$$

    while after step 4 a confirmation email is sent to $\mathsf{email}_{P_{\alpha}^{\mathsf{C}}}$ address with T_{det} and a function F of *nonce*, to which C is required to respond for her purchase to be completed. If no confirmation is received within a particular time interval a transaction rejection procedure on behalf of CIB takes place.

  However, we do need to emphasize on the fact that involving electronic mail procedures in the online purchase procedure, is likely to introduce other privacy related concerns: Because of source tracing information they may include, account owners' confirmations may enable Banks/Merchants to link individual transactions from different accounts as having been done by the same person.

9.2.3.4 Credit Reduction Calculation

This is the method used for the bank to estimate the amount by which the monthly payment of a user should be reduced based on the coupons the latter has acquired. Let the following notation:

- r the interest applied in the card remaining amount each month. We basically need to "undo" interest rate application for the coupon-amount.
- t_p^M: Threshold value for issuing coupon agreed with a merchant M.

- v_p^M: Value of each coupon issued by Merchant M.
- N_M: Number of times the amount of money spend by client can be payed out in without interest.
- A_C: Amount of money customer C spent in merchant M. It is obvious that for a coupon to be issued, it should be: $A_C \geq t_p^M$.

A customer who takes advantage of the offer, obtains $n_C = \lfloor \frac{A_C}{v_p^M} \rfloor$ coupons. As mentioned before, depending on the agreement merchant has made with the ACCA, outpayment of a coupon will be distributed in N_M months. Thus, if the customer payed the normal interest rate r for outpaying amount $A'_C = n_C \times v_p^M$, he would have to pay:

$$CustomerOvercharge =$$
$$r \times \{1 + \frac{N_M - 1}{N_M} + \frac{N_M - 2}{N_M} + \ldots + \frac{1}{N_M}\} \times A'_C =$$
$$r \times \{\frac{N_M \times (N_M - 1)}{2 \times N_M}\} \times A'_C =$$
$$r \times \frac{N_M - 1}{2} \times A'_C$$

The main concept is here that we make customer pay this additional amount per month but after having removed this amount from his credit card in advance. Namely, during the montnly credit card amount update procedure, his credit amount gets "enhanced" [4] by the additional amount she will have to pay during the following months. However, there is a chaveat: the amount substructed from customer's credit account should be calculated in a fair way towards the Bank. In particular, assume that $CreditReduction$ is the amount removed from customer's credit account, Bank will lose all the interest payments on it. Namely, Bank will lose at most

$$CreditReduction \times (1 + r)^{N_M} - CreditReduction.$$

Thus $CreditReduction$ should satisfy the following:

$$CreditReduction = CustomerOvercharge - CreditReduction \times \{(1 + r)^{N_M} - 1\} \Longleftrightarrow$$

[4] By enhanced we mean in favor of the customer, namely the amount customer is credited is reduced.

$$CreditReduction \times \{(1+r)^{N_M} - 1\} = CustomerOvercharge \Longleftrightarrow$$
$$CreditReduction = \frac{CustomerOvercharge}{(1+r)^{N_M} - 1} \Longleftrightarrow$$
$$CreditReduction = r \times A'_C \times \frac{1}{2}\frac{N_M - 1}{(1+r)^{N_M} - 1}$$

Given the fact that r ranges from 0.01 to 0.05, and N_M from 3 to 20, we can see how $CreditReduction$ is about 30-40% of the initial amount paid. The reason we decided that this amount is added to customer's credit account as opposed to her savings one, is the way Banks operate, depositing actual money to customers savings accounts would require from them cash that may not be willing to pay in advance.

9.2.3.5 Other Issues

Bank Dishonesty. $BackUp_B$ is used to avoid any attempt of a CIB to trick a cardholder into tracing more of the latter's transactions: Assuming the CIB provided a less recent backup, then a bigger part of W_p would be double-spent and more merchants would be directly linked to the cardholder. $BackUp_B$ will act as an undeniable proof of the date and integrity of the backup kept.

Merchant's Honesty. Coupons's deposit procedure enables a merchant-cardholder to use the coupons he can issue to his customers for his own favor. We address this problem by enforcing ACCA to grant a particular number of coupons to each merchant. Coupons' number is proportionate to merchant's sales and, if restricted, merchant will be motivated to use it to attract customers as opposed to use them for his own purposes.

ACC Data Organization. ACCs' content is organized in the following way: $\{E_{T^1}, \ldots, E_{T^\ell}, padding, E_{W'_p},$ where ℓ is the number of transactions a cardholder has participated in and *padding* is used to avoid any information leakage regarding ℓ. This modular way of encryption is necessary for each of the procedures mentioned before to be able to be executed individually.

Computing power. Credit card customers in our system lack in computing power: not all of them have or know how to install software able to encrypt/ decrypt text, verify hashes,

which are used in our system. A solution on this problem would be that CIBs provide their customers with special machines dealing with card encryption/decryption issues. The extra cost of these devices, may be provided by the cardholder as an extra price for her privacy.

9.2.4 Related Work

Privacy Preserving Payment Mechanisms have been introduced in the past. A very detailed report on the state of the art of electronic payment systems was first done by Asokan e.t.l. in [Asokan *et al.*, 1999]. [Krawczyk, 1999] and [Bellare *et al.*, 2000] are credit card related protocols securing or blinding credit card information from third parties or merchant respectively but *not towards banks*. Credit Cards providing cardholder anonymity even towards the banks were introduced in 1994 by Low etl. [Low *et al.*, 1994]. However, their scheme involves many trusted parties and offers no expense report or error correction service. Current schemes have some of the privacy problems mentioned in the introductory section of this chapter and none of them provide the functionalities of our scheme:

Ecash Ecash [Camenisch *et al.*, 2005; D. Chaum and Naor, 1990] is a substitute of money on the Internet which cannot be faked, copied or spent more than once. It is known to provide absolute anonymity, namely no one can relate a particular ecash coin (ecoin) with its owner. One would argue that ecash could solve the problem we described before. Consumers can indeed buy anonymous ecoins from a bank/mint and use them in their online transactions. [Brands, 1995] is an ecash based electronic payment system taking in consideration real world system threats. However, ecash is a prepayment based — as opposed to the most popular credit based — scheme, and used strictly for online transactions; additionally, the complete anonymity it guarantees gives no opportunities for error correction or expense reporting.

Anonymous Debit Cards(ADCs) Anonymous Debit Cards are prepaid ecash -based cards, which are recharged by cardholders and used to pay for goods anonymously. However, their use is very limited; among the reasons are the lack of error correction and proof of purchase mechanisms; additionally, they operate in a debit rather than a credit fashion, i.e. the amount of money paid by it, is subtracted from one's account,

when the card is initially obtained.

9.3 Privacy-Preserving Risk Management

Credit score management is an critical part of bank risk management. Banks tend to be very accurate regarding this global reputation value as it measures each individual's payment credibility. The exact formula for the extraction of such a value is confidential. However, banks considerate, among others, the amount of credit or credit sources available (credit cars, home equity lines of credit, etc.), income, assets, payment history etc. Requiring consumer anonymity and transaction unlinkability, transaction privacy seems to harden such a detailed calculation. We deal with this in this chapter.

In particular, assuming a credit score mechanism consisting of organizations, users, the credit bureau, and collaborations between banks and the credit bureau, we will show how the technique used in the previous section for the taxation of bank accounts can be used by the credit bureau to evaluate the user on the aforementioned parameters, without compromising transaction privacy.

In the following sections we refer to the requirements and threat model of our system, while we detail how the technique of section 9.1 is applied in the credit score case.

9.3.1 Credit Score Mechanism: Specifications

In addition to the banks and the bank-customers, i.e., the *users*, there is the credit score administration authority, i.e., the *credit bureau* CB and various *organizations*, i.e., companies interacting with users. Organizations represent all the companies or individuals who have currently the authority to affect users' credit scores. The credit bureau receives from the organizations' statements regarding the users' behavior on various activities and updates users' credit records accordingly. Organizations may be classified in groups based on the service they offer.

Similar to banks, the credit bureau is assumed to adopt an *"honest but curious"* behavior, i.e., is trusted to perform all of their functional operations properly, while, they may use the information they have legally acquired to compromise users' privacy and construct

users' transaction profiles. Aiming to maintain their clientele, organizations adopt the same behavior. In particular, although they are trusted to do act legally various organizations may collaborate to build users' profiles.

9.3.1.1 Operations

Similar to the ones of the anonymous bank accounts' taxation, the operations supported by the credit score mechanism include the user membership to organizations, credit report's generation and deposit, and credit score calculation. To define them more strictly we will use the following notation: when an operation is an interactive procedure (or a protocol consisting of multiple procedures) between two entities A and B, we denote it by $\langle O_A, O_B\rangle \leftarrow \mathsf{Pro}(I_C)[A(I_A), B(I_B)]$, where Pro is the name of the procedure (or protocol). O_A (resp. O_B) is the private output of A (resp. B), I_C is the common input of both entities, and I_A (resp. I_B) is the private input of A (resp. B).

1. $(pk_{\mathsf{CB}}, sk_{\mathsf{CB}}) \leftarrow \mathsf{CBkeygen}(1^k)$ is the key generation algorithm for the credit bureau.
2. $(pk_{\mathsf{O}}, sk_{\mathsf{O}}) \leftarrow \mathsf{Okeygen}(1^k)$ is the key generation algorithm for organizations.
3. $(pk_{\mathsf{U}}, sk_{\mathsf{U}}) \leftarrow \mathsf{Ukeygen}(1^k)$ is the key generation algorithm for the users. We call pk_{U} the (master) public key of U, and sk_{U} the master secret key of U.
4. $(\mathsf{P}, sec_{\mathsf{P}}) \leftarrow \mathsf{Pnymgen}(1^k)$ is the pseudonym generation algorithm for users. The sec_{P} is the secret information used to generate the pseudonym P.
5. $\langle(sec_\alpha, pub_\alpha), (\mathsf{D_O}', pub_\alpha)\rangle/\langle\perp,\perp\rangle \leftarrow \mathsf{AccountOpen}\ (pk_{\mathsf{O}}, pk_{\mathsf{CB}})\ [\mathsf{U}(sk_{\mathsf{U}}, \mathsf{cred}), \mathsf{O}(sk_{\mathsf{O}}, \mathsf{D_O})]$. A user U opens an account with an organization O. O's database $\mathsf{D_O}$ is updated with the public account information pub_α, while the user obtains secret membership information. It is apparent that U is not required to deposit his identity, but demonstrate knowledge of a validity credential cred .
6. $\langle R^i, \mathsf{D_O}'\rangle/\langle\perp,\perp\rangle \leftarrow \mathsf{CreditReportIssue}\ (pub_\alpha, pk_{\mathsf{O}})\ [\mathsf{U}(sec_\alpha, sk_{\mathsf{U}}), \mathsf{O}(sk_{\mathsf{O}})]$. A user U having registered to O with account α^{O} (secret information sec_α) and O collaborate for the former to issue a credit report for the behavior of the owner of α^{O}. As in

the tax report case, U demonstrates the account ownership and contributes his secret information to issue credit report R^i.

7. $RR \;/\; \perp \leftarrow \mathsf{CreditReportTransform}(R, sk_{\mathsf{U}})$. A user U having issued a credit report R, transforms it in an unlinkable — nevertheless valid — form RR, for which he can still demonstrate ownership.

8. $\langle \top, \mathsf{D_{reg}}' \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{CreditReportDeposit}\ (pk_{\mathsf{CB}}, pk_{\mathsf{U}}, RR)\ [\mathsf{U}(sk_{\mathsf{U}}), \mathsf{CB}(sk_{\mathsf{CB}}, \mathsf{D_{reg}})]$. User U who has already obtained a credit report RR deposits the latter to the credit bureau CB. CB updates the $\mathsf{D_{reg}}$ database accordingly.

Requirements *Correctness*, *privacy*, *security* are defined in a very similar way as in the anonymous bank accounts' case. We briefly refer to each of them.

Privacy. Having assumed collaboration between various organizations, and having already equalized it as user *transaction untraceability*, privacy in the context of credit score mechanism can be equivalent to *user-account unlinkability* and *account-account unlinkability*, where by "account" we denote a user-membership to an organization. As in the case of tax reports privacy definition is extended to pseudonyms' and credit reports' unlinkability. A small difference with the bank taxation protocol is that as we require a degree of accountability on the organizations' behalf, w.r.t. the credit bureau a user is hidden among the users-members of a particular organization. As organizations who affect the credit score of a user are usually of large scale, we do not consider it to be a serious privacy compromize.

Security. Security in the credit scoring mechanism — consisting of *fairness* and *accountability* — has many similarities with the bank account taxation protocol. More specifically,

- *Fairness* requires that an accurate statement regarding the credibility of the contents or liability of all accounts belonging to a given individual is reported to the the credit bureau per normal practice (i.e., quarterly or annually). As in the case of the taxing system, fairness is directly translated to *credit report unforgeability* and *credit report non-transferability.*

- *Accountability* requires that t also It consists of fairness in the report declaration and users who attempt to avoid to avoid showing the correct reports, are detected.

9.3.2 A privacy-Preserving Credit Score Mechanism

In our scheme an individual U registers to the credit bureau CB to initialize their credit scoring and collaborates with a number of organizations $\mathsf{O}_1, \ldots, \mathsf{O}_N$. Organizations evaluate U's financial activity based on various metrics, which depend on the service they provide.

As in the taxation case, after having registered to the credit bureau, U obtains a single identity (master-secret and the corresponding public information) and the required credentials to initiate collaborations with various organizations. U then contacts organizations anonymously, while authenticating himself through his CB-registration credentials. To accountably manage his memberships he generates membership pseudonyms secretly —but provenly — connected to his master identity. On an annual basis, organizations evaluate their members and issue credit reports to each one of them, which the users transform and deposit *in person*, i.e., as identities to the credit bureau. Credit scores are then calculated by CB based on the aggregated user-behavior against these agents.

In is obvious that the credit bureau plays a role similar to the one taxation authority plays in the taxation protocol, while organizations play the same role as banks play. In opposition to the banking system were each bank would issue multiple tax reports, per user, here, there is exactly one credit report per user membership. It is important to note here that each user can have at most one membership in each organization.

In the following we will show how each of the operations used in the taxation protocol are applied here to provide the same type of functionality.

Setup The credit bureau plays the role of a central authority in our system and thus runs the PS.Setup for the setup of the anonymous credential systems used. All organizations, simulated as organizations in the anonymous credential system of 4.3, run the PS.OKeyGen procedure to generate the public parameters for the generation of their credentials. In addition each of them generates two key pairs: an RSA signature key pair, whose public

information it provides to the CB and an encryption key pair of a known randomized encryption algorithm, whose decryption key is given to the CB, e.g., Paillier (see [Paillier and Pointcheval, 1999] for more details).

Registration with the CB It is identical to the Registration procedure of the taxation protocol (see section 9.1.2 for details). The user obtains a master secret $\mathsf{ms_U}$, which he is highly motivated not to share, a registration credential $\mathsf{credcb_U^{CB}}$ and a wallet of organization membership permissions $\mathsf{perm_O}$. The ecash scheme used for $\mathsf{perm_O}$ is the accountable ecash (see [Camenisch *et al.*, 2006] for more details), which restricts U into using at most one $\mathsf{perm_O}$ per organization.

Membership in an organization It is identical to the Account Open operation of our taxation protocol; U makes use of his $\mathsf{perm_O}$s (ecash coins) and $\mathsf{credcb_U^{CB}}$ to authenticate himself to an organization O_i and open an O_i-membership account. He then generates a pseudonym $\mathsf{P^i}$ which secretly matches his $\mathsf{ms_U}$. As opposed to the taxation system, because of the nature of the ecash scheme used for $\mathsf{perm_O}$s, U cannot register to the same organization more than once. However and as we will discuss in this section, to avoid profiling, users are encouraged to change pseudonyms occasionally.

Credit Report Issue It is very similar to the TaxReportIssue operation of our taxation protocol. In particular, there are three differences from the latter: (a) U obtains the single use credit report numbers CRNs (similar to the TRNs of the taxation protocol) from the CB and not the organization (corresponding to the banks of the taxation mechanism), (b) the R^M (corresponding to T^M) has the following form:

$$R^M = h_{\mathsf{O}}^{a_1:g_1,\ldots,a_m:g_m} \cdot C,$$

where $a_i, g_1, i = 1 \ldots m$ is the name and grade of each attribute O evaluates, C is U's commitment on his $\mathsf{ms_U}$ and CRN, g_{O} is a public parameter of O, and (c) the number x of O-signatures applied on R^M is encrypted with the key pair O exclusively shares with the CB (as opposed to the taxation case, where x is encrypted with the TA-specific encryption key). The first version of credit report (R^σ, R^M) is then transformed by U to its depositable

form (RR^{σ}, RR^{M}) .

Credit Report Deposit. U deposits all the credit reports he has acquired from the various organizations $\mathsf{O}_1, \ldots, \mathsf{O}_N$, $(RR^{\sigma/M,1}, \ldots, RR^{\sigma/M,N})$ to the CB. As in the TaxReportDeposit case, U proves that (a) $RR^{\sigma/M,i}$s for $i = 1 \ldots N$, all correspond to the same user U′ (b) he is U′, i.e., that U ≡ U′, (c) each credit report $RR^{\sigma/M,i}$ has been created with O_i's collaboration (signature), and (d) each of them has not been deposited before. Note that CB knows the organization who issued each credit report. However, because of the unlinkability of $RR^{\sigma/M,i}$ and $R^{\sigma/M,i}$ provided by the transformation function U cannot be linked to a particular pseudonym, and thus U is hidden among O_i's members.

Credit Score Calculation. U reveals and proves knowledge of the sequence of $a_1^i : g_1, \ldots, a_m^i : g_m$ corresponding to credit report $R^{\sigma/M,i}$ from each organization O_i. It is important to note here that $a_1^i : g_1, \ldots, a_m^i : g_m$ is not considered enough to link a pseudonym to an identity when O_i and CB collude, as we assume that given the scale of these organizations there are many users sharing the same description.

Optional Operation: Modifying an Account Credentials. For better privacy provisions, users are encouraged to occasionally change their organization memberships' credentials, so that their updated accounts are not linked to their older ones. In particular: CB knows the exact sequence of $a_1^i : g_1^i, \ldots, a_m^i : g_m^i$ evaluation of user U for every O_i the former is a member of. In short term this cannot link a U to a particular O_i pseudonym, even when CB collaborates with the latter. However, in long term the sequence of various O_i-evaluations of U may constitute a strong distinguisher for the latter.

We, thus add the AccountUpdate procedure, which takes place at fixed time periods. For the setup of this procedure, each organization O_i runs $PS.OKeyGen$ to generate parameters similar to the ones of the organizations in anonymous credential system, aiming to simulate the account update environment. After requesting an account update, U's pseudonym with O_i, P, runs PS.GrantCredOnNym for U to generate a single use anonymous credential for account update purposes. U anonymously contacts the O_i and run PS.VerifyCred to verify the

credential's validity, PS.PNymGen to generate a new pseudonym and PS.VerifyCredOnNym to prove that it corresponds to the same master secret as the old one. Because the ticket is of single-use, each user may update his account once.

9.4 Conclusion

In this chapter we dealt with privacy in online banking activities of an individual. In particular, we introduced a banking system which supports:

- the opening of anonymous accounts for eligible individuals: individuals may open arbitarily anonymous and unlinkable accounts w.r.t. the bank and tax authority collaborations, which are ultimately and in zero knowledge fashion connected to their owner. We supported our construction with a bank account taxation mechanism, where individuals report the aggregated amount of tax withheld by all of their accounts in a fair and privacy preserving way.

- a privacy preserving credit card mechanism; we presented a credit card system which guarantees cardholder anonymity and transaction unlinkability, even towards to credit card associations or card issuing banks, while preserving many of the essential benefits of credit cards: error correction, expense reporting, special card promotion offers, etc. In special circumstances the transactions of a party may be revealed but only with that party's consent.

- a credit score mechanism, where we apply the protocol introduced for anonymous bank accounts' taxation for individuals to have their credit scores updated without endangering their transaction privacy.

Chapter 10

Employment: Payment - Access Control

In chapter 3, we referred to the privacy and security issues raised from the individuals' involvement with the online world. Our primal focus being the the protection of financial data and having already dealt with the commercial activity of individuals, we now will refer to employment. More specifically, in this chapter we will elaborate on privacy and security issues raised from individuals' payments and rightful access to critical cyber infrastructures.

As it involves no purchase activity, privacy here may acquire a slightly different meaning than the transaction privacy we dealt with before. More specifically, it is likely that complete user anonymity is not an issue towards employers. Employers do need to know the full identity of their employees. In fact, in some extreme cases of national security organizations, details of the personal life of the individual are necessary to decide for that individuals credibility. Consequently, the privacy definition in the context of users employment systems is restricted to the avoidance of any bank-account privacy breach due to employers interaction with the banks or taxation authority throughout employees payment and taxation respectively.

Another employment-wise cybersecurity concern is access control for specified sites. We emphasize the common case where we need to have strong authentication within the company but complete anonymity outside the company. A typical example for that would

be in critical infrastructure systems: when an employee of a particular company logs in to a critical infrastructures website, such as a SCADA system, the employees exact identity should be knowable by that particular companys department, while for any entity outside the company, the employee should be hidden among the employees of that department (company). In any case, it should be possible to trace a misbehaving party.

Considering the aforementioned functionalities as an extension to our centralized identity management system and in response to the presented privacy and security issues, we provide a set of communication protocols, that guarantee:

1. privacy preserving and fair monthly payment; an employee should be fairly paid by his employer, without risking his bank account related privacy provisions (see chapter **??**): bank-employer collaboration should not be able to link an identity to a particular anonymous bank account.

2. privacy preserving access control in critical infrastructure; in particular, we construct an access control system, where only authorized individuals may access a particular space and do their job as entitled. The exact identity of the party entered the system should not be revealed towards any entity outside the employee's company, nevertheless the employer company of the individual should be able to recover the identity of that person when the latter uses his employment credentials to misbehave.

Organization. In what follows we elaborate on our threat model and requirements, while in sections 10.2 and 10.3 we will present and discuss our protocols.

10.1 Employment Architecture

The core entities in this part of our system, are the registration and taxation authorities, RA and TA respectively, the individual users (we will refer to them as users), who may interact with the aforementioned authorities to register to the system and get taxed, the banks, where the users may maintain nominal or anonymous accounts (see chapter 9) and the employers. The employers form employment relationships with users and are responsible for reporting income and corresponding tax withholding. Employers may be any type of real-world employers.

Threat Model. Our threat model is built on top of the assumptions we have made in the previous chapters for each of the entities involved. More specifically, we assume the following:

- *Users may try to cheat.* A user may actively or passively try to increase its income, i.e., to avoid paying taxes or lie for not having received his payment, try to impersonate other users to use their funds (impersonation attack) or frame other users to appear guilty for the his malicious actions. Under the aforementioned assumptions, we further assume that a user is motivated enough to attempt any type of forgery.

- *Banks are "honest but curious".* Aiming to maintain their clientele, banks are trusted to perform all their functional operations correctly, i.e., they issue credentials, open and update accounts as instructed by their customers. On the other hand, we assume that banks may use the information they possess to compromise customers' privacy for other reasons: motivated by profits gained through selling their costumers' profiles to e.g. advertising companies, banks may collaborate with tax authority or employers to reveal the identity behind a (swiss) anonymous account.

- *Employers may be either "honest" or "malicious".* In the general case of powerful employers, we assume "honesty" in payments towards the users, while they may try to avoid paying taxes properly. On the other hand, smaller employers may try to avoid paying employees on time or avoid paying amounts due to former employees.

- *Tax* (TA) *and Registration* (RA) *Authorities are considered to be "honest but curious".* Although we assume that are operated by the government who wants to protect honest users, the authority officials are not assumed to protect privacy; indeed, there have been a number of incidents in the U.S. of privacy violations by tax authorities or by unscrupulous individuals employed by the tax authorities.

Requirements. Privacy in this case is limited to the maintenance of individuals' transaction privacy and incorporation of two levels of individual's professional activity untraceability, where by "professional activity" we denote any company related data access or data manipulation. More specifically, we require:

- w.r.t. transaction privacy, we require that no employer-bank or employer-taxation authority interactions for payment and taxation purposes respectively, would reveal the owner of a particular account. It is conspicuous that in such a case, that account's transaction history would be linked to a particular identity.

- w.r.t. users' professional activity, we define two degrees of user-activity traceability:

  1. *absolute traceability within the company or department* a user is a member of.
  2. *user group-anonymity — user-activity untraceability* outside the particular department. In this case, we require that every time an honest user enters an external database, he will be identified as a member of his employer's company, while consequent visits of the same user cannot be linked. However, because of the previous point, the transcript of the login procedure, should be enough for the employer to identify a misbehaving member of his company who entered an external database.

Security on the other hand consists of fairness and accountability as we require that all entities are paid in accordance to their work, while misbehaving parties are traceable and identifiable.

10.2 Privacy-Preserving Employment Payment and Access Control

As in the case with banks, employers will not accept registration without knowing the identity of the person they are about to hire. In fact, as mentioned before, in special national security organizations, details of the personal life of the individual may be required to decide for that individuals credibility. Proper identification at the registration procedure facilitates the issue of a single employee identifier per employee, which agrees with the accountability requirement of such a system according to which each user of the system should have a unique identifier to be ultimately accounted for if misbehaves.

Except for the registration procedure, there are three important operations in our system, which the employee obtains credentials for: the employee's monthly payment Salary-

Payment, the issue of employee's tax report TaxReportIssue and the data access operation AccessDB. All these protocols are based on group signature schemes, blind signatures and anonymous credentials, which have been detailed in chapter 4.

More specifically, SalaryPayment bases its functionality in one time use anonymous credentials the user acquires at his registration procedure. There are different types of credentials depending on the month and the salary level they correspond too. Blind towards the employer, nevertheless strongly connected to the employee's identifier, monthly issued payment credentials authorize each employee to set an account number for his salary to be deposited. In this way, employees and account numbers are grouped together w.r.t. salary level. Fired employees do not receive these credentials.

For TaxReportIssue the employer directly contacts the tax authority with the exact income each employee obtains. Note that there is no need for employee anonymity in this case. The employer already knows the identity of the employee, while for security purposes, it is realistic to assume that the tax authority knows the primal source of income of each user. We emphasize on the fact that, as long as employee's transaction activity remains concealed, we do not consider *it* to be a privacy breach.

AccessDB is basically a simple application of group signature schemes. Each employee generates group membership credentials at the registration procedure, which he may use to be authorized to access external websites. From the properties of group signatures, the user maintains his anonymity outside the company, while the company administration may always revoke the former's anonymity. In addition, he can use his employee id to access internal data.

More, thoroughly, the detailed series of interactions are the following:

10.2.1 System Setup

All employment administration offices commit in the following procedures:

- run the GS.Setup procedure to generate the administration information to handle the group of its employees. In large companies, as we may realistically assume that the registration of each employee is handled by the corresponding department, we equivalently transfer the setup of the employee group to the departments. In the end of

this procedure, each department D generates a public group key and an administration group key: $(\mathsf{gpk}^{\mathsf{D}}, \mathsf{gsk}^{\mathsf{D}})$.

- run the GS.Setup procedure to generate the administration information for all the groups realizing the various data access control roles. The role of these groups is more specialized: only employees who may — because of their post — be entitled access to particular critical infrastructure websites may participate in those. Thus, the group administration key is generated for each access role: $(\mathsf{gpk}^{\mathsf{D}^{ac}}, \mathsf{gsk}^{\mathsf{D}^{ac}})$
- run the Pseudo.Setup procedure to simulate the anonymous and unlinkable anonymous credentials website. This will serve payment privacy purposes.

10.2.2 Registration (EmployeeRegistration)

This takes place between an Employer or departmental administration Dep and the new employee U.

1. $\mathsf{D} \leftrightarrow \mathsf{U}$: run GS.Join for the two groups: the D's staff and U's access control level (let it be i) group DepAC_i. Thus, U obtains the two group membership key pairs:

$$(\mathsf{gpk}_{\mathsf{U}}^{\mathsf{D}}, \mathsf{gsk}_{\mathsf{U}}^{\mathsf{D}}) \text{ and } (\mathsf{gpk}_{\mathsf{U}}^{\mathsf{D}_i^{ac}}, \mathsf{gsk}_{\mathsf{U}}^{\mathsf{D}_i^{ac}}).$$

2. $\mathsf{D} \leftrightarrow \mathsf{U}$: run PS.UKeyGen and PS.NymGen for U to generate his identifying pair (P_{U},ms_{U}). Here, P_{U} may be considered as the employee identifier while ms_{U} the secret key information, which plays the role of the master secret in credential systems.

3. $\mathsf{D} \leftrightarrow \mathsf{U}$: run PS.GrantCred for D to issue a one time use salary credential to U, $\mathsf{cred}^{\mathsf{s}_i}$. U is required to update this credential every month.

4. D: stores the tuple

$$\{\ \mathsf{U},\ \mathsf{P}_{\mathsf{U}},\ \mathsf{gpk}_{\mathsf{U}}^{\mathsf{D}_i^{ac}},\ \mathsf{gpk}_{\mathsf{U}}^{\mathsf{D}},\ \mathsf{AC}_i,\ \mathsf{s}_j\ \},$$

where AC_i is the data access level of employee U, and s_j the salary level of U.

10.2.3 Salary Payment (SalaryPayment)

At some point within the month, U contacts D, and they both run PS.VerifyCredOnNym, for U to demonstrate ownership of $\mathsf{cred}^{\mathsf{s}_i}$. U provides an anonymous account number, where D may deposit the payment. It is apparent that if U is fired he will not have $\mathsf{cred}^{\mathsf{s}_i}$. At the end of each month D deposits the payments in the indicated account numbers.

10.2.4 Access Control Management

This protocol takes place between a user U and a verifier V, where Uneeds to prove to V that he is of a particular access role AC_i. V may be any external data administration authority. Both parties collaborate in a GS.Sign operation for U produce a signature on a V-specified random quantity R, σ. V runs GS.Verify on σ, to concurrently verify that U is a member of D with access role AC_i and stores σ. If U misbehaves while accessing the data, σ is given to D for the latter to run GS.Identify and identify the signer of σ. Inside the company U is directly identified through $\mathsf{P_U}$. Current group signature schemes allow direct member removal from the group without the participation of the revoked member to be required. Such revocation is utilized when an employee is fired or changes data access level.

10.3 Discussion

Both privacy and security are satisfied directly from the primitives we made use of.

10.3.1 Privacy

We will elaborate separately on each privacy type we mentioned in section 10.1:

- *Transactional Privacy Maintenance*, where we required that privacy provisions in online banking are maintained through the interactions of user's employer with the banks and taxation authority. This is achieved through the security properties of anonymous and unlinkable credentials of [Camenisch and Lysyanskaya, 2001]. In particular anonymity of payment credentials guarantees that $\mathsf{cred}^{\mathsf{s}_i}$ s are unlinkable

to the particular U for whom it was issued.[1] Thus, the account number provided to the employer's administration office is unlinkable to the particular employee towards the employer, or any bank- employer collaboration. Here we need to note that the privacy provided here is proportional to the number of the employees who have account to the same bank.

- *Professional activity untraceability*, which is guaranteed through the *anonymity*, *unlinkability* security properties of group signature schemes, which are used for authorization purposes in our system. A valid employee (group member) may be able to enter critical infrastructure websites while being identified as member of his company of a particular access level. No more information may be inferred regarding the user. Conditionality of untraceability, however, is achieved through the *openess* property of group signatures which can be utilized by the employer in case a user misbehaves for the user to be accurately identified.

10.3.2 Security

Fairness and Accountability are also achieved in our system.

Fairness in payments is satisfied through the payment credential unforgeability and its secret connection to its owner: only the owner of the secret key of the user who participated in the issue of the credential may successfully demonstrate ownership of it. To prevent transfer of such credential, our system adopts "all or nothing non transferability": employee's authentication and authorization is strongly connected with his employee id and the secret related to it. Employees are thus strongly motivated to share their sk. Fired employees are not entitled of payment credentials and are thus not paid.

Accountability in payments is achieved through the "conditionality" of privacy, which we mentioned before. As each payment credential is of single use, if its owner tries to demonstrate it multiple times, i.e., to be paid twice, his identity and secret information will be revealed. Thus the user is strongly motivated not to attempt to cheat in this way. In access control case, accountability is satisfied through the corresponding property of the

[1] However, only U or the one owning the sk_{U} may be able to successfully prove ownership of $\mathsf{cred}^{\mathsf{s}_i}$.

underlying group signature schemes.

10.4 Conclusion

In this chapter we dealt with privacy and security issues regarding employment. In particular, we introduced two sets of protocols which, based on existing groups signature and anonymous credential systems, provide

- privacy preserving and fair monthly payment to the employees of a company; in particular, we guarantee that payments are performed without employer-bank collaboration to compromise the anonymity of an employee-owned anonymous accounts.
- privacy preserving access control in critical infrastructure; in particular, we construct an access control system, where only authorized individuals may access a particular space and do their job as entitled. The exact identity of the party entered the system should not be revealed towards any entity outside the employee's company, nevertheless the employer company of the individual should be able to recover the identity of that person when the latter uses his employment credentials to misbehave.

We also take in consideration the revocability of payment and access control rights in the case where an employee is fired.

Chapter 11

Binding all together

In previous chapters we presented in detail how we may achieve privacy in many parts of individuals' online activity. Having incorporated "real world" assumptions in our threat model and security requirements, we combined privacy and accountability in eBanking and eCommerce activities, and developed protocols to maintain both properties through individuals' credit score calculation, tax reporting and salary payment operatoins, which are closely connected to each user's identity. Acting as a single point of reference for all distinct procedures to guarantee that a single identity is accounted for repetitive misbehaviors of the same individual within our privacy-preserving architecture, a privacy-preserving but centralized authentication mechanism equipped with many "real world" mechanisms, e.g., credentials' loss recovery, credentials' blacklisting etc. has been designed. Privacy and security have been defined and proven to hold in each section of this architecture independently. It is however very interesting to see the limitations in the application of privacy and accountability of each section when separated functionalities are combined together. In this chapter we present an overview of the privacy and security provided by each of our protocols when applied separately and discuss their limitations when all protocols are combined. For the latter, we demonstrate how resistant privacy and security are against an extended threat model, which includes collaboration of more entities and analyze the results that a privacy or security breach of an individual mechanism would cause to the rest.

11.1 Individual Privacy and Security Provisions' Overview

Absolute privacy has been defined in section 3.1 as the combination of user anonymity w.r.t. a particular activity (user anonymity) and user activities' unlinkability (non-profiling). *Activities* range from browsing, clicking ads and transaction payments to account management and employee salary payment. Security on the other hand requires that all entities are paid in proportion to their services (fairness) and that users are not able to misbehave without being caught and punished (accountability). In many cases, accountability is strongly connected to conditionality of privacy: users who misbehave are detected and thus have their privacy revoked. Table 11.1 depicts the privacy provisions in each part of our architecture and their underlying assumptions, while table 11.2 demonstrates how various types of misbehaviors (security attacks) are dealt with. In more detail, we dealt with privacy issues in:

1. *browsing*, where we mitigated privacy violations by

   (a) strengthening privacy provisions of existing anonymizing networks. In particular, we designed a payment mechanism for Tor, where payments act as incentive for both participation and honest behavior. Our payment mechanism maintains the privacy provisions offered by the underlying anonymizing network: user-anonymity and connection unlinkability w.r.t. a local adversary. Misbehaving users, i.e., nodes that do not properly deliver their packets as directed, do not get paid.

   (b) designing a targeted-ad mechanism which guarantees that as long as payments are privacy-preserving, i.e., anonymous and unlinkable to other purchases, a user cannot be profiled or identified. In this case, the privacy adversary is passive and may obtain information from curious publishers, ad-networks, advertisers and their collaborations. The security threats in this case are strongly connected click fraud and thus in our security model, publishers are considered malicious. For fairness purposes, a user having clicked on the same link more than a predefined number of times (M) should allow his activity for that link to be traceable. In this way, existing (non-privacy-preserving) click fraud detection methods may apply

for proper accounting to take place. Otherwise, if a malicious user attempts to hide his clicks, accountability-provisions of our target system imposes that that user's identity is revealed, while it is resistant to impersonation attacks.

2. *online transactions*, where we introduced a credit-card-based anonymous payment system and a privacy-preserving product-delivery service. Anonymous credit cards enable the user to open an anonymous account or add money to an existing one to make the payments from. The fewer the transactions that are linked to an anonymous account, the less accurate the profiling of its anonymous owner would be by the banks (user-dependent profiling). Users who attempt to exceed their credit limit will either have their identity revealed or their transaction failed. Assuming anonymous payment and privacy-preserving browsing, our product-delivery service guarantees user-merchant unlinkability w.r.t. delivery companies and user anonymity maintenance w.r.t. merchant and merchant – delivery companies collaborations. On the other hand, users who attempt to cheat at delivery payment procedure never receive their purchase.

3. *online banking*, where – apart from the credit card mechanism, we extended the privacy provisions

   (a) to any bank account activity by developing an anonymous account mechanism on top of the existing non-anonymous bank system. A user may open an arbitrary number of anonymous accounts which are unlinkable one to the other and he may use to pay monthly subscriptions in various services. It is apparent the user's control over his profiling: the more anonymous accounts a user owns and the fewer services associated with each of them, the less accurate an account profile can be. The owners of account owners are taxed fairly, i.e., in proportion to their balances, while users who attempt to cheat in taxes are detected and punished.

   (b) to credit score calculation, by designing a privacy-preserving method for user to accurately report his "payment behavior" score from all the services he has registered to in a way so that the honest but curious credit bureau will not be able to link a service membership (and its profile) to a particular identity, even

with the collaboration of that service's manager.

4. *employment payment*; aiming to maintain bank account privacy, we introduced a salary payment method, where anonymous accounts are not linked to particular employees even when the employer colludes with the bank.

5. *user-authentication*; we enhanced the existing non-privacy-preserving authentication techniques with techniques for online authentication which incorporate user-anonymity and authentication transcripts' unlinkability. This authentication mechanism also supports proof of ownership of various credentials. However, if a person misbehaves i.e., acts against the rules of the particular service or organization he has registered to and depending on the importance of the latter, the user can be locally or globally blacklisted and be no longer able to authenticate himself as honest user of the system.

For convenience, table 11.2 presents an overview of various types of misbehaviors and their punishments. It is noticeable how fairness and accountability alternate each in cases of user-misbehavior. More specifically, assuming that malicious parties aim to increase their profit, misbehavior acts may either be proven worthless (fairness) or be detected and punished (accountability). Evidently in our payment mechanism for anonymous networks we prioritize fairness as parties who attempt not to pay receive no service; on the other hand, in our anonymous credit card system fairness is achieved through accountability: parties who attempt to exceed their credit limit, are immediately traced.

Having already overviewed privacy provisions in individual parts of our architecture, in the following sections we will discuss the privacy and accountability achieved through their combination.

11.2 Overall Privacy and Security Provisions

Our overall architecture consists of a set of protocols achieving *real-world* privacy in online banking, online transactions, employment access control, and a set of protocols that guarantee that the aforementioned privacy provisions are maintained even through collaborations for operations of global and non-anonymous nature, i.e., credit score calculation, tax re-

porting, employees' payment etc. By *real-world* we emphasize on the fact that the privacy achieved in our protocols is not unconditional, but rather applied to the degree it can be combined with accountability. Our architecture dealing with such a variety of functions, it is conceivable how fundamental the authentication mechanism is for both privacy and security maintenance through the interaction of system's components. In the what follows we will elaborate on privacy and security definitions of the overall system (see section 3.3) and on how they are achieved.

11.2.1 Privacy

The definition of generalized privacy requires that the privacy definitions adopted in each component of our architecture are not violated when entities from different components interact or combine the information they have legally obtained. As , as mentioned in section 3.2, our privacy adversary consists of honest but curious behavior of all entities[1] participating in our architecture (including malicious users) and collaborations that are restricted to money-dependent entities.

Activity unlinkability aspect of privacy requires that given the transcripts of distinguished operations that do not belong to a corrupted party, the adversary has no advantage in telling whether they belong to the same user or not. Generalized privacy also requires that user anonymity in anonymous activities is maintained despite of the non-anonymity of others. In particular, it requires that assuming a set of mixed (identified and anonymous) transcripts of users' online activities, i.e., operations in online banking, a complete transaction (ad clicking, payment, product delivery, transaction evaluation), user (anonymous or not) authentication, employee payments $T_1, \ldots, T_n$, tax or credit score calculation, that do not belong to a corrupted party, the adversary should be able to find the owner of an anonymous T_i no better than guessing at random among all non-corrupted users that appear consistent with that set of transcripts. More specifically,

1. ***Transaction privacy*** requires:

[1]Note that in chapter 7 we consider malicious publishers; however this threat model corresponds to system's security rather than to users' privacy.

(a) *Privacy in Payment-Delivery.* Consumer's anonymity requires that given the data/transcripts of a payment procedure T_p(anonymous account number, payment authorization information) and the corresponding product delivery T_d, that do not belong to a corrupted party, the adversary cannot link either each transcript separately or their combination to an identity with a better probability than choosing at random among non-corrupted users. Transactions' unlinkability requires that given transcripts of two completed transactions (payment-delivery pairs) of different type T^1_{pd} and T^2_{pd}, which do not belong to a corrupted party, the adversary has no advantage in telling whether T^1_{pd} and T^2_{pd} correspond to the same user.

Proof. For the first case, since both operations adopt the same adversarial model, anonymity property is achieved directly from the anonymity properties of the two set of protocols utilized. As far as unlinkability is concerned, as in the case of anonymous payments, the user who performs the transaction has absolute control over the degree of his transactions' unlikability (to which he can be profiled): use of the same account for multiple transactions (with or without the same merchant) may enable the bank (or the merchant) to accurately estimate user's preferences. On the contrary, because of the unlinkability property of the anonymous accounts, use of a different account for each type of transaction would considerably restrict the extend of profiling.

(b) *Privacy in Payment-Delivery-Evaluation.* Given the data/transcripts of a completed transaction, i.e., payment and product delivery T_{pd} and the corresponding transaction evaluation procedure T_e which do not belong to a corrupted party, the adversary does not gain any advantage in recovering the identity of the user. In addition, given two transcripts of such sets of collaborations T^1_{pde} and T^2_{pde}, where $T^i_{pde} = T^i_{pd}||T^i_e, i = 1, 2$ that do not belong to a corrupted party, the adversary does not have an advantage in deciding whether T^1_{pde} and T^2_{pde} belong to the same party.

Proof. Both properties are satisfied through the anonymity and unlinkability properties of the individual operations: given the privacy property in payment and product delivery operations, transaction evaluation guarantees that users

who rate each other maintain their anonymity, and that evaluation transcripts are totally unlinkable one to the other.

2. ***eCommerce privacy*** requires that transaction privacy is maintained throughout the combination of transactions and browsing. Assume that I_{ad} is the information an ad-network has collected through our privacy-preserving ad-system for an honest user who clicked on an ad and completed a transaction with transcript T_{pde}. First of all, it is essential that the adversary does not gain any information w.r.t. the identity behind T_{pde} or I_{ad}. More importantly, we require that I_{ad} will not enable linkability of transactions, i.e., that given the transcripts of two independent and completed transactions T_{pde} and T'_{pde} and their click information I_{ad} and I'_{ad}, the adversary has no advantage in deciding whether the two groups of transcripts EC and EC' (, where $EC = \{T_{pde}||I_{ad}\}$ and $EC' = \{T'_{pde}||I'_{ad}\}$) belong to the same, non-corrupted user or not.

   Proof. Both properties are achieved through the anonymity and unlinkability properties of transaction and click-info transcripts. More specifically, our privacy preserving ad-system guarantees that each click is independent in terms of information it carries and that I_{ad} and I'_{ad} are both anonymous and unlinkable one to the other. In addition, we showed before that T_{pde} and T'_{pde} are also anonymous and unlinkable one to the other, which implies anonymity and unlinkability of EC and EC'.

3. ***Privacy in Banking*** requires that anonymous accounts' anonymity and unlinkability is maintained through user's banking activities. Banking activities my be classified in

   (a) <u>*Transactions*</u>. We assume two anonymous accounts α_1 and α_2 which belong to non corrupted parties. We also assume that each of them has participated in a set of anonymous transactions with transcripts σT^1_{pde} and σT^2_{pde}. Note that since our threat model permits collaborations of bank, merchants and their delivery-company, it is implicitly assumed that in its most powerful version, our adversary has acquired data emitted by our protocols in all three operations: transaction payment, product delivery and evaluation. Banking privacy requires that given

σT^1_{pde} and σT^2_{pde}, the adversary does not have an advantage in linking the accounts to their owners or deciding whether α_1 and α_2 belong to the same user.

Proof. It is direct consequence of anonymity and unlinkability of transactions (see previous points), which guarantee that σT^1_{pde} and σT^2_{pde} are both anonymous and unlinkable one to the other.

(b) *Taxation*, which is extensively discussed in section 9.1.

(c) *Direct deposits from various organizations the user has a nominal membership in.* Employers who have to pay their employees constitute a representative example of organization in which users (employees) register non-anonymously and which require account numbers to deposit the salaries. In chapter 10 we present in detail a set of protocols that enable maintenance of account anonymity and account-account unlinkability towards the bank and bank-employer collaborations.

In addition, privacy in banking requires that credit score calculation reveals no information regarding the exact companies he has been collaborating. This is exactly the topic of section 9.3.

4. ***Organization Registration Privacy.*** Ideally, privacy here requires that multiple identified interactions of a user are not linked to any of his anonymous ones, and that anonymous interactions of a user within an organization are unlinkable to similar interactions of the same user with other organizations.

   Proof. This property is directly achieved through our authentication scheme, as it guarantees that user's authentication procedures are and unlinkable to each other. Thus it is impossible that any identified user activity is linked to any of his anonymous ones. On the other hand, assuming that all user's registrations are anonymous the only chance that multiple registrations are linked together is if they refer to the same anonymous account, e.g., if a user has provided the same account number to two distinct but anonymous-subscriptions of his, the bank may link the two subscriptions. Anonymity is nevertheless protected.[2] Apparently, it is on user's decision the degree

[2]We on purposely omitted the cases where a user makes use of a non-anonymous account for his anonymous subscriptions or that he makes use of an anonymous account for an identified membership of his (unless

of untraceability that he wants to enforce.

11.2.2 Security

As mentioned in chapter 3, security consists of fairness, and accountability. Fairness requires that all entities are paid in proportion to their services and accountability that an individual accounts for any malicious behavior of his. Table 11.2 displays our interpretations of misbehavior in each part of our system and the way we deal with them. As implied, motivated to increase their profit, users are considered capable of performing any type of cheating, i.e., forge credentials, impersonate others, lie for payments. Banks and merchants are considered honest as they are incentivized to maintain their clientele. Merchants may lie only regarding having received their payments.

In banking a user is considered to be misbehaving when he attempts to impersonate another user, to spend more than his account balance or credit limit, to lie for the taxes withheld by his accounts or claim ownership of another person's account. We face all these attacks by adopting strong authentication in all banking activities. In particular, each user owns a single master secret which he can prove ownership of privately, and which he is required to authorize himself to make use of his funds. In this way, all-or-nothing non transferability of authentication credentials acts towards a privacy-preserving centralization of bank activities and — thus — against impersonation attacks. Exceeding the credit limit results in revelry of the identity of the attacker and traceability of all the accounts owned by the latter. More naive attacks, e.g., attempts to spend more that the account balance, result in transaction failure, while users who attempt to lie in tax reporting are immediately detected in tax-report submission phase (see section 9.1 for more details).

In eCommerce malicious acts involve all types of cheating in payments, click fraud and any type of reputation credential forgeries. Depending on the case, cheating in payments may result either in transaction failure or in revocation of misbehaving party's anonymity and/or traceability of his transactions (see table 11.2). Click fraud results in detection of fraudulent clicks and the corresponding accounting procedures and/or in blacklisting of the

---

he trusts the merchant) as it is conceivable how in such cases anonymity of his subscription or account is endangered.

person having performed it. Misbehavior in transaction evaluation is related to misuse of the evaluation tokens and may result either in anonymity revocation of the malicious person involved or to the negative rating of the latter.

It is apparent that not all misbehaviors are equivalent. Others may be the result of a mistake and considered minor, i.e., attempt to overuse one's account balance or act against a website's policy, while others e.g., cheating in taxes, constitute an indication of possible future repetition of similar behaviors from that user. Two levels of blacklistability of users are supported in our authentication protocols, local and global and facilitate the distinction between the two cases of misbehavior. Minor misbehaviors or of local importance may result in a local blacklisting of user, while global blacklisting one would imply serious misconduct.

11.3 Compartmentalization

Unlinkability of users' activities immediately provides a notion of our architecture's compartmentalization, i.e., how the components of our architecture behave when one of them is compromised, fails to be applied or is not applied at all.

Anonymous Browsing. Anonymous browsing is a prerequisite for all the online operations. It is conceivable that, if anonymous browsing is not supported, the network-level information which is emitted in every online activity of a user would act as a strong identifier for the latter and thus enable accurate user-profiling.

Privacy-preserving Online Ads. Assuming that no privacy-preserving technique is applied on online ads, user-profiling would be enabled. Nevertheless, as long as payments are anonymous, any occurring purchase will simply strengthen the accuracy of the generated profile without implications in anonymity. A very extreme case where such a profiling may lead in privacy-compromise in banking is the following: a user, whose advertising profile has been very accurately constructed, participates in two or more transactions using distinct account numbers; if all accounts are anonymous, the former scenario would violate account unlinkability, whereas if one of the payments is not anonymous, account anonymity

is compromised. It is important, however, to note that as this threat requires extended collaboration of multiple entities that are not directly financially dependent, i.e., banks, many merchants, ad-networks, it is considered in principal out of scope.

Privacy-preserving Payments. Anonymous payments is important for the privacy provided in the entire chain of eCommerce. Evidently, if payments are anonymous but linkable, i.e., if a user uses the same anonymous accounts for all his transactions, accurate user-profiling can be done by the bank which holds the account and merchants from which the user makes frequent purchases. Even worse, If payments are non-anonymous, then these profiles are linked to particular identities by all entities and no sense of privacy is applied. In our protocols, privacy- preserving offline payments are secure under the security of the underlying ecash schemes. However, if the user's bank-related secret is compromised, the adversary will be able to trace all users' offline transactions.

Privacy-preserving Product Delivery. If we are aiming for privacy w.r.t. merchants, this set of protocols can be effectively applied only if anonymous payments are supported. Interchangeably, when the product has not been bought anonymously our method can be used to protect consumers' privacy against curious delivery companies, only if the former trusts the merchant not to share information with the delivery company. Protection of privacy in the delivery of online purchases is essential for privacy in eCommerce. A regular (non-privacy-preserving) delivery method would compromise payment privacy w.r.t. non-trusted merchants who collaborate with the delivery company: the delivery information could successfully play the role of consumer's identifier and thus transactions become linkable. In the extreme case where merchants and the bank collude and delivery information includes the identity of the recipient, account privacy may be violated. [3]

Privacy-preserving Transaction Evaluation. Anonymity of a transaction evaluation procedure and unlinkability of ratings that correspond to the same identity is crucial for the

[3]Of course, the latter is out of our threat model as we consider that entities may combine their information with entities they collaborate directly.

anonymity and unlinkability of the corresponding transactions: linking two ratings would directly imply linking of the corresponding transactions and their payment information. Taking the extreme case of adversary described in the previous paragraph, the latter scenario may directly associate the bank accounts which were used for the two online payments.

Privacy-preserving Banking. Privacy in banking is crucial for privacy in transactions. More specifically, without privacy-preserving payments, transactions become linkable to particular identities and to other transactions of the same individual. On the other hand, assuming that the set of protocols presented in chapter 9 is applied, privacy is very carefully preserved: revoking anonymity of a single account does not affect the privacy of the rest; revoking anonymity of a credit card-based payment does not reveal anything for the rest of the transactions of the same credit card as long as they do not refer to the same account.

Privacy-preserving Authentication. Assume that a user has registered in many organizations using the authentication protocols presented in chapter 5 and has his identity revealed in one of them. Because of the unlinkability property of authentication transcripts, this will not affect user's privacy in the rest of his subscriptions. If we further assume that there is no privacy-preserving authentication scheme adopted, then our privacy provisions may be affected as follows:

- In online subscriptions, in cases of organizations that require strong accountability, users will have to use traditional and non-privacy preserving authentication for the base of their registration.
- In banking nothing will change, as we already assumed that strong authentication information is used as the base for a user's membership.
- In eCommerce, if proof of attribute ownership is required, i.e., proof of age, proof of owning a driving license etc., privacy will not be supported. In cases of transactions where no strong authentication is required there will be no effect: in transaction evaluation system, user strong identification is required already for users to be able to participate in it; in payment only account-authorization information is required.

In addition, when a user is globally blacklisted, he will no longer be able to open new subscriptions in strongly accountable organizations. His existing memberships in such organizations will be deactivated in the following memberships' validity check of each organization. Local blacklisting does not affect other memberships of the same individual.
Having already discussed the failure-implications of various parts of our architecture to the rest, we proceed to exhibit our architecture limitations.

11.4 Limitations

In the threat model we adopted for the entire architecture (see chap 3) the banks, the registration and tax authorities as well as any financial organization is assumed to be honest but curious. More specifically, aiming to increase their profit and thus to maintain their clientele, banks and financial organizations, e.g., commercial websites or online services, are motivated to be honest in their functional operations. On the other hand, as they are financially incentivized to construct consumers' profiles, we assumed that they are curious and may share their legally obtained information to link transactions to consumers. In terms of collaboration we assumed that collusions my take place only between financially dependent entities and to a limited depth. Thus we assumed that merchants may share information regarding their clients with the banks where they keep accounts at, delivery companies may share information with the merchants they are paid from, ad-networks and publishers share information with the advertisers they are in agreement with. Similar assumptions we have adopted for tax and registration authorities which are part of the government and considered honest w.r.t. their functional operations. We also tentatively assume that the former are curious, to represent unscrupulous individuals who are always tempted to sell information they have access to.

Unfortunately, banks and tax authorities are not always honest w.r.t. their functional operations. A very representative example which is not included in our threat model is when the owner of the bank owns an anonymous bank account. Bank is then malicious and may issue valid tax reports for fake tax numbers. In a similar way, tax authority employees may be bribed to accept forged tax reports. Our architecture is not secure against such behaviors

which may be represented by a threat model with malicious banks and tax authorities.

Another example of collusion our architecture does not protect privacy against is the one involving the registration authority with all the organizations. As the registration authority maintains data regarding users' attributes, the former's collaboration with organizations that require demonstration of the same attribute would be able to group together registrations of the same user. Although such a collusion would enable users' profiling, it would not link that profile to particular identities. In addition, a corrupted registration authority would enable the issuing of fake identities with disastrous ramifications to all activities supported in our design. Distribution of power of identity card issuing seems to be a very reasonable measure towards dealing with corrupted authorities with —nevertheless — not few implications to privacy.

In eCommerce, as mentioned in previous chapters there are some restrictions to user-behavior for the eCommerce system to support privacy. In particular, a user who clicks more than a predefined number of times on an ad, has his clicks on that ad monitored, while if he attempts to hide his clicks his identity in the ad system is recovered. In payments, transactions' unlinkability is strongly related to the number of accounts each user uses for its online purchases.

Applying privacy in the online world is a rather complicated procedure. Evidently, there are many privacy functionalities that are not supported by our protocols and would probably constitute challenging problems for future work. Closing an anonymous account or renewing its authorization credentials and update the tax information accordingly is one. Realization of shared anonymous accounts, account transferring system with measures against money laundering are some of them. Fair and immediate update of the tax information are some of the challenges that apply in those cases as well.

Table 11.1: Privacy Provisions Overview. A = Anonymity, NP = Non profiling, P = Profiling, [x,y]: possible x-y collusion

Service	**Privacy Provisions**	**Adversarial Assumptions**
Browsing (chap. 6 – PAR)	A&NP	Local adversary
Online Ads (chap. 7 – PPOAd)	A&NP iff clicks on ads¡M& A&P* iff clicks on ads>M	[Honest but curious Admin, advertisers, malicious publishers]
Online Payment (chap. 8 – ACCs) (chap. 9 – Anon. Accounts)	A& user-dependent NP	[Honest but curious banks, merchants]
Account Taxation (chap. 9)	A& user-dependent NP maintenance	[Honest but curious banks, tax authority]
Offline Payment (chap. 8 – ACCs)	A&NP	[Honest but curious banks, merchants]
Product Delivery (chap. 8 – APOD)	A&NP	[Honest but curious DCs, MSs, merchants]
Transaction Evaluation (chap. 8 – RepSys)	A&NP maintenance	[Honest but curious users, reputation Admin]
Credit Score Calculation (chap. 9)	A&NP maintenance	[Honest but curious banks, credit bureau]
Authentication Mechanism (chap. 5)	A&NP	Honest but curious registration authority, [Organizations]

Table 11.2: Privacy Provisions Overview. A = Anonymity, NP= Non profiling

Service	**Privacy Provisions**	**Adversarial Assumptions**
Browsing (PAR)	Cheat at payment	No service
Online Ads	Click fraud	Partial profiling, No charge
Payment (ACC) *Payment* (Anon. Accounts)	Supersede credit limit Supersede account balance	No processing of payment Anonymity revocation
Account Taxation	Exchange Tax reports, Lie for taxes	Detection - Fine
Product Delivery (APOD)	Cheat at payment	Product Delivery Delivery failure
Transaction Evaluation (RepSys)	Reputation token misuse Avoid negative reputation	Anonymity revocation Negative reputation
Authentication Mechanism	Impersonation attack Malicious actions	Authentication failure Local/Global blacklisting

Part IV

Conclusions

Chapter 12

Conlcusions

Privacy is an undeniable right of every individual and security is a requirement for the proper operation of financial systems that want to attract individuals. Currently it is often the case where the need for accountability and fairness oversees any attempt of privacy application as it seems to impedes the security mechanisms. Evidently, assuming a user who often makes online purchases and has subscribed to multiple online organizations, banks have a very accurate view of his transactions (through credit card payments) and overall financial activity (through his credit history), advertisers and ad networks may approach in detail all of the user's subscription and transaction preferences. Individuals' privacy is consequently hardly compromised under the umbrella of severe need for security: banks and other financial institutions are justified by risk management, ad networks and advertisers are legitimized through click fraud detection mechanisms. Unfortunately, privacy violations are prone to be more acute, as there are still cases of online systems where security is crucial and has not been applied yet: in governmental systems strong authentication seems to be a necessity to enforce proper access control. Combining the two concepts of privacy and security against a real-world–based adversarial model to derive a new and more practical definition for privacy as well as the latter's application in many common activities of individuals are the the main contributions of this thesis.

Previous work on online privacy and security has shown that these two concepts are hard to be combined: systems who protect individuals' privacy tend not to offer any sense of user-accountability, while strictly accountable systems fail to protect. In response, we

suggested an identity management system that offers both accountability and privacy, where privacy is offered as an option. Users may interact with other individuals and register to multiple organizations achieving various degrees of privacy depending on each user's privacy preferences and organizations' standards. The real-world nature of the assumptions we make in this architecture, the variety in the types of organizations and the privacy-levels achieved system as well as the notion of privacy as an option are the main innovations of this work.

Real-world was incorporated in our architecture in the form of our threat model, requirements and applicability properties. More specifically, in our threat model we adopted the real-world assumption that all parties are motivated to maximize their profit. Thus, we consider banks, anonymous delivery company, privacy-preserving online-ad authority honest but curious as they are all motivated to maintain their clientele. We designed our architecture to offer most functionalities that existing systems offer such as credit card payments in banking, taxation mechanisms, credit scoring, which is necessary for any case of our protocols' application. In many cases we offered monetary incentive for entities' proper behavior. Regardless of how realistic such a threat model can be, one would foresee here many improvements for this work. One would be the one considering banks able to attempt any type of forgeries w.r.t. taxation or payments, as in the cases where user and bank (bank owner) are the same person. It has been stated before that against such a threat model our system's security fails. Tax authority and users' collaboration may be another case of collusion that is not present in our threat model. Although we could restrict such attacks by imposing that bank owners do not hold anonymous bank accounts in their banks and that tax authority employees deposit their tax reports in other tax offices, nevertheless, the aforementioned collusions tend to be very common nowadays in many countries and finding solutions towards them would be a challenging problem to work on.

It is conceivable that as the online and the offline world converge, the more critical these two concepts become for the individuals' life. In this thesis we dealt with privacy and security issues in eBanking, eCommerce and employment-based access control issues. However, there are fields where privacy-security combination is important and where our identity management system can be adopted. EHealth is a very representative example of such system as individuals need to have their medical record (credit score) updated without

unauthorized entities obtaining access to it, receive funding for operations that have to be kept confidential or buy medicines for treatments they do not want to be public.

A secondary contribution of this work is the careful examination of the variety of uses of existing crypto primitives (see chapter 4), i.e., digital cash, anonymous credentials, blind signatures, zero knowledge proofs of knowledge, etc., which we combined in various ways. Based on the security definitions and properties of the primitives, we built the security models and proofs of our protocols. More specifically, we made use of ecash to instantiate reputation tokens in our reputation system, authorization tokens in our advertising system and registration protocols and authentication tokens in banking. In the tax reporting case we built our own primitive to guarantee fair calculation of taxes withheld by all users' anonymous accounts. This same protocol we applied in the credit score calculation case.

Part V

Bibliography

Bibliography

[Akagi *et al.*, 2008] Norio Akagi, Yoshifumi Manabe, and Tatsuaki Okamoto. An efficient anonymous credential system. In *Financial Cryptography and Data Security: 12th International Conference*, pages 272–286, Berlin, Heidelberg, 2008. Springer-Verlag.

[Andministration,] International Trade Andministration. Defining the electronic commerce sector.

[Asokan *et al.*, 1999] N. Asokan, P. Janson, M. Steiner, and M. Waidner. State of the art in electronic payment systems. *IEEE Computer*, 30:28–35, 1999.

[Aura, 2005] T. Aura. Cryptographically generated addresses (cga), rfc3972, 2005.

[Back *et al.*, 2001] Adam Back, Ian Goldberg, and Adam Shostack. Freedom systems 2.1 security issues and analysis. White paper, Zero Knowledge Systems, Inc., May 2001.

[BBC, 2009] BBC. Q&A: Identity cards, 2 July 2009.

[Belenkiy *et al.*, 2007] Mira Belenkiy, Melissa Chase, C. Christopher Erway, John Jannotti, Alptekin Küpçü, Anna Lysyanskaya, and Eric Rachlin. Making p2p accountable without losing privacy. In *WPES*, pages 31–40, 2007.

[Bellare *et al.*, 2000] M. Bellare, J. Garay, R. Hauser, H. Krawczyk, A. Herzberg, G. Tsudik, E. van Herreweghen, M. Steiner, Gene Tsudik, and M. Waidner. Design, implementation and deployment of the ikp secure electronic payment system. *IEEE Journal on Selected Areas in Communications*, 18:611–627, 2000.

[Bhattacharjee and Goel, 2005] Rajat Bhattacharjee and Ashish Goel. Avoiding ballot stuffing in ebay-like reputation systems. In *P2PECON*, pages 133–137, 2005.

[Boucher *et al.*, 2000] Philippe Boucher, Adam Shostack, and Ian Goldberg. Freedom systems 2.0 architecture. White paper, Zero Knowledge Systems, Inc., December 2000.

[Boudot, 2000] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *EUROCRYPT*, pages 431–444, 2000.

[Brands, 1995] Stefan Brands. Electronic cash on the internet. In *Proceedings of the Symposium on the Network and Distributed System Security*, 1995.

[Brands, 2000] Stefan A. Brands. *Rethinking public key infrastructures and digital certificates: building in privacy.* Cambridge, Mass. : MIT Press, 2000.

[Brickell *et al.*, 1988] Ernest F. Brickell, David Chaum, Ivan Damgård, and Jeroen van de Graaf. Gradual and verifiable release of a secret. In *CRYPTO '87: A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology*, pages 156–166, London, UK, 1988. Springer-Verlag.

[Camenisch and Lysyanskaya, 2001] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *Advances in Cryptology - EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118. Springer-Verlag, 2001.

[Camenisch and Lysyanskaya, 2002a] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *International Conference on Security in Communication Networks – SCN*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer Verlag, 2002.

[Camenisch and Lysyanskaya, 2002b] Jan Camenisch and Anna Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO '02: Proceedings of the 22nd Annual International Cryptology Conference on Advances in Cryptology*, pages 61–76, London, UK, 2002. Springer-Verlag.

[Camenisch and Michels, 1999] Jan Camenisch and Markus Michels. Separability and efficiency for generic group signature schemes. In *CRYPTO '99: Proceedings of the 19th*

Annual International Cryptology Conference on Advances in Cryptology, pages 413–430, London, UK, 1999. Springer-Verlag.

[Camenisch and Stadler, 1997] Jan Camenisch and Markus Stadler. Effcient group signature schemes for large groups. In *Advances in Cryptology — CRYPTO '97*, volume 1296 of *Lecture Notes in Computer Science*, pages 410–424. Springer-Verlag, 1997.

[Camenisch and Van Herreweghen, 2002] Jan Camenisch and Els Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 21–30, New York, NY, USA, 2002. ACM.

[Camenisch *et al.*, 2005] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 302–321. Springer-Verlag, 2005.

[Camenisch *et al.*, 2006] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Balancing accountability and privacy using e-cash (extended abstract). In *Security and Cryptography for Networks*, 2006.

[Chatterjee, 2008] Koushik Chatterjee. System and method for anonymous mail delivery services, 2008.

[Chaum and Pedersen, 1993] David Chaum and Torben P. Pedersen. Wallet databases with observers. In *CRYPTO '92: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 89–105, London, UK, 1993. Springer-Verlag.

[Chaum *et al.*, 1988a] David Chaum, Jan-Hendrik Evertse, and Jeroen van de Graaf. An improved protocol for demonstrating possession of discrete logarithms and some generalizations. In *Advances in Cryptology EUROCRYPT 87*, 1988.

[Chaum *et al.*, 1988b] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *Proceedings of CRYPTO '88*, 1988.

[Chaum, 1981] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 1981.

[Chaum, 1991] David Chaum. Zero-knowledge undeniable signatures. In *Advances in Cryptology EUROCRYPT '90*, 1991.

[Choi *et al.*, 2005] Eun Young Choi, Hyun-Jeong Kim, and Dong Hoon Lee. Efficient member revocation in group signature schemes. In *TrustBus*, pages 195–205, 2005.

[Claessens *et al.*, 2003] Joris Claessens, Claudia Diaz, Raimondo Faustinelli, and Bart Preneel. Secure and privacy-preserving web banner system for targeted advertising, 2003.

[Clarke *et al.*, 2001] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *International Workshop on Design Issues in Anonymity and Unobservability*, 2001.

[CO, 1999 2007] Continental Relay CO. Protect your privacy with your own offshore private maildrop, 1999-2007.

[D. Chaum and Naor, 1990] A. Fiat D. Chaum and M. Naor. Untraceable electronic cash. In *Crypto '88*, 1990.

[Danezis and Serjantov, 2004] George Danezis and Andrei Serjantov. Statistical disclosure or intersection attacks on anonymity systems. In *Information Hiding*, pages 293–308, 2004.

[Dierks and Allen, 1999] T. Dierks and C. Allen. The tls protocol version 1.0, 1999.

[Dingledine *et al.*, 2003] Roger Dingledine, Nick Mathewson, and Paul Syverson. Reputation in p2p anonymity systems. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.

[Dingledine *et al.*, 2004] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[Figueiredo *et al.*, 2003] Daniel R. Figueiredo, Jonathan K. Shapiro, and Don Towsley. Using payments to promote cooperation in anonymity protocols, 2003.

[Foundation,] The Eclipse Foundation. Higgins: Open source identity framework.

[Foundation and publish vision for open government through open identity technologies,] OpenID Foundation and Information Card Foundation publish vision for open government through open identity technologies. Openid foundation.

[Franz *et al.*, 1998] Elke Franz, Anja Jerichow, and Guntram Wicke. A payment scheme for mixes providing anonymity. In *TREC '98: Proceedings of the International IFIP/GI Working Conference on Trends in Distributed Systems for Electronic Commerce*, pages 94–108, London, UK, 1998. Springer-Verlag.

[Goldschlag *et al.*, 1996] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding routing information. In Ross Anderson, editor, *Workshop on Information Hiding*, pages 137–150. Springer-Verlag, May 1996. LLNCS 1174.

[Group,] PRIME Reference Group. Prime project.

[Gupta *et al.*, 2003] Minaxi Gupta, Paul Judge, and Mostafa Ammar. A reputation system for peer-to-peer networks. In *NOSSDAV*, 2003.

[Hayes, 1990] Barry Hayes. Anonymous one-time signatures and flexible untracable electronic cash. In *AusCrypt '90: A Workshop on Cryptology, Secure Communication and Computer Security*, January 1990.

[Hsu,] Spencer S. Hsu. Senate democrats address immigration.

[Jakobsson and Yung, 1996] Markus Jakobsson and Moti Yung. Revokable and versatile electronic money (extended abstract). In *CCS '96: Proceedings of the 3rd ACM conference on Computer and communications security*, pages 76–87, New York, NY, USA, 1996. ACM.

[Jakobsson *et al.*, 1999] Markus Jakobsson, Philip D. MacKenzie, and Julien P. Stern. Secure and lightweight advertising on the Web. *Computer Networks (Amsterdam, Netherlands)*, 31(11–16):1101–1109, 1999.

[Jiang *et al.*, 2003] Zhimei Jiang, Hui Luo, Yu-Ngok Li, and Henry P. icard - foundation for a new ubiquitous computing architecture. In *ICC '03: Proceedings of the IEEE International Conference on Communications, 2003.*, pages 1211– 1217, 2003.

[Johnson *et al.*, 2007] Peter C. Johnson, Apu Kapadia, Patrick P. Tsang, and Sean W. Smith. Nymble: Anonymous ip-address blocking. In *Privacy Enhancing Technologies*, pages 113–133, 2007.

[Juels, 2001] Ari Juels. Targeted advertising ... and privacy too. In *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, pages 408–424, London, UK, 2001. Springer-Verlag.

[Kent and Millett, 2002] Stephen T. Kent and Lynette I. Millett, editors. *IDs—Not That Easy: Questions About Nationwide Identity Shystems.* National Academies Press, 2002.

[Kent and Millett, 2003] Stephen T. Kent and Lynette I. Millett, editors. *Who Goes There? Authentication Through the Lens of Privacy.* National Academies Press, 2003.

[Kesdogan *et al.*, 2006a] Dogan Kesdogan, Dakshi Agrawal, Vinh Pham, and Dieter Rautenbach. Fundamental limits on the anonymity provided by the mix technique. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2006.

[Kesdogan *et al.*, 2006b] Dogan Kesdogan, Dakshi Agrawal, Vinh Pham, and Dieter Rautenbach. Fundamental limits on the anonymity provided by the mix technique. In *Securuty and Privacy*, pages 86–99, 2006.

[Kinateder and Pearson, 2003] Michael Kinateder and Siani Pearson. A privacy-enhanced peer-to-peer reputation system. In *EC-Web*, pages 206–215, 2003.

[Kinateder and Rothermel, 2003] Michael Kinateder and Kurt Rothermel. Architecture and algorithms for a distributed reputation system. In *iTrust*, pages 1–16, 2003.

[Kinateder *et al.*, 2005] Michael Kinateder, Ralf Terdic, and Kurt Rothermel. Strong pseudonymous communication for peer-to-peer reputation systems. In *SAC*, pages 1570–1576, 2005.

[Krawczyk, 1999] Hugo Krawczyk. Blinding of credit card numbers in the set protocol. In *FC '99: Proceedings of the Third International Conference on Financial Cryptography*, pages 17–28, London, UK, 1999. Springer-Verlag.

[Krishnamurthy and Wills, 2006] Balachander Krishnamurthy and Craig E. Wills. Generating a privacy footprint on the internet. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 65–70, New York, NY, USA, 2006. ACM.

[Low *et al.*, 1994] Steven H. Low, Sanjoy Paul, and Nicholas F. Maxemchuk. Anonymous credit cards. In *CCS '94: Proceedings of the 2nd ACM Conference on Computer and communications security*, pages 108–117, New York, NY, USA, 1994. ACM.

[Low *et al.*, 1996] S. Low, N. F. Maxemchuk, and S. Paul. Anonymous credit cards and its collusion analysis. *IEEE Transactions on Networking*, December 1996.

[Lysyanskaya and Ramzan, 1998] Anna Lysyanskaya and Zulfikar Ramzan. Group blind digital signatures: A scalable solution to electronic cash. In *In Financial Cryptography (FC)*, pages 184–197. Springer-Verlag, 1998.

[Lysyanskaya *et al.*, 1999] Anna Lysyanskaya, Ronald Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In *Selected Areas in Cryptography '99*, volume 1758 of *Lecture Notes in Computer Science*, pages 184–199. Springer-Verlag, 1999.

[members, 2008] Commission members. Securing cyberspace for the 44th presidency, 2008.

[Micali and Rivest, 2002] Silvio Micali and Ronald L. Rivest. Micropayments revisited. In *CT-RSA*, pages 149–163, 2002.

[Nakanishi and Funabiki, 2006] Toru Nakanishi and Nobuo Funabiki. Group signature schemes with membership revocation for large groups. *IEICE Transactions*, 89-A(5):1275–1283, 2006.

[Okamoto, 2006] Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In *TCC*, pages 80–99, 2006.

[Øverlier and Syverson, 2006] Lasse Øverlier and Paul Syverson. Locating hidden servers. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2006.

[Paillier and Pointcheval, 1999] Pascal Paillier and David Pointcheval. Efficient public-key cryptosystems provably secure against active adversaries. In K. Y. Lam and E. Okamoto, editors, *Advances in Cryptology ASIACRYPT 99*, volume 1716, page 165179. Springer-Verlag, Lecture Notes in Computer Science, 1999.

[Pavlov *et al.*, 2004] Elan Pavlov, Jeffrey S. Rosenschein, and Zvi Topol. Supporting privacy in decentralized additive reputation systems. In *iTrust*, pages 108–119, 2004.

[Reed *et al.*, 1998] Michael G. Reed, Paul F. Syverson, and David M. Goldschlag. Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, 16(4), 1998.

[Reiter *et al.*, 2005] Michael K. Reiter, XiaoFeng Wang, and Matthew Wright. Building reliable mix networks with fair exchange. In *ACNS*, pages 378–392, 2005.

[Rivest, 2004] R. Rivest. Peppercoin micropayments. In *Proceedings of Financial Cryptography '04*, volume 3110, pages 2–8, February 2004.

[Rudolf,] Carsten Schmidt Rudolf. A framework for micropayment evaluation.

[Shamir, 1979] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[Shirey, 2000] R. Shirey. Network dictionary rfc2828, 2000.

[Shouhuai Xu and Zhang, 2000] Moti Yung Shouhuai Xu and Gendu Zhang. Scalable, tax evasion-free anonymous investing, 2000.

[Smith and Shao, 2007] Rhys Smith and Jianhua Shao. Privacy and e-commerce: a consumer-centric perspective. *Electronic Commerce Research*, 7(2):89–116, 2007.

[Smith, 2001] Jonathan M. Smith. Iprivacy white paper, 2001.

[Steinbrecher, 2006] Sandra Steinbrecher. Design options for privacy-respecting reputation systems within centralised internet communities. In *SEC*, pages 123–134, 2006.

[Stolfo and Smith, 2001] Salvatore J. Stolfo and Jonathan M. Smith. Method and system for user defined filtering of communications to anonymous users in a computer network, 2001. U.S. patent WO/2001/065442.

[Syverson *et al.*, 1997] Paul F. Syverson, Stuart G. Stubblebine, and David M. Goldschlag. Unlinkable serial transactions. In *Proceedings of Financial Cryptography*, pages 39–55. Springer-Verlag, 1997.

[Szerwinski and Guñeysu, 2008] Robert Szerwinski and Tim Guñeysu. Exploiting the power of GPUs for asymmetric cryptography. In *Proceedings of the 10th International Workshop on Cryptographic Hardware and Embedded Systems (CHES, 2008)*, August 2008.

[Toubiana *et al.*, 2009] Vincent Toubiana, Arvind Narayanan, Dan Boneh, Helen Nissenbaum, and Barocas Solon. Priveads: Privacy preserving targeted advertising, 2009. http://crypto.stanforf.edu/privads/.

[Tsang *et al.*, 2007] Patrick P. Tsang, Man Ho Au, Apu Kapadia, and Sean W. Smiths. Blacklistable anonymous credentials: blocking misbehaving users without ttps. In *CCS '07: Proceedings of the 14th ACM conference on Computer and communications security*, pages 72–81, New York, NY, USA, 2007. ACM.

[Tunstall, 1989] J.S. Tunstall. Electronic currency. In *Proceedings of the IFIP WG 11.6 International Conference*, October 1989.

[Turow *et al.*, 2009] Joseph Turow, Jennifer King, Chris Jay Hoofnagle, Amy Bleakley, and Michael Hennessy. Americans reject tailored advertising and three activities that enable it, September 2009. `http://ssrn.com/abstract=1478214`.

[Voss *et al.*, 2005] Marco Voss, Andreas Heinemann, and Max Muhlhauser. A privacy preserving reputation system for mobile information dissemination networks. In *SECURECOMM*, pages 171–181, 2005.

[Voss, 2004] Marco Voss. Privacy preserving online reputation systems. In *International Information Security Workshops*, pages 245–260, 2004.

Part VI

Appendices

Appendix A

Anonymous Credit Cards: Security Analysis

Security is a principal requirement in our system. In this section, we provide strict definitions of our system's operations and we base on them to define our system's security properties. Some of the definitions presented in this section are inspired by previous work on other primitives, such as [Camenisch and Lysyanskaya, 2001; Camenisch *et al.*, 2005].

In what follows, we assume the typical Cardholder-Merchant-Bank system architecture.

A.1 Operations

When an operation is an interactive procedure (or a protocol consisting of multiple procedures) between two entities C and B, we denote it by $\langle O_{\mathsf{C}}, O_B \rangle \leftarrow \mathsf{Pro}(I_{\mathsf{C},\,B})[\mathsf{C}(I_{\mathsf{C}}),\ B(I_B)]$, where Pro is the name of the procedure (or protocol). O_{C} (resp. O_B) is the private output of C (resp. B), $I_{\mathsf{C},\,B}$ is the common input of both entities, and I_{C} (resp. I_B) is the private input of C (resp. B). We also note that depending on the setup, some operations may require additional global parameters (i.e. some common parameters for efficient zero-knowledge proofs, a modulus p, etc). Our system will need these additional parameters only when using underlying schemes that use such parameters, e.g., ecash systems or blind group signatures. To simplify notation, we omit these potential global parameters from the inputs to all the operations.

- $(pk_{\mathsf{B}}, sk_{\mathsf{B}}) \leftarrow \mathsf{Bkeygen}(1^k)$ is the key generation algorithm for Bank B. We denote by pk_{B} all public information regarding B, namely her public ([group]blind) signature and encryption keys and by sk_{B} the overall secret information of B.
- $(pk_{\mathsf{C}}, sk_{\mathsf{C}}) \leftarrow \mathsf{Ckeygen}(1^k)$ is the key generation algorithm for cardholders. We denote by pk_{C} all public information regarding a cardholder C, namely her public signature and encryption keys and by sk_{C} the overall secret information of C.
- $\langle (W_p, W_{id}), (T_{p,id}, D_{\mathsf{credit}}{}') \rangle / \langle \perp, \perp \rangle \leftarrow$

$$\leftarrow \mathsf{AnonymousCreditCardIssue}(pk_{\mathsf{B}}, pk_{\mathsf{C}}, \mathsf{L}_{\mathsf{credit}})[\mathsf{C}(sk_{\mathsf{C}}),\ \mathsf{B}(sk_{\mathsf{B}},\ D_{\mathsf{credit}},\ D_{\mathsf{debit}})]$$

  A customer Cissues an ACC of credit limit $\mathsf{L}_{\mathsf{credit}}$. value ecoins in the form of two wallets W_p and W_{id} from Bank B. Bank, using D_{debit}, checks if C is eligible for that. If so, ecash withdrawal procedure procedure is carried out for C to acquire two $\mathsf{L}_{\mathsf{credit}}$ valued wallets W_p and W_{id}. B's $D_{\mathsf{credit}}$is updated accordingly while B maintains $T_{p,id}$ to trace double-spenders in case of emergency.
- $\langle (W'_p,\ W'_{id}), (S_p,\ \pi_p,\ S_{id},\ \pi_{id}) \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{OfflinePayment}(pk_{\mathsf{M}},\ pk_{\mathsf{B}},\ T_{det})\ [C(W_p, W_{id}, sk_{\mathsf{C}}),\ \mathsf{M}(sk_{\mathsf{M}})]$. Customer C, using her ACC spends V_p value from each wallet to M. Spent ecoins are pairs of (S, π), where S is a serial number and π is the proof of its validity. For convenience, we will denote with $(S_p,\ \pi_p)$ and $(S_{id},\ \pi_{id})$ the set of serial numbers and spending-proofs of ecoins spent from W_p and W_{id} wallets, respectively.
- $\langle \top, (D_{\mathsf{debit}}{}', D_{\mathsf{hist}}{}') \rangle / \langle \perp, \perp \rangle \leftarrow \mathsf{MerchantPaymentDeposit}\ (pk_{\mathsf{M}}, pk_{\mathsf{B}})\ [\mathsf{M}(sk_{\mathsf{M}},\ S_{p,id},\ \pi_{p,id}),\ \mathsf{B}(sk_{\mathsf{B}},\ D_{\mathsf{debit}},\ D_{\mathsf{hist}})]$, where for convenience

$$(S_{p,id},\ \pi_{p,id}) = (S_p,\ \pi_p) \cup (S_{id},\ \pi_{id}).$$

  A merchant M deposits the ecoins he has received throughout his selling activity. If the ecoins deposited $(S_{p,id},\ \pi_{p,id})$ are valid and not double-spent, Bank B updates the entry of M in its debit database D_{debit}. Deposited ecoins are then stored in the history database D_{hist}.
- $\langle (W'_{p,id},\ \alpha_{\mathsf{C}}(m)), (S_{p,id},\ \pi_{p,id},\ D_{\mathsf{anon}}{}') \rangle / \langle \perp, \perp \rangle \leftarrow$

$$\leftarrow \mathsf{AnonymousAccountSetup}(pk_{\mathsf{B}},\ M_{ot},\ m,\ R_{ot})[\mathsf{C}(W_{p,id},\ sk_{\mathsf{C}}), \mathsf{B}(sk_{\mathsf{B}},\ D_{\mathsf{anon}})].$$

  In this procedure, C interacts through her ACC with Bank B to setup an anonymous account for online transactions. Both parties know B's public information (pk_{B}), the new account's balance M_{ot}, m and R_{ot}. The main output is the new D_{anon} entry for both parties.
- $\langle \top, \mathsf{Paym}_{\mathsf{CIB}},\ (\ \mathsf{D}'_{\mathsf{anon}},\ \mathsf{Rec}_{\mathsf{T}}) \rangle / \langle \mathsf{Rej}_{\mathsf{CIB}},\ \mathsf{Rej}_{\mathsf{CIB}},\ D_{\mathsf{anon}}{}' \rangle \leftarrow$

$$\leftarrow \mathsf{OnlinePayment}(pk_{\mathsf{G}},\ pk_{\mathsf{CIB}},\ V_p)[C(\alpha_{\mathsf{C}}(x)),\ \mathsf{G}(sk_{\mathsf{G}}),\ \mathsf{CIB}(\mathsf{D}_{\mathsf{anon}}, sk_{\mathsf{CIB}})].$$

In this procedure, C uses her anonymous account information to pay for the purchase of a product of value V_p. The CIB-related outputs are the updated D_{anon} and the transaction receipt, $\mathsf{Rec_T}$. Gateway G receives the payment $\mathsf{Paym_{CIB}}$, while if something goes wrong only CIB's D_{anon}changes.

- $\langle \top, \top,\ D_{\mathsf{debit}}{}' \rangle / \langle \bot, \bot,\ \bot \rangle \leftarrow \mathsf{OnlinePaymentDeposit}(pk_{\mathsf{G}},\ pk_{\mathsf{CIB}},\ pk_{\mathsf{AB}})$ $[\mathsf{G}(sk_{\mathsf{G}},\ \mathsf{Paym_{CIB}}),\ \mathsf{CIB}(sk_{\mathsf{CIB}}),\ \mathsf{AB}(D_{\mathsf{debit}},\ sk_{\mathsf{AB}})]$. Gdeposits $\mathsf{Paym_{CIB}}$ to M's AB, AB. AB, G and C's CIB, CIB, collaborate for M's debit account to be updated (D_{debit}').
- Identify operation consists of two suboperations:
  - $(pk_{\mathsf{C}}, \Pi_p^G) / \bot \leftarrow \mathsf{Identify_p}(S_p, \pi_p^1, \pi_p^2)$ and
  - $(pk_{\mathsf{C}}, \Pi_{id}^G) / \bot \leftarrow \mathsf{Identify_{id}}(S_{id}, \pi_{id}^1, \pi_{id}^2)$.

  If an ecoin from W_p (or W_{id}) is double-spent, with (S^p, π_p^1) and (S_p, π_p^2) (or (S_p, π_p^1) and (S_p, π_p^2)), Bank can find the customer who double-spent the ecoin with serial S_p (S_{id}) using $\mathsf{Identify_p}$ (or $\mathsf{Identify_{id}}$). Π_p^G (Π_{id}^G) is a proof that pk_{C} double-spent the ecoin with the serial number S_p (S_{id}).
- VerifyGuilt operation also consists of two suboperations, one for each wallet in the anonymous card:
  - $\top / \bot \leftarrow \mathsf{VerifyGuilt_p}(S_p,\ \Pi_p^G,\ pk_{\mathsf{C}})$ and
  - $\top / \bot \leftarrow \mathsf{VerifyGuilt_{id}}(S_{id},\ \Pi_{id}^G,\ pk_{\mathsf{C}})$.

  Both operations output $\top$ if the customer C (represented by pk_{C}) double-spent S_p from W_p and the corresponding ecoin S^{id} from wallet W_{id}. Π_p^G and Π_{id}^G are the proofs of guilt of C, i.e. the outputs from the preceding Identify procedure.
- $\langle S_{id}^i,\ \Pi_{id}^i \rangle \leftarrow \mathsf{Trace_{id}}(\mathsf{S_{id}},\ pk_{\mathsf{C}},\ \Pi_{\mathsf{id}}^{\mathsf{C}},\ D_{\mathsf{hist}},\ \mathsf{L})$, where $i = 1 \ldots L$. This algorithm first checks whether $\mathsf{VerifiyGuilt_{id}}(S_{id},\ \Pi_{id}^G,\ pk_{\mathsf{C}})$ accepts. If yes, the procedure outputs all L ecoins $(S_{id}^i, i = 1 \ldots L)$ issued by the customer pk_{C} along with the proof Π_{id}^i of pk_{C}'s ownership. In any other case, this algorithm does nothing.
- $\top / \bot \leftarrow \mathsf{VerifyOwnership_{id}}(S_{id},\ \Pi_{id},\ pk_{\mathsf{C}},\ L)$. This algorithm allows to publicly verify the proof Π_{id} that an ecoin with serial number S_{id} belongs to a doublespender with public key pk_{C}. We need to emphasize on the fact that the latter two procedures can only be applied to ecoins spent from W_{id} type of wallets.
- $\langle (BackUp_{\mathsf{B}}, (D_{\mathsf{hist}}{}',\ BackUp_{\mathsf{C}}) / \langle \bot, \bot \rangle \leftarrow \mathsf{BackUp}\ (pk_{\mathsf{C}},\ pk_{\mathsf{B}},\ date)[\mathsf{C}(sk_{\mathsf{C}},\ ACContent), \mathsf{B}(sk_{\mathsf{B}}^s,\ D_{\mathsf{hist}})]$. Customers follow this procedure in order to back their ACC up. Cardholders provide the encrypted content of their card $ACContent$ as well as a backup encryption password $pass_e$, while both parties output obtain a signed receipt of the final form of the backup stored $BackUp_{\mathsf{B}}$, $BackUp_{\mathsf{C}}.D_{\mathsf{hist}}$ is also updated with the new backup-record.

- $\langle ACC', (\mathsf{D}'_{\mathsf{credit}},\ \mathsf{D}'_{\mathsf{hist}},\ T_{p,id})\rangle / \langle \perp, \perp\rangle \leftarrow\ \mathsf{CardLossRecovery}(pk_{\mathsf{C}},\ pk_{\mathsf{B}})[\mathsf{C}(sk_{\mathsf{C}}), \mathsf{B}(sk^s_{\mathsf{B}},\ BackUp)]$. C checks whether $BackUp$ provided by her CIB B, is valid and collaborates with B to generate a new ACC. B provides proof of $BackUp$ validity. If the BackUp is not up-to-date, B checks $D_{\mathsf{hist}}$for double-spending case.
- $\langle (W'_{p,id}, monthly\_payment), (S_{id},\ \pi_{id},\ T'_{p,id},\ \mathsf{D_{hist}}',\ \mathsf{D_{credit}}')\rangle / \langle \perp, \perp\rangle \leftarrow$

  $\rightarrow\ \mathsf{CreditUpdate}(pk_{\mathsf{B}},\ pk_{\mathsf{C}})[\mathsf{C}(sk_{\mathsf{C}}),\ \mathsf{B}(sk_{\mathsf{B}}, D_{\mathsf{hist}}, D_{\mathsf{credit}})]$.

  CreditUpdate operation takes place between customer and Bank in order for customer to update her credit account according to her purchases. Customer provides her secret information (passwords, sk^s_{C} and wallets) and her outputs involve new wallets and her monthly bill. Bank checks D_{hist} for double-spending and if everything is fine, it updates its D_{credit}. Bank also obtains double-spending related tracing information for the new wallets withdrawn ($T'_{p,id}$).

- ErrorCorrection operation has two versions.
  - *Version 1:* This is the case, where an error has been detected in the expense report and we have two steps:
    1. $\langle \mathsf{RefCoup_{ACCA\rightarrow C}},\ \top, (\mathsf{Ref_M},\ \mathsf{RefConf_{AB}}), \mathsf{Ref_M}\rangle / \langle \mathsf{RefCoup_{ACCA\rightarrow C}}, \top, \mathsf{Rej_M}, -\rangle\ \leftarrow$
       $\leftarrow\ \mathsf{RefundConf}(pk_{\mathsf{M}}, pk_{\mathsf{CIB}},\ pk_{\mathsf{AB}})[\mathsf{C}(\mathsf{Rec_T}),\ \mathsf{M}(sk_{\mathsf{M}}),\ ACCA(sk_{ACCA}),\ \mathsf{AB}(sk_{\mathsf{AB}},\ \mathsf{D_{debit}})]$, where the C declares a charge error, i.e. requests a refund for a purchase she did not participate in. C obtains $\mathsf{RefCoup_{ACCA\rightarrow C}}$, which is basically a payment check either from merchant's AB, AB, or from the ACCA if merchant M rejects ($\mathsf{Rej_M}$) the refund. ACCA obtains (a) a proof of whether merchant accepted to provide the refund ($\mathsf{Ref_M}$) or not ($\mathsf{Rej_M}$), (b) a proof from AB of M's account balance, (RefundConf). AB simply updates $D_{\mathsf{hist}}$with $\mathsf{Ref_M}$.
    2. $\langle W'_{p,id}, (T'_{p,id}[,\ \mathsf{D_{hist}}']), (\mathsf{D_{debit}}',\ \mathsf{D_{hist}}')\rangle / \langle \perp, \perp,\ \perp\rangle \leftarrow$
       $\leftarrow\ \mathsf{RefCoupDeposit}(pk_{\mathsf{C}},\ pk_{\mathsf{CIB}}, pk_{\mathsf{AB}})[\mathsf{C}(sk_{\mathsf{C}},\ RefCoup_{ACCA\rightarrow C}),\ \mathsf{CIB}(sk_{\mathsf{CIB}}),\ \mathsf{AB}(sk_{\mathsf{AB}},\ \mathsf{D_{hist}},\ \mathsf{D_{debit}})]$.
       C interacts with her CIB, CIB, who contacts the AB/ACCA involved to deposit RefCoupDeposit. C's outputs are the new wallets. CIB updates D_{hist} if the error refers to an online transaction. M's entry in AB's $D_{\mathsf{debit}}$is updated.
  - *Version 2:* It refers to a plain purchase cancelation procedure:
    $\langle \mathsf{Ref_M}, \top, \mathsf{D_{debit}}'\rangle / \langle \perp, \perp\rangle\ \leftarrow\ \mathsf{PurchaseCancelation}(pk_{\mathsf{M}},\ pk_{\mathsf{C}})[\mathsf{C}(\mathsf{Rec_T}),\ \mathsf{M}(sk_{\mathsf{M}})]$, where C obtains a refund $\mathsf{Ref_M}$ from M, which he deposits to her CIB.
- $\langle (T^i_{det}, i = 1 \ldots x), \mathsf{D'_{anon}}\rangle \leftarrow\ \mathsf{OnlineExpenseReportRequest}[\mathsf{C}(\alpha_{\mathsf{C}}(x)),\ \mathsf{B}(\mathsf{D_{anon}})]$. C requests for an

expense report of her anonymous account with Bank B and provides the $\alpha_{\mathsf{C}}(x)$-related secret information. OnlineExpenseReportRequest operation outputs the series of A_{C}^{m} online transactions of C.

- $\langle W_{\mathsf{coupons}}{}', N_{\mathsf{M}}^{issued}\rangle/\langle\perp,\perp\rangle \leftarrow$ $\mathsf{CouponIssue}(bgpk, N_{\mathsf{coupons}})[\mathsf{C}(Randomness), \mathsf{M}(gbusk_{\mathsf{M}}, N_{\mathsf{M}}^{issued})]$. Merchant M and cardholder C collaborate for the latter to obtain N_{coupons} discount coupons (credit coupons). At the end of this procedure, C obtains a wallet W_{coupons} of N_{coupons} while M updates the number of coupons he has provided (N_M^{issued}).
- $\langle\top, D_{\mathsf{credit}}{}', \mathsf{D}'_{\mathsf{merchants}}\rangle/\langle\perp,\perp\rangle \leftarrow$

$$\mathsf{CouponDeposit}(pk_{\mathsf{C}},\ pk_{\mathsf{B}})[\mathsf{C}(W_{\mathsf{coupons}},\ sk_{\mathsf{C}},\ N), \mathsf{B}(sk_{\mathsf{B}},\ \mathsf{D}_{\mathsf{credit}},\ \mathsf{D}_{\mathsf{merchants}})].$$

In this operation, customer C deposits N coupons from coupon wallets W_{coupons}. Bank updates the merchants' statistics database as well as the customer's credit account according to N.

A.2 Security Properties

In this section we provide formal definitions of the security and privacy properties of the anonymous credit card system: *Correctness*, *No Overspending*, *Credit card Unforgeability*, *BackUp Integrity*, *Strong Credit Card Use Authorization*, *Conditional User Activity Untraceability*, and *Coupon Security*.

A.2.1 Correctness

1. If an honest customer C, eligible to own an anonymous credit card, runs AnonymousCreditCardIssue with an honest Bank B, then neither will output an error message; if an honest customer C, who has collaborated with a CIB CIB into a AnonymousCreditCardIssue procedure to issue an anonymous credit card, runs OfflinePayment protocol using that card with an honest merchant M, then M accepts the payment; if M runs MerchantPayment with an honest Bank AB, to deposit the payments he received from honest clients, then M's account will be increased by the value of the ecoins deposited.

2. If an honest customer C, owning a valid ACC, collaborates with her honest CIB CIB, into a AnonymousAccountSetup procedure to create an anonymous account A_{C}, then CIB accepts. In addition, if C

- runs OnlinePayment protocol using A_{C}'s secret information, then merchant's gateway G accepts the payment: when G runs OnlinePaymentDeposit, the cardholder's CIB CIB accepts and the merchant's account will be increased by the price of the product purchased.
- runs OnlineExpenseReportRequest with CIB, for her A_{C} account, C receives a detailed report of her A_{C} related online transactions.

3. If an honest cardholder C runs CreditUpdate with an honest Bank B, C's credit account will be increased by the aggregated amount of her purchases since her last CreditUpdate. C will be billed accordingly.

4. If an honest cardholder C detects a charging error and runs either version of ErrorCorrection protocol with honest merchants and/or Banks, then all parties involved will accept and C obtains refund wallets.

5. If an honest cardholder C runs Loss Recovery protocol with an honest Bank B, then she obtains an ACC with the same balance and transaction history as the one lost.

6. If an honest cardholder C obtains discount coupons W_{coupons} from an honest merchant M through CouponsIssue procedure and runs CouponDeposit with an honest Bank, ACCA accepts and updates C's credit account accordingly.

A.2.2 No OverSpending

No cardholder should be able to spend more money than her credit limit.

Offline ACC Use.

- No collection of cardholders should be able to spend more ecoins than the ones contained in their anonymous cards. Suppose that N cardholders $\mathsf{C}_1, \ldots, \mathsf{C}_N$ collude together, and that the sum of the amount of ecoins allowed to them is

$$N_C = \sum_{i=1}^{N} \mathsf{L}_{\mathsf{credit}}{}^{\mathsf{C}_i}.$$

Then, the number of different serial numbers of ecoins that can be spent to merchants — in OfflinePayment procedures — and/or Banks — in AnonymousAccountSetup/sf CreditUpdate — procedures is at most N_C.

- Suppose that one or more colluding peers run the OfflinePayment protocol with two merchants M_1 and M_2, such that M_1 gets $({S_{p,id}}^1, {\pi_{p,id}}^1)$ and M_2 gets $({S_{p,id}}^2, {\pi_{p,id}}^2)$, where ${S_{p,id}}^1$, ${S_{p,id}}^2$, ${\pi_{p,id}}^1$ and ${\pi_{p,id}}^2$ the sets of serial numbers of pairs of spent payment-identity ecoins and the corresponding proofs of validity, i.e.

$$ {S_{p,id}}^i, {\pi_{p,id}}^i = (S_p^i, \pi_p^i) \cup (S_{id}^i, \pi_{id}^i), i = 1, 2. $$

  Assume that $S_{p,id} = {S_{p,id}}^1 \bigcap {S_{p,id}}^2 \neq \emptyset$, and $\pi_{p,id}$ the set of spending validity proofs that correspond to $S_{p,id}$. Then, we require that $\mathsf{Identify}_{\mathsf{x}}(S_x, \pi_x), x = p, id$ outputs a public key pk_{C} and a proof of guilt Π^G such that $\mathsf{VerifyGuilt}_x(pk_{\mathsf{C}},\ S_x,\ \Pi^G)$ accepts.

- Each ecoin pair contained in payment-identity wallets that is accepted but not double-spent in the MerchantPayment protocol increases by its exact value merchant's AB's D_{debit} irrespective of the beneficiary of the ecoins. However, we don't regard it as a breach of security when a merchant M_1 received an ecoin but passed it to M_2, who deposited it into his debit account; in any event, this is just another form of collusion. Another justification is that the peer M_1 sacrifices his money.

Online ACC Use.

- No collection of cardholders should be able to spend more in online purchases than the sum of their current active anonymous accounts' balance. Suppose that N cardholders $\mathsf{C}_1, \ldots, \mathsf{C}_N$ collude together, and that of the balance $(M_{ot}^{m_i})$ of their active online accounts is

$$ N_{AC} = \sum_{\ell=1}^{N} M_{ot}^{m_i}. $$

  Then the overall amount of money that can be spent to merchants in online transactions or returned to their owners is N_{AC}.

Error correction may constitute an exception in this property if the cardholder lies for the error having taken place and acquires from the ACCA — if we assume that merchant does

not accept to provide a refund — new wallets. We consider this to be a general risk credit card associations take. In addition, as in error correction case customers identify themselves, it is a reasonable assumption may not attempt to cheat in this way multiple times.

A.2.3 Credit Card Unforgeability

- No customer/merchant or collection of customers and/or merchants should be able to create a credit card of the form of the Anonymous Credit Card described before, which when used to run OfflinePayment or AnonymousAccountSetup protocols with an honest merchant or Bank respectively noone outputs error message.

A.2.4 BackUp Integrity.

- Let that a cardholder C has run BackUp protocol with her CIB, CIB, at a particular date. There should be impossible for C to run Loss Recovery protocol with CIB, with a different backup than the most recent one.

A.2.5 Authenticated Use of Services

We require that an unauthorized individual may not make use of an ACC's functionalities. In particular, we consider the following cases:

ACC Payment.

- (Offline Payments) No cardholder or merchant or collection of customers and/or merchants should be able to use another customer's ACC in OfflinePayment, or AnonymousAccountSetup protocol without possessing sk_{C}.

- (Online Payments) No cardholder or merchant or collection of cardholders and/or merchants should be able to run Online Payment protocol on an anonymous account they do not possess the secret of succesfully, i.e. without error message.

Expense Report Request.

- No cardholder or merchant or collection of customers and/or merchants should be able to run OnlineExpenseReportRequest for an account A_{C} successfully, without possessing

A_{C}'s secret information.

Another aspect of Authenticated Service request is the *user non framability.* More specifically:

- No coalition of customers, even with Bank, can forge a proof Π_G that $\mathsf{VerifyGuilt_p}(pk_{\mathsf{C}},\ S,\ \Pi_G)$ or $\mathsf{VerifyGuilt_{id}}(pk_{\mathsf{C}},\ S,\ \Pi_G)$ accepts where pk_{C} is the public key of an honest cardholder, i.e. a cardholder who did not double-spent an ecoin with the serial number S.

A.2.6 Conditional Non Traceability

- *Non Tracing of non-double-spenders.* Given that a cardholder C has issued an ACC with an honest CIB CIB through an AnonymousCreditCardIssue protocol, participated in an OfflinePayment or AnonymousAccountSetup procedure, it should computationally impossible for any merchant or CIB or any collusion between the two to infer any information regarding sk_{C} or pk_{C}. In addition, given the outputs of two different offline spending procedures of C it should be impossible to link one to the other as having been done by the same person.

- (Non Frameability)No coalition of customers, even with Bank, can forge a proof Π_G that $\mathsf{VerifyGuilt_p}(pk_{\mathsf{C}}, S, \Pi_G)$ or $\mathsf{VerifyGuilt_{id}}(pk_{\mathsf{C}}, S, \Pi_G)$ accepts where pk_{C} is the public key of an honest cardholder, i.e. a cardholder who did not double-spent an ecoin with the serial number S.

- *Tracing of double-spenders.* Given that a cardholder C is shown guilty of double-spending ecoin S_p by a proof Π_G such that $\mathsf{VerifyGuilt_p}$ accepts, this property guarantees that $\mathsf{Trace_{id}}(\mathsf{params},\ S_{id},\ pk_{\mathsf{C}},\ \Pi_G,\ D_{\mathsf{hist}})$ will output the serial numbers $S_{id_1}, \ldots, S_{id_m}$ of all coins that belong to customer along with proofs of ownership $\Pi_{id_1}, \ldots,\ \Pi_{id_m}$ such that for all i with high probability, $\mathsf{VerifyOwnership_i d}(S_{id_i},\ \Pi_{id_i},\ pk_{\mathsf{C}})$ also accepts.

A.2.7 Coupon Security

- (Correctness) If a customer obtains discount coupons W_{coupons} from an honest merchant M through CouponsIssue procedure and runs CouponDeposit with an honest Bank, ACCA accepts and increases M's popularity and customer's credit account accordingly.

- (Balance) No collection of customers should be able to deposit more coupons than the ones legaly issued by merchants $\mathsf{M}_1, \ldots, \mathsf{M}_m$ they interacted. Suppose that N customers $\mathsf{C}_1, \ldots, \mathsf{C}_n$ collude together, and that the sum of the coupons allowed to them is $N_{coupons}$. Then, the number of different coupons that can be accepted when deposited to Bank is at most $N_{coupons}$.

- (Unforgeability) No customer or coalition of customers should be able to create coupon-wallet W_{coupons}, such that when provided to CouponDeposit are accepted by Bank as issued by an honest merchant *merchant*, without M's collaboration.

- (Anonymity) Bank or any coalition of merchants should not be able to link a particular coupon to a particular identity.

A.3 Security Proof

The following theorem states the correctness and security of our general scheme.

Theorem 8 *If the underlying primitives (ecash system, blind signatures and group blind signatures) are secure, then our scheme satisfies* correctness, no overspending, credit card unforgeability, backUp integrity, no unauthorized use of ACC, conditional non-traceability *and* coupon security.

Lemma 9 *If the underlying primitives (blind signatures, blind group signatures and ecash system) are secure, then our scheme satisfies* correctness.

Proof Sketch. From the correctness of the secure ecash scheme and the secure blind signature scheme, our scheme satisfies the first (1) condition of the correctness. Hash functions' invertibility satisfies the second (2) and fourth (4) parts of correctness definition, while

the balance property of the ecash scheme guarantees the third ACC correctness property. The combination of correctness and anonymity revocability attribute of the underlying compact ecash scheme [Camenisch *et al.*, 2005]satisfy the fifth (5) condition. The correctness of the secure group blind signature scheme guarantees and of the plain blind signature scheme satisfy the last (6) correction condition (see coupons' security proof analysis for more details).

Lemma 10 *If the underlying primitives (digital signatures, blind signatures, blind group signatures and ecash system) are secure, then our scheme satisfies* Credit Card Unforgeability*.*

Proof Sketch. From the unforgeability and consistency of the secure ecash scheme, our scheme guarantees that no valid wallets can be created without Bank's collaboration. Thus, Anonymous Credit Card Unforgeability is satisfied.

Lemma 11 *If the underlying primitives (blind signatures and ecash system) are secure, then our scheme satisfies* No OverSpending*.*

Proof Sketch. In *Offline Payment.* Being ecash -based, the offline conditions of No OverSpending definition are all satisfied by the no overspending property of the underlying secure compact ecash scheme [Camenisch *et al.*, 2005].
In *Online Payment.* This is basically controlled by the bank, where the anonymous account is situated. After each transaction, bank substracts from the anonymous account, the amount spent, while it rejects any transaction requiring more money than the anonymous account's balance.

Lemma 12 *If the underlying primitives (digital signatures) are secure, then our scheme satisfies* BackUp Integrity*.*

Proof Sketch. It is satisfied by the unforgeability property of digital signature. In particular, in the end of the backup procedure, both Bank and customer sign a hash of the encrypted content of the card concatenated with the corresponding date. Customer has no motivation to lie (because of double-spending consequences) and Bank's signature on the hash of the backup acts as undeniable proof of authenticity on client's behalf.

Lemma 13 *If the underlying primitives (ecash, digital signatures, hash functions) are secure, then our scheme satisfies* No Unauthorized Use of ACC.

Proof Sketch. We will address each ACC use seperately. *Non framability* is satisfied from the exculpability of the secure ecash scheme. *No unauthorized offline payment* requirement is satisfied through the all or nothing non transferability property of the underlying ecash scheme. In particular, to make a valid offline payment using an ACC's wallets, the secret key of the ACC owner is required. *No unauthorized online payment* property is also satisfied by the non invertibility properties of hashes. To authenticate herself the person attepting to make a payment is required to provide the pre-image of the current anonymous account number she only knows. Automatic changes of the account number and timestamps prevent a third party to succeed in replay attacks. Authentication in the *Expense Report Request* case is achieved in a similar case as in online payment protocol.

Lemma 14 *If the underlying primitives (blind signatures and blind group signatures) are secure, then our scheme satisfies* Coupons Security.

Proof Sketch.

- (Correctness) Verifiability of a secure goup blind signature scheme guarantees that an "honest but curious" Bank will accept all the credit coupons generated by honest merchants and update the merchant statistics database accordingly. Verifiability of a secure blind signature scheme guarantees that an "honest but curious" Bank will accept all the debit coupons generated by honest customers and update the credit database accordingly.
- (Balance) Credit coupons are subjected to double-use check when deposited and debit coupons are only issued in a rate one for each valid credit coupon. Since debit coupons are also subject to double-using check when deposit, no customer or coalition of customers can eventually deposit more coupons (debit) than the ones initially obtained by merchant.
- (Unforgeability) Unforgeability of blind group signatures guarantee that no customer or coalition of customers can forge credit coupons. Unforgeability of blind signatures guarantees debit coupons' unforgeability.
- (Anonymity) Blindness property of blind signature scheme guarantees that Bank cannot

link a debit coupon or a set of debit coupons to a credit or set of credit coupons. Thus *Anonymity* property is satisfied.

Lemma 15 *If the underlying primitives (blind signatures, group blind signatures and ecash system) are secure, then our scheme satisfies* Credit card conditional non traceability*.*

Proof Sketch. Simply from traceability attributes of the secure ecash scheme, our scheme satisfies the *conditional non-traceability.*

From the ecash properties, the only case for a customer's identity to be revealed, is when she double-spends part of her wallets to exceed her credit limit (identity or payment). As the two wallets are spent concurrently in all payment procedures, double-spending a part of the payment wallet implies double-spending the corresponding part of the identity one as well. Thus, the double-spender's identity is revealed.

In our scheme, there are three cases, where a customer C double-spends parts of her wallet(s):

1. C loses her card, reports her loss but in the period between the last card backup and the loss of the card, C has used the card to make purchases. In this case, C spends the rest of *only* the payment wallet to Bank. Thus, although C's transactions within this time period (critical) get revealed — when merchants deposit the double-spent parts of C's wallet — her transaction activity preceeding and following the critical period remains secret.

2. C copies her card, and uses two cards at the same time. In this case, identity wallet is spent twice and because of the tracing attribute of tracing enabled ecash scheme, the serial numbers of all coins C has withdrawn are revealed.

3. C copies her card, updates her credit limit using one copy and uses the other copy to make purchases. This is another case of on purpose double-spending, similar to case 2.