Electronic cash: cryptography & distributed systems

Van Khanh Nguyen
University of Wollongong
NOTE

This online version of the thesis may have different page formatting and pagination from the paper copy held in the University of Wollongong Library.

UNIVERSITY OF WOLLONGONG

COPYRIGHT WARNING

You may print or download ONE copy of this document for the purpose of your own research or study. The University does not authorise you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site. You are reminded of the following:

Copyright owners are entitled to take legal action against persons who infringe their copyright. A reproduction of material that is protected by copyright may be a copyright infringement. A court may impose penalties and award damages in relation to offences and infringements relating to copyright material. Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.
ELECTRONIC CASH:
CRYPTOGRAPHY & DISTRIBUTED SYSTEMS

A thesis presented
by

Van Khanh Nguyen

to
The School of Information Technology and Computer Science

in partial fulfillment of the requirements
for the degree of

Master of Science (Hons.)
in the field of
Computer Science

University of Wollongong
NSW, Australia
August 1997
DECLARATION

This is to certify that the work presented in this thesis was carried out by the author in the School of Information Technology and Computer Science of the University of Wollogong and has not been submitted for a degree to any University or Institution.

Van Khanh Nguyen
Acknowledgments

I would like to greatly thank Associate Professor Rei Safavi-Naini whose professional and personal supervision over the last years has made the completion of this project possible. Rei has been an excellent supervisor and I will always be grateful for what I have been able to learn from her. Given that at the start of working with Rei my English was poor and my background was not in Computer Security and Cryptography, I have always been surprised how effectively she has directed me to achieve the level of knowledge that I am enjoying now.

Gratitude is also expressed to the following organizations:

• The former Department of Computer Science of the University of Wollongong that provided the course and the study environment that I always enjoyed.

• The Australian Association of International Development that provided me with the scholarship to study for a Master degree in Australia.
VITA & PUBLICATIONS

A. VITA

May 17, 1970  Born, Hanoi, Vietnam
1992  Bachelor in Computing Science & Engineering, Hanoi University of Technology, VN
1995  Ms of Science, Hanoi University of Technology, VN
1997  Ms of Science (Hons.), University of Wollongong, Australia

B. PUBLICATIONS


# TABLE OF CONTENTS

VITA & PUBLICATIONS ........................................................................................................ VI

TABLE OF CONTENTS ........................................................................................................ VI

ABSTRACT ............................................................................................................................. X

I. INTRODUCTION ................................................................................................................. I-1

1.1 INTRODUCTION TO ELECTRONIC CASH ................................................................. I-1

1.2 BASIC E-CASH MODEL ............................................................................................... I-4

1.3 RESEARCH METHOD ................................................................................................. I-9

1.3.1 Achieving the fundamentals ......................................................................................... I-9

1.3.2 Achieving a framework ............................................................................................... I-10

1.3.3 My own treatment: a new approach in modeling ....................................................... I-10

1.3.4 A new system ............................................................................................................. I-12

1.4 THE ORGANIZATION OF THE THESIS WITH A BRIEF STATEMENT OF THE MAIN RESULTS ... I-13

II. PRELIMINARIES .............................................................................................................. II-17

II.1 A BRIEF SURVEY ON E-CASH DESIRABLE PROPERTIES ...................................... II-18

II.1.1 Physical independence ................................................................................................ II-18

II.1.2 Security ...................................................................................................................... II-18

II.1.3 Privacy protection ....................................................................................................... II-19

II.1.3.1 Untraceability & unlinkability ................................................................................ II-19

II.1.4 Off-line payment ....................................................................................................... II-20

II.1.4.1 On-line vs. Off-line .............................................................................................. II-20
II.1.5 Transferability
II.1.6 Divisibility
II.2 Protection against double-spending
   II.2.1 Detection after the fact
      II.2.1.1 Cut-and-choose based e-cash
      II.2.1.2 Single-term off-line e-cash
      II.2.1.3 Pseudonym-based e-cash
      II.2.1.4 Probabilistic random auditing
   II.2.2 'Wallet with Observer'
      II.2.2.1 Incorporating 'detection after the fact' with 'wallet with observer'
   II.2.3 Employing trusted third-party
II.3 Revocable anonymity
II.4 Blinding techniques
   II.4.1 A formal definition of blind signature
   II.4.2 Brands' restrictive blind signature
   II.4.3 Blinding by trusted third party
   II.4.4 Versatility
III. A Framework for combining off-line & on-line e-cash
   III.1 Introduction: A framework to overcome the drawbacks in previous e-cash systems
   III.1.1 drawbacks in previous e-cash models
   III.1.2 Our proposal
   III.1.3 Off-on-line
   III.2 Conceptual model
      III.2.1 The Anti Cheating Server
      III.2.2 Hierarchical network of ACS
      III.2.3 Using hash value
III.2.4 Definitions and notations ........................................ III-50

III.3 THE SYSTEM ..................................................... III-53

III.3.1 Specification .................................................. III-53

III.3.1.1 Procedure for small-medium value coins (A) .................................. III-55

III.3.1.2 Procedure for tiny value coins (B) ................................................ III-59

III.3.1.3 Procedure for big value coin (C) .................................................. III-60

III.3.1.4 Procedure for Check-at-the-end-of-day (D) ................................... III-61

III.3.2 System parameters ............................................. III-62

III.3.3 Example .......................................................... III-65

III.3.4 Possible attacks ............................................... III-66

III.3.4.1 Organization cheating ................................................ III-66

III.3.4.2 Common boundary effect ................................................ III-67

III.3.5 Extension for shopping over the Internet ................................ III-68

III.4 CONCLUDING REMARKS ........................................ III-69

IV. E-CASH WITH 'MIRROR WALLET' ................................ IV-72

IV.1 THE BASIC MODEL WITH NEW NOTIONS ................... IV-73

IV.2 THE SCHEME .................................................... IV-75

IV.2.1 Notations, message formats and cryptographic Components ........ IV-76

IV.2.1.1 Partial blinded signature scheme - How to date blind signature .... IV-76

IV.2.1.2 Collaborative generation of a pseudo-random number ............... IV-77

IV.2.1.3 Coin construction .............................................. IV-78

IV.2.2 Protocols ...................................................... IV-79

IV.2.2.1 Wallet Setup ................................................ IV-79

IV.2.2.2 Withdrawal ................................................... IV-80

IV.2.2.3 Payment ...................................................... IV-82

IV.2.2.4 Auditing ..................................................... IV-83

IV.2.2.5 Deposit ....................................................... IV-84

IV.2.2.6 Wallet loss recovery ........................................ IV-84
ABSTRACT

Electronic Payment Systems that deal with daily shopping and payment over the Internet are one of the most important concerns of today’s electronic commerce. There is a large body of research in this field, and in particular, on electronic cash (e-cash) systems. In e-cash systems, protecting customer’s privacy has received much attention and many challenging problems relating to the trade-off between multi-party security and system efficiency are proposed.

This thesis gives a brief survey of e-cash systems. We investigate various problems raised in building secure and efficient e-cash systems. This includes efficient off-line e-cash, perfect anonymity, revocable anonymity, combining the advantages of on-line and off-line approaches and wallet loss recovery.

We will have a wide approach to the study of e-cash, which includes traditional cryptographic approaches and approaches inspired by distributed computation and processing. Our principal idea is to extend traditional e-cash models to allow distributed processing and storage, and to combine the efficiency of off-line e-cash with the security (against double-spending) of on-line e-cash. To realize these ideas we introduce new notions:
(1) mirror wallets to support various desirable e-cash properties such as revocable anonymity and wallet loss recovery.

(2) auditing servers to audit the customers’ transactions.

(3) a dual-named server, the so-called ACS (*Anonymous Communication Server and Anti-Cheating Server*), to provide anonymous communication and restrain cheating and crime.

This results in a new proposal for e-cash, which allows various trade-offs between security and efficiency. The proposal is efficient and versatile and yet succeeds to control the potential risk of trust abuse where an insider in the trusted third party (auditing server) is corrupted.
I. INTRODUCTION

1.1 INTRODUCTION TO ELECTRONIC CASH

Over the last decade business organizations have increasingly automated and computerized their activities. With the rapid growth of the financial networks and the Internet, electronic financial transactions have become increasingly common place. Electronic Payments Systems (e-payment systems) dealing with daily shopping and payment over the Internet are one of the most important concerns of today’s electronic commerce.

The efforts to build e-payment systems can be divided into two general approaches: account-based systems and token-based systems. Account-based systems such as credit card based systems have been dominating the market since early days of electronic payment as the idea has been attractive to the customers and credit card companies. In these systems, payment security can be achieved without technical difficulty and possible abuses are largely eliminated. However the customers must ignore the privacy they deserve: the financial institutions can easily build customers’
transaction profiles and direct their marketing policy. Moreover, if these profilers are leaked, adversaries and competitive parties might misuse them and hence cause unpredictable harm to the customers’ business.

Later, electronic cash, the main token-based model, was introduced with the novel idea of simulating the untraceable flow of the real cash in society. A digital token bound to a certain money value can be produced by a bank and be withdrawn by a customer as replacement for real cash. The token can be electronically spent at a shop without leaking any information about the customer’s identity. The property of preserving customer’s anonymity can be achieved just like using real cash. Although e-cash approach has some implementation difficulty, as it does not rely on any existing industrial infrastructure (while credit card – based approaches does) and sometime satisfying the required conditions is not easy, e-cash approach deserves a broad and deep treatment because of the very important properties it promises.

Despite these problems, e-cash approach is becoming more mature. This can be easily seen by having widely available systems such as DigiCash’s e-cash system (More information can be referred to [DigiCash]).
The background

Cryptographic techniques are always important tools in building e-payment systems, especially e-cash systems. Throughout the last ten years, this area has received an wide interest from many cryptographers. E-cash is a very good meeting place for ambitious and imaginative thinking to combine the theoretic cryptography with the practice of electronic commerce. Motivations can come from different practical scenarios and can result in new notions and techniques, enriching both the two fields.

This effort from the cryptographic community has brought about many e-cash proposals with new properties (that one could not enjoy in other systems such as credit card-based): customer anonymity, off-line payment, coin divisibility (in that coins are multi-spendable).

However, ambitious goals result in very complex cryptographic protocols which are very difficult to assess and examine. Many attacks have been considered against these schemes which in turn have given birth to revisions and other new schemes, which again have been attacked and modified, making e-cash an active area in the field of applied cryptography. These all cause e-cash an interesting but difficult field of study.

Chapter 2 – ‘Preliminaries’— will provide a brief survey on e-cash literature.
Thesis aim statement

This thesis aims at examining Electronic Cash in the background of applied cryptography and distributed processing. We consider various approaches in constructing e-cash schemes, pointing out the main problems and propose improvements.

We study the technical aspects of e-cash schemes and are not concerned with policy aspects related to development and growth of e-cash systems in practice.

1.2 BASIC E-CASH MODEL

We first show the basic e-cash model, which will support us to present the main problems investigated in the thesis and the research methodology taken to tackle them. Firstly we describe the traditional model, and then we provide our proposed system aimed at distributed processing.

In an abstract level, a basic e-cash system is a set of protocols between three main parties:

+ The user (customer, payer, prover, Alice)

+ The shop (merchant, payee, verifier, service provider, Bob)

+ The issuer bank
Other possible parties are the judge and the certification authority.

There are three main procedures accomplished between these parties:

- Withdrawal protocol: Alice identifies herself to the bank and asks to withdraw money. The bank gives her some e-coins and debits the equivalent amount from her account.
- Payment protocol: Alice pays Bob by e-coins. Bob checks the validity of these e-coins by verifying the bank’s signature on them.
- Deposit protocol: Bob deposits the e-coins to the bank. The bank verifies and updates Bob’s account.

Other possible procedures may be:

- Opening customer’s account: the bank creates an account for Alice and sets up parameters for later use.
- Check of double-spending: the bank checks if the e-coins have been spent before.

The e-coins are just strings of bits and so can be easily duplicated and hence cheaters’ double-spending is possible. In on-line systems, the bank takes the check for double-spending in real-time as the shop contacts and deposits e-coins at the time of payment. This can be done as the bank maintains a database of all the e-coins spent in
the past. Therefore, the bank is able to make the shop accept or cancel the purchase, but with an excessive communication and computation cost in real-time.

In off-line systems, the shop deposits e-coins at off-peak time (probably at the end of day or the end of week) and double-spending can only be detected after the event. Complex cryptographic means are required to detect the double-spender while preserving the anonymity of the honest customers.

![Figure 1. Basic E-cash model](attachment:image.png)
Electronic cash system can also be seen as a distributed system with three processing node – the customer (C), the bank (B) and the shop (S). There are three pairwise interactive processes – withdrawal (\( P_w \)), payment (\( P_p \)) and deposit (\( P_d \)) – between the pairs (C,B), (C,S) and (B,S), respectively. The bank has the central role in the system and maintains 2 databases:

+ A list of reports as views of withdrawals (\( L_w \))
+ A list of reports as views of deposits (\( L_d \)).

\( L_W \) is updated during \( P_W \) and \( L_D \) is updated during \( P_D \).

In anonymous e-cash\(^2\), items in \( L_w \) are different and unlinkable to the ones in \( L_d \); this is due to the *blinding process*\(^3\) in \( P_w \).

In on-line e-cash, \( P_p \) and \( P_d \) are simultaneous, while in off-line e-cash, they are separate.

\(^2\) This name is used to separate the considered systems from some other unimportant systems, which are also claimed as of e-cash approach in spite of not providing anonymity. They are actually account-based and are not considered in this thesis.

\(^3\) The process occurs in a blinding digital signature, where a message has been masked before giving to the signer who is still able to sign on without seeing the real message. Blind signature schemes will be discussed in the next chapter ‘Preliminaries’.
Thus, e-cash model is a protocol among three or more parties, skillfully designed to prevent the bank from linking and tracing the customer’s payments, even if the merchant colludes with the bank.

The following diagram illustrates the correspondence that maybe found between traditional and electronic shopping systems.

<table>
<thead>
<tr>
<th>Traditional</th>
<th>Computerized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customers, banks, shops</td>
<td>Computers, smart cards ...</td>
</tr>
<tr>
<td>Payment methods</td>
<td>Cryptographic protocols</td>
</tr>
<tr>
<td>Communication means</td>
<td>Computer networks</td>
</tr>
<tr>
<td>(usually, face-to-face)</td>
<td></td>
</tr>
<tr>
<td>Cash</td>
<td>Electronic coin</td>
</tr>
<tr>
<td>Wallet</td>
<td>Electronic wallet</td>
</tr>
<tr>
<td>Credentials (certificates, licenses,</td>
<td>Database Wallet</td>
</tr>
<tr>
<td>records ...)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. Electronic Cash: Automation of daily shopping by cash
1.3 Research Method

It is hard to say that I started studying the field with a ready and clear research methodology in mind. Actually the topic has been absorbed gradually through all the different stages. Below I will present the main steps that I have followed through the whole work.

1.3.1 Achieving the Fundamentals

Given that the literature on e-cash was large with there were many proposals with sophisticated strong cryptographic techniques, I tried to gradually understand the basics of the most important schemes. Following this I produced a draft classification of the proposed schemes based on their properties and functionalities and the approaches they had followed. To understand an approach, I made a ‘history’ report of the approach, listing and studying all the important results from the beginning to the most up to date one. This helped me to clearly understand the development of each approach, the aims of the approach and the cryptographic tools used or devised for that.

Some results were really difficult to understand. Without this proper strategy, given that my background of the first degree did not relate to cryptography, it would be unable for me to reach to ‘the state of art’ knowledge in this field.

---

4 Some results were really difficult to understand. Without this proper strategy, given that my background of the first degree did not relate to cryptography, it would be unable for me to reach to ‘the state of art’ knowledge in this field.
1.3.2 **ACHIEVING A FRAMEWORK**

After obtaining a thorough knowledge in this field, I was able to construct a framework in which the notions and techniques in e-cash systems were classified. From this, it was possible to see what the problems in earlier e-cash systems had been, and how they were solved later. Also which schemes were considered good at that time. This helped to point out the deficiencies and the shortcomings that still existed to that date and suggested where I could extend the systems and introduce new construction.

1.3.3 **MY OWN TREATMENT: A NEW APPROACH IN MODELING**

During the initial survey period I noted that most e-cash proposals were devised by cryptographers and mathematicians and were quite abstract and high level. However I could not find much work on distributed methods, which looked very promising and required much more development.

Here I noted that, if one put more participants into the basic model and distribute the functions and the workload, then the workload of each participant will be reduced and new useful functions can be implemented.
Following this came the idea of extending the basic model and reorganizing the system functions. I used the distributed processing approach and from that, three initial considerations were made and later refined:

(1) **Prior restraint by third-party servers.** I noticed that in on-line e-cash systems, the 'shield' to protect against double-spending (prior restraint rather than detection after the fact) was in the bank's module, while in 'wallet with observer' e-cash, it was in the customer's module. I then proposed a 'shield' to be deployed by using another party different from C, B and S. This party would play a role independent from the bank's and the customer's, giving no favor to any other party. This middle party could also help to protect against attacks from criminals and provide a desirable trade-off between the most important three aspects of e-cash: security for the bank, privacy for the customer and law enforcement when customer anonymity is abused.

(2) **Distribution of processing.** Secondly, I found that there were two extremes in e-cash: on-line payment and off-line payment. On-line payment systems were preferred because they could effectively stop double-spending. On-line payment systems are a 'must' for high value purchases. However, on-line payment creates a bottleneck in the bank which becomes more critical when $L_d$ grows in size. Adding other

---

5 This concept will be discussed in detail in chapter 2 'Preliminaries'
processing servers between the bank server and the shop computers could help to absorb the overhead that the bank host had used to handle alone. Moreover, the range between a fully on-line and a fully off-line payment could be filled in by distributing and varying the processing in the network of intermediate servers in the coin-checking process.

(3) **Distribution of storage.** To reduce the computation-load in checking double-spending at the deposit time, I proposed to distribute $L_0$ into sub-lists maintained by different servers. Using this approach, the computation load at the bank could be reduced by a factor determined by the number of $L_0$ sub-lists.

### I.3.4 A NEW SYSTEM

The above directions were followed to develop a new system with new properties. First a high level model was set up, defining participating parties and their general functionalities. In the second phase, many step-by-step refinements were used to obtain a detailed design that could be implemented.

In this refinement phase, the flow of information between different parties were analyzed, and to achieve the goals, new notions were introduced. A thorough
understanding of e-cash systems allowed me to learn many cryptographic tools that I later used to build my own system.

Finally, I had to learn how to analyze the security of a systems under all possible attacks and employ this skill to the new proposed system achieved.

In summary, my research strategy was as follows:

'studying the proposed systems → development of a basic model → analyzing the main approaches → a framework → analyzing deficiencies and shortcomings → new model → refinement → designing a new systems'.

1.4 The organization of the thesis with a brief statement of the main results

The thesis is divided into six chapters. Chapter 1 is an introduction. Chapter 2 provides preliminaries on applied cryptography for the field of e-cash. Chapters 3, 4 and 5 present the main body of this thesis – our main results is a new e-cash scheme. Chapter 6 reviews all the investigated and the findings.

6 Or, this can be simply said 'analysis – modeling – refinement – achieving new systems'.
In the following we briefly introduce the main results which will be discussed in depth in chapters 3, 4 and 5.

**An e-cash framework with a ‘near-line’ checking**

In chapter 3 we have proposed intermediate servers to setup a ‘near-line’ checking: these servers are always available to shops in a small neighborhood and are to be contacted at the payment time, or with varying delay period, depending on the purchase value. This helps to setup a ‘near prior-restraint’ against double-spending. Also, it helps to distribute and moderate the checking process. This results in a framework to extend existing systems to achieve a continuum between on-line and off-line payment.

**E-cash schemes with distributed storage and processing**

The following will discuss a novel idea used in this thesis. Multiple (N) servers for processing deposits are used (*deposit server*). Each customer is bound to one particular deposit server. This can be done by encoding the name of that server into each coin minted at the withdrawal phase by that customer, using some special tools. This means that the deposit transactions from a particular customer are always processed by a fixed deposit server to check the $L_0$ sub-list maintained by that server. Clearly, communication, computation and storage required in each deposit server is
an order of N less than that required in the bank host computer in the traditional on-
line e-cash system.

We develop further the above idea in chapter 4 & 5 where customers are not only
bound to fixed deposit servers but also their submitted coins (after deposit) are stored
in separate storage compartments of each deposit server, called 'mirror wallets'\(^7\). 
This provides further saving in computation time (by the factor that is to be
determined by the number of customers per each deposit server).

Moreover, we propose an auditing functionality for the deposit server that makes it
the 'auditing server'\(^8\), which has an independent role having no favor to either the
bank or the customer. We present the basic scheme with one auditing server in
chapter 4.

Finally, we propose an architecture with multiple auditing servers which anonymously
communicates to outside world (banks, customers, shops) and thus, we have the final
and the most effective scheme in chapter 5.

\(^7, 8\) These notions will be discussed in depth in the chapter 4 when the context is fully described
The following figure will illustrate the stages of the development of the ‘distributed system’ idea.

A/

Customer

Shop

Bank

e-coin

B/

Customer

Shop

e-coin

Deposit servers

C/

Customer

Shop

e-coin

Anonymous auditing servers

mirror wallet

Figure 3. A) The traditional e-cash model. B) The extension with multiple deposit servers. C) The extension with multiple anonymous auditing servers.
II. PRELIMINARIES

Existing digital payment systems are often designed with a high priority for the security of the financial institutions and thus there is not much to guarantee that the customers' trust is not abused. The customer must trust the system that is employed by the bank and there is no way to ensure that the system will not be attacked by a dishonest insider in the bank. In the case of dispute, no party has any legal base to prove itself and disclose the fraud of the other party. Electronic cash is the only realm, in which this issue is addressed and the customers' rights of privacy is protected.

Recently, there has been a large body of research in this field and so an attempt to review the field in depth would require a great effort that is probably out of the scope of this thesis\(^9\). Here we shall try to address the most major issues and problems and

\(^9\) A review like that can be seen in [Tsi97]
briefly discuss the main underlying techniques, giving a technical background for the presentation of our results in the next chapters.

Firstly, we present a brief survey on the most important system properties in e-cash.

**II.1 A BRIEF SURVEY ON E-CASH DESIRABLE PROPERTIES**

"Universal Electronic Cash" by Okamoto and Ohta ([OkOh91]) is the first attempt at defining an ideal electronic cash system. In this work, six most desirable properties are identified. They are: *physical independence, security, protecting privacy, off-line payment, transferability* and *divisibility*.

**II.1.1 PHYSICAL INDEPENDENCE**

Unlike real cash, the security of e-cash does not rely on the physical means. The e-coins are digital numbers and can be easily transferred through the world wide networks.

**II.1.2 SECURITY**

The ability to copy (reuse) and forge the e-cash must be prevented. Strong cryptographic tools are needed in e-cash to protect against abuses.
II.1.3 PRIVACY PROTECTION

The privacy of customer must be protected. That is, the relationships between the user and his purchases must be untraceable. This is the most important property of e-cash systems that distinguishes them from other e-payment systems.

It should be noticed that real cash does not strictly obey 'the rule of untraceability' as mentioned above. This is true because theoretically, real cash can be traced using serial numbers on the notes. But this is rarely used in practice.

II.1.3.1 Untraceability & unlinkability

There are two levels in protecting a customer's privacy in electronic cash: untraceability (anonymity) and unlinkability. Untraceability means the inability of the bank to link the customer to her purchases, even with the help of the shops. Unlinkability is stricter: a purchase can not be linked to the customer who made it and also can not be linked to the other purchases made by the same customer in the past.

To understand that why unlinkability is necessary, let us consider an e-cash system that provides untraceability but does not provide unlinkability. The bank here can not link customers to their purchase transactions, however still the bank can create a dossier recording all purchase transactions, with multiple 'pages' that each of them
stores transactions originated by the same customer. Then if accidentally, a customer’s identifier is revealed in a certain purchase, e.g. the shop-keeper knows the customer, this dossier can be used to trace all the activities of this unfortunate customer for years. This will not happen with the systems capable to support unlinkability.

II.1.4 OFF-LINE PAYMENT

The deposit and then checking of double-spending is made after the payment instead of on-line and during in the payment. This saves real-time communication and computation cost.

II.1.4.1 On-line vs. Off-line

Most early e-cash systems are on-line systems. By using a central host which keeps live communications with all the shops’ computers, double-spending is prevented. The shop contacts the bank in real-time and hence payment and deposit are both done at the same time. This allows double-spending to be easily checked and prevented.

The main drawback of the on-line model is the overhead of live communication and real-time computation at the bank host. When the system grows, the bank host computer soon becomes a bottleneck, flooded with huge flow of transactions from vast number of shops and customers.
In off-line payment systems, the shop does not need to contact the bank in real-time and double-spending can be checked later at off-peak time. Detection instead of
prevention is used against double-spending. However off-line detection requires complex cryptographic primitives and still suffers the risk of misuse by powerful and malicious attackers. These drawbacks will be mentioned throughout this thesis. Today, most e-cash research focuses on off-line e-cash.

II.1.5 TRANSFERABILITY

The cash can be transferred, ‘hand by hand’, between customers, just like real cash. This property is later argued not to be really necessary, while introducing too much complexity in the system. A valid electronic coin must be encoded with the customer’s identity to detect double-spending. On transferring this e-coin to another person, there must be a reconstruction phase to re-encode the new owner’s identity into the e-coin. This actually results in another protocol which has a complexity equal to the withdrawal protocol and requires the involvement of the bank to sign the new coin ([Bra93a]).

II.1.6 DIVISIBILITY

An e-coin can be used many times in different purchases with different values, such that the total of all the ‘spent values’ does not exceed the coin’s ‘whole value’, determined in the withdrawal. This property is very important as otherwise the customer always need to keep a large number of e-coins of various values in order to pay with the exact needed value.
Real cash does not fully have this property as the changes that are refunded from the shops, are not ‘original parts’ of the notes that the shop received from the customers, and are usually many in quantity then.

Divisibility is first mentioned and implemented by Okamoto and Ohta ([OkOh89, OkOh91]). They both use the binary tree approach in that an e-coin is constructed with a tree structure that consists $N$ levels of nodes, each of them having zero or two offsprings. Each node bears a value so that is equal to the sum of the values on its direct two offsprings. The customer can gradually spend the coin ‘bit-by-bit’ by cut-&-pay nodes from this tree. If the value of a node at level $N$ (the leaves) is 1 cent then the customer can always pay exact amounts in cents by this tree-structured e-coin.

However the proposed schemes are impractical as they both rely on the cut-and-choose method and require too much storage and computation. Later, Okamoto constructed a much more efficient divisible e-cash ([Oka95]) that still uses binary approach but is based on the Brands’ single-term e-cash ([Bra93a], the most efficient off-line e-cash designed so far).

Another approach to achieve divisibility is to employ the concept of account balance. A coin can be spent many times, each time with a different value, and still remain
spendable as long as the total paid value has not reached the total value assigned to the coin at the withdrawal (or the balance/counter is still positive). However in this case, the problem of how to prevent overspending (that replaces the notion of double-spending in indivisible e-cash) becomes harder as it is much more difficult to trace overspending.

SmartCash and Mondex are existing systems with this feature, however the security of these systems depends on the security of the smart-card wallet and if the attacker successfully breaks into the smart-card he can freely change the counter to make a large profit. A better solution is the e-cash system with counter-based wallets ([Bra94]) based on the basic e-cash in [Bra93a].

Another solution for this approach is introduced by Jakobsson and Yung in [JaYu96]. A trusted party called ‘ombudsman’ takes part in the withdrawal will help the bank to trace the overspenders. When it is required, the ombudsman will participate in payment phase (then payment is on-line) to check and catch the overspenders.

**II.2 PROTECTION AGAINST DOUBLE-SPENDING**

Detection and prevention double-spending (or over-spending for divisible e-cash) is the most important consideration in building off-line e-cash systems. There are three main approaches:
II.2.1 DETECTION AFTER THE FACT

A number of authors ([ChFN88, OkOh89, OkOh91, FrYu93, Bra93b, Fe94, Yac94, etc.]) have proposed systems to detect the dishonest customer’s ID from two deposit views of the same coin, while the anonymity of the honest customers is maintained. The main idea is to encode an e-coin with the identity of the customer who withdraws the coin. The coin consists of two secret components so that:

+ the coin can be seen as a line determined by 2 points - 2 secret components - on it.
+ during the payment, a secret component (a point on the line) is revealed and if the customer double-spends, these two revealed ‘points’ will help to determine the customer’s identity encoded into the coin.

This idea can be extended to build e-cash with k-spendable coins. The system will use the fact that a (k+1)-dimensional hyperplane can not be determined with k known points on it but is completely determined by k+1 different known points (of course, with the condition that these points do not lie in a k-dimensional hyperplane). These k-spendable coins have been constructed in [Bra93a] to extend the Brands’ basic e-cash with divisibility.

However encoding e-coins with the customers’ identity was never easy and, it took a long time for research on off-line e-cash to build practical off-line e-cash systems
(from 1988 with the first off-line scheme in [ChFN88] to 1993 with efficient schemes in [Bra93a, Fe94]).

11.2.1.1 Cut-and-choose based e-cash

Early off-line e-cash schemes ([ChFN88, OkOh89, OkOh91, FrYu93, etc.]) rely on the cut-and-choose method of Zero-Knowledge proofs. This is later criticized as being inefficient. To withdraw an e-coin, the customer constructs and presents to the shop $2n$ “terms”, each of which encoding customer’s identity and consisting of 2 inner components ($n$ is the security parameter). The bank randomly “cuts and chooses” $n$ “terms” and the customer discloses the structure of these $n$ “terms” to show that indeed they are honestly constructed. Then the bank signs (using blind signature) the remaining $n$ “terms” and returns them to the customer. In payment phase, a similar cut-and-choose mechanism is employed such that the shop by a challenge-response process can have 1 of the 2 (by random choice) inner components of each of the $n$ terms. Then if the customer dishonestly spends a coin twice, with overwhelming probability there exist a term, of which 2 different components are revealed to the shop, which when combined will reveal the full identity of the customer.

11.2.1.2 Single-term off-line e-cash

Cut-and-choose based e-cash is inefficient as communication, computation and storage requirements are multiplied by a factor of $n$. Single-term e-cash schemes are
[ChPe92, Bra93a&b, Fe94] introduced to get rid off of using cut-and-choose method in encoding customer’s ID into the e-coins. Single-term means that only one term is used to represent the e-coin instead of $n$ terms used in cut-and-choose based systems. Clearly, compared to cut-and-choose based e-cash, the efficiency of single-term e-cash is multiplied by a factor of $n$.

Single-term schemes are typically based on the Discrete Logarithm Problem (DLP) with the most efficient scheme to date, due to Brands’ ([Bra93a,b]). Brands’ scheme relies on the presentation problem, which is a generalization of DLP. Brands’ basic e-cash based on presentation problem has been used by different authors to construct different extensions. This includes:

+ incorporating with the model of ‘wallet with observer’ ([Bra93a,b]).
+ building an efficient divisible e-cash ([Oka95]).
+ building an efficient anonymity revocable e-cash ([FrTY96]).

A cryptographic primitive that Brands invented, and had all his construction based on it, is the so called ‘restrictive blind signature’. We will say more on this primitive in section II.4.
II.2.1.3 Pseudonym-based e-cash

In some e-cash schemes ([OO92, Oka95, CPS96, FTY96]) a pseudonym which is not traceable to the customer’s identity, is produced during the account setup phase. In [Yac94, Mao96], each customer has a public-key certificate that is shown to the shop to verify the coin. The customer’s identifier embedded into the customer’s secret key, will be revealed if the customer double-spends the same coin. Using pseudonym or public key certificate results in a simpler mechanism for detection and improves system efficiency. However e-coins are linkable as the bank knows that they originate from the same pseudonym, which is less desirable compared to unlinkable e-coin schemes.

II.2.1.4 Probabilistic random auditing

Recently, some authors ([JaOd97, Yac97]) have considered the mechanism of random auditing based on probabilistic polling: during the payment, with some small probability the shop forwards the payment transaction report to the bank to audit the customer’s spending status ([JaOd97]). This probability parameter can be varied in accordance with the value of payment. This helps to create a hybrid system from the two traditional models – on-line and off-line payment – that takes advantages of both these models.
II.2.2 ‘WALLET WITH OBSERVER’

An innovative off-line model, the so-called ‘wallet with observer’, was introduced in [ChPe92, CrRe93, Bra93a&amp;b, CAFE94, Fe93]. The basic idea in this model is the introduction of an observer inside the user’s module (wallet). The observer, acting as a representative of the issuer bank, observes all the information flow between the wallet and other parties and records all spent coins. The user module needs the cooperation of the observer in the processes of withdrawing new coins from the bank, and in paying coins to the shop. Then if the observer discovers that the user module is trying to reuse an already spent coin, it simply refuses to cooperate and hence the cheating fails. Using this mechanism a prior restrain against double-spending can be introduced which seems to provide a level of security equivalent to the on-line model.

In practice, the concept of using an observer inside the wallet is achieved by using smart-cards with tamper-resistant module. Smart-cards are small hand-held computational devices that can perform cryptographic operations. Here, e-wallet is typically a smart-card with its own display and keyboard which is capable of storing e-coins, and supports all payment functionalities. More information about smart-card and its application to e-commerce can be found in [Dev92] and [GSTY96].
Chaum ([Cha92]) introduced the novel idea of various digital pseudonym cards which store a database of user credentials and can represent the user in various activities.

E-cash systems can be used in both the environments of ordinary shopping and shopping over the Internet. In the first case, the customer goes shopping with a smart-card wallet. A payment transaction is commenced by connecting the smart-card to the shop’s computer to transfer e-coins for the payment. For the ‘wallet with observer’ model, the wallet consists of two modules: user module and observer module - a tamper-resistant device. In the latter case of shopping over the Internet, the user module is the user’s PC and the observer module is an PCMCIA card ([Bra95]).
II.2.2.1 Incorporating ‘detection after the fact’ with ‘wallet with observer’

Brands [Bra93a,b] even built a scheme which was able to combine both the above mentioned mechanisms, namely ‘detection after the fact’ and ‘prior restrain by inside observer’. The ‘detection after the fact’ mechanism acts as a backup solution if the observer is broken. This incorporation is used in CAFE (Conditional Access For Europe), a European project for an electronic wallet solution, supposed to be used throughout the Europe [CAFE94].
II.2.3 EMPLOYING TRUSTED THIRD-PARTY

Recently, there are some proposals [JaYu96, M'Ra96] that employ a trusted third-party in the withdrawal and sometimes in the payment, to protect against double-spending and, at the same time, provide a solution to revocable anonymity (that will be mentioned in the next section). The trusted party, the so called ombudsman in [JaYu96] (or the so called blinding office in [M'Ra96]), cooperates with the bank in the withdrawal such that:

+ Customers and transactions are anonymous w.r.t. (with respect to) the bank or the ombudsman.

+ The blinding process is between the ombudsman and the bank. The coin obtained by customer is blind w.r.t. the bank but not to the ombudsman. When the bank cooperates with the ombudsman, double-spending and potential crimes such as blackmailing, money laundering can be effectively disclosed and stopped.

In this thesis we will present two proposals that also employ trusted third-parties. In the first one, we introduce the notion of Anti-Cheating Servers (ACS), which acts like a ‘consultant’ to the shops to prevent against double-spenders. However, the balance between on-line and off-line consulting is such that the lower value coins are checked ‘more’ off-line with longer delay.
Our second proposal is based on the notion of the *auditing server* that works somehow like the ombudsman in [JaYu96] but does not involves directly in the withdrawal (on-line) and hence is more efficient.

### II.3 Revocable Anonymity

In previous anonymous e-cash systems there is no way to link a coin to its owner as long as the owner has not double-spent the coin. Recently, some authors ([vSN92, CMS96a, CPS96, FrTY96, JaYu96]) have pointed out that perfect anonymity offers potential for committing *perfect crimes* such as blackmailing and money laundering. For instance, a customer may be forced to withdraw e-coins to transfer to an anonymous blackmailer. With perfect anonymity, there is no way to catch the blackmailer if he does not double-spend. This property would deter electronic money from being widely used and obtaining government support.

*Anonymity control* is the main goal of the recent works in electronic cash. Various proposals ([CPS94, CMS96a, CPS96, FrTY96, JaYu96, M’Ra96, JaYu97b]) have suggested revocable anonymity which is the possibility of tracing a customer or a coin, given a court order, while honest customers still enjoy anonymity.

This is often achieved by using a trusted party (trustee). The customer’s ID is encrypted and embedded into the coin in such a way that the coin remains anonymous.
unless both the bank and the trustee collaborate: basically, each party keeps a share of a secret knowledge, which when combined can provide a way of extracting the owner’s ID.

In addition to the set of traditional protocols (withdrawal, payment & deposit), two more protocols are discussed in anonymity revocable e-cash:

+ Owner tracing protocol: This protocol is used to prevent money laundering or illegal purchases as to trace the origin of suspicious coins. Typically, it is required to trace the payer in an illegal purchase (drugs, etc.) if the seller is disclosed. The bank gives the trustee the report it has received during the deposit of the coin. The trustee then return some information that the bank can use to identify the coin’s owner against its client account database.

+ Coin tracing: This protocol is used to trace un-spent coins in case of blackmailing. Typically, the victim of a blackmail alarms the authority after the threat has gone. With the help of the victim, suspicious withdrawal transactions can be singled out from the bank’s withdrawal database. The trustee then uses this information to work out the ‘features’ of the coins withdrawn due to this blackmailing, and informs the shops to blacklist coins.
In [JaYu96], another potential attack that takes advantage of perfect anonymity is introduced - the so called bank robbery. In this attack, the attacker forces the bank to hand out the secret key and produces invalid e-coins. This is an extremely strong attack which is prevented in [JaYu96] by involving another trusted party, called the 'ombudsman', in the blinding process.

### II.4 BLINDING TECHNIQUES

The blinding process is the 'core' of minting electronic coins as it provides coins’ anonymity. An e-cash scheme without blinding process (such as NetCash, [MeNe93]) would hardly be anonymous.

Blind signature schemes were first introduced in [Ch83]. Today, there are many blind signature schemes using different basic signature schemes (RSA, Schnorr, El-Gammar, DSA, etc.).

#### II.4.1 A FORMAL DEFINITION OF BLIND SIGNATURE

A blind signature scheme consists of a 4-tuple of functions

\[(Bl(m, \rho), Sig(m), Unbl(m, \rho), Ve(m, m')).\]

The user blinds the message \(M\) with the blinding factor \(\rho\)

\[M' = Bl(M, \rho)\]

then sends \(M'\) to the signer who computes and returns the signed blinded message
\[ M'' = \text{Sig}(M'). \]

The user then un-blinds \( M'' \) to obtain
\[ M''' = \text{UnBl}(M'', \rho). \]

\( M''' \) is the signed and un-blinded message and can be verified by
\[ \text{Ver}(M, M''') = 1. \]

For short, the 4-tuple of functions must satisfy the following equation
\[ \forall M, \forall \rho \quad \text{Ver}(m, \text{Unbl}(\text{Sig}(\text{Bl}(M, \rho)), \rho)) = 1. \]

**Example: A blind signature scheme.**

The difficulty of computing cube root modulo \( N \) (without the knowledge of \( N \)'s factorization) is used in blind signature [ChFN88]. Alice, instead of giving the hash value of the original message that is to be signed, gives the bank the product of the hash value and a secret factor \( \rho \) (blinding factor):
\[ h(x) \times \rho^{1/3} \]

In return she will get,
\[ h(x)^{1/3} \times \rho \]
and easily divide it by \( \rho \) to get the required signature. That is, she will obtain the cube root of \( h(x) \) modulo \( N \). However, the bank only knows the message \( h(x) \times \rho^{1/3} \) which is untraceable to \( h(x) \) as long as the bank does not know \( \rho \).

**II.4.2 BRANDS' RESTRICTIVE BLIND SIGNATURE**

As mentioned earlier, the early off-line e-cash proposals suffered from the problem of using inefficient cut-&-choose method to solve the problem of encoding an e-coin with the owner's identity. In the Brands' e-cash based on the presentation problem
([Bra93a&b]), an special blind signature scheme, the so called restrictive blind signature, is invented to help to avoid using cut-&-choose method. That is: some internal aspect of a message will not be changed during the blinding process whereas it is still blinded to the bank. This suggests to encode the customer's identity into this internal structure. This internal structure is based on the notion of the presentation problem then may be called presentation structure. In payment the shop will challenge the customer to obtain partial knowledge of the presentation structure. The result is that this presentation structure will be revealed if the customer double-spends the coin and the double-spender will be caught as her identity was encoded into this presentation structure at the withdrawal.

II.4.3 BLINDING BY TRUSTED THIRD PARTY

The above mentioned nice feature of Brand's restrictive blind signature causes the efficiency for his e-cash scheme in [Bra93a&b], however it requires complex constructions. Single-term off-line e-cash following this approach is, then, usually difficult to understand. In [JaYu96, M'Ra96], another approach for blinding is introduced. In these proposals, the customer 'borrows' a trusted third party (the ombudsman in [JaYu96] or the blinding office in [M'Ra96]) to represent her in the blinding process with the bank. Therefore the coins obtained by the customer later are blinded to the bank but not to, say, the ombudsman, whereas the interactive process between the three parties - customer, bank and ombudsman - is deployed
such that the e-coins are anonymous to both the bank and the ombudsman. Hence, overspending and criminals (blackmailing, money laundering, bank robbery) can be traced if and only if both the bank and the ombudsman cooperate.

**II.4.4 Versatility**

Versatility ([JaYu96]) is the property of a basic e-cash model that can easily be extended to provide many desirable features: divisibility, e-cheque payment, credit-card payment, micro payment, surety bond, etc. The customer has the ability to choose and agree with the shop on different options for a payment. For example:

+ Pay for an exact amount up to the total value of a coin.
+ Pay in the forms of cheque or credit card payment (the customer’s identity then needed to be revealed to debit the customer’s account with the paid amount in the deposit).
+ The coins obtained by the receiver can only be cashed if the sender does not follow some certain condition (e.g. to pay rent in time).

These properties can be achieved through a mechanism called challenge semantics, introduced in [JaYu96]. That is to encode certain different bits of a challenge sent by the shop to the customer in the payment phase with some certain facts, conditions, etc. The response of the customer to this meaningful challenge number can be seen as a contract that both sides - customer and shop - agree on. The customer can not
deny any violation which is traced by the authority (bank and ombudsman in [JaYu96]).

Versatility and how to achieve it, as shown in [JaYu96], can be seen as a new methodology which can be applied to various e-cash schemes. The e-cash scheme that, in our opinion, can be extended to achieve versatility, are probably these that:
+ employ a trusted third-party in the withdrawal process to authorize the e-coins;
+ this trusted third-party works in a combined on-line and off-line manner to cooperate with the bank to trace the customer when legally required.

The proposals in [JaYu96, M'Ra96] and our proposal presented in chapter IV satisfy these properties.
III. A FRAMEWORK FOR COMBINING OFF-LINE & ON-LINE E- CASH\(^{10}\)

The two main categories of electronic cash systems are on-line and off-line systems, each with its own advantages and drawbacks. In this chapter we propose a hybrid system that combines the advantages of the on-line and off-line model and achieves a high degree of security and flexibility. We describe the structure of the system and examine the importance of its parameters in providing security.

\(^{10}\) This chapter is on realizing the ideas of 'near prior-restraint' and 'continuum between on-line and off-line payment' that have been mentioned in chapter 2. However we do not use these terminology in this chapter and base the chapter on [NgSa97a]. This helps to keep the chapter as a consistent and self-contained structure, and reflect the way it is naturally developed.
III.1 INTRODUCTION: A FRAMEWORK TO OVERCOME THE DRAWBACKS IN PREVIOUS E-CASH SYSTEMS

III.1.1 DRAWBACKS IN PREVIOUS E-CASH MODELS

Firstly we discuss the drawbacks of the previous off-line e-cash models.

The drawback of Chaum model. The drawback of the e-cash model, that is firstly introduced by Chaum ([ChFN88]), is that unlike the on-line model it does not offer a prior restraint against double-spending. Although, the double-spender's identity can be disclosed and the proper action can be taken afterwards, but the delay may allow the malicious cheater to make vast number of purchases with the same coin that she might have stolen from some other honest user. A malicious cheater can maximize her profit using her knowledge of the time delays in the system. She might even corrupt certain shops to have long delays for depositing coins to the bank so that the cheater succeed in her double-spending attack.

From this discussion one can see that the 'detection after the fact' mechanism used in this model is not effective. The honest users and the ones who do not have detailed knowledge of the system will never try to double spend, while the malicious ones, who have made a careful plan to cheat, can avoid all traps to gain a fortune. In both cases the 'detection after the fact' mechanism seems of no practical use.
The weaknesses of the ‘Wallet with Observer’ model.

1. The future of ‘wallet with observer’ e-cash systems depends on the development in the smart card technology. However, smart cards are not going to be perfectly secure. Trusting tamper-resistant property of smart-cards has been considered problematic and recent research ([AnKu96, BDL96]) is cautioning the community not to carelessly trust the tamper-resistance of smart-cards. Recent news over the Internet are signaling some severe flaws in the smart-card chip of well-known e-payment systems like Master Card, VISA.

2. In shopping over the Internet, the use of an observer chip causes some inconvenience, compared to an all software system. To install an e-cash system, in the latter case, the user only needs to download some software from the Internet, while in the former, he also need to acquire some special hardware.

3. Another disadvantage of employing observer modules is that the user might fear that the observer chip acts as a spy for the manufacturer or the bank, and breaches her privacy ([Fin94]). This may discourage the spread of the e-cash systems based on ‘wallet with observer’.
III.1.2 OUR PROPOSAL

It is commonly admitted that an e-cash system that provides the three main features – anonymity, off-line payment and robustness against double-spending – is not easy to construct. Recently, 'wallet with observer' e-cash is proposed but as it has been mentioned above, this model also has some weaknesses. Some approaches have considered backup systems when the smart-card chip is broken. For example, the e-cash system in [CAFE94] is based on the Brands’ scheme of combining prior restrain by observer and detection after the fact (see II.2.2).

Noting the nature and weaknesses of the on-line and off-line models we propose a hybrid system in which some on-line features are added to the Chaum off-line model. Our framework can be seen as an extension of the Chaum off-line model; alternatively it can extend the 'wallet with observer' model. The new system effectively removes the drawback of the Chaum off-line model and yet does not suffer from the problem of overloading the bank host in the on-line model. The key point is to look at the on-line and off-line systems as the two extremes of a continuum and consider systems that are in between.

In the on-line model, the submitted coin is checked in the real-time against the database of all the spent coins. However, in the Chaum off-line model, coins are submitted to the bank, at an off-peak time. One can say that the delay in double-
spending detection in the former case is zero while in the latter is one day. We propose variable levels of delay for coins, with more valuable coins having less delays and the smaller coins having more delays. For instance, the biggest coin would have no delay (as in the on-line model) and the lowest value coin may have one day delay (as in the Chaum off-line model).

### III.1.3 Off-on-Line

Our hybrid model is obtained by inserting some degree of real-time (or on-line degree) into the double-spending checking phase of the Chaum model. The key idea is to vary the real-time degree in proportion to the coin value. This varying real-time degree helps to optimize the system load: the probability of double-spending is not absolutely removed as in the on-line model but is effectively reduced by making the cost of the attack greater than the potential gain from it. The off-line detection after the fact mechanism is still used as the security back-up: vandals or hostile parties who ignore the attacking cost will eventually be disclosed through this mechanism.

From now on we use the term ‘off-on-line’ to refer to our proposed method of combining off-line and on-line checking. The off-on-line scheme is implemented through the deployment of a network of Anti Cheating Servers (ACS), which replaces the central host in the previous models. In the payment phase, the shop contacts an ACS to check the coins submitted by a customer. The ACS does not
have the same power as the central host in the previous schemes (storing the whole history database of spent coins) but it does have some regional information which can be used to detect the repeated attempts to spend a coin in the region covered by that ACS. We assume coins are classified into value-levels based on their values. We use the term “value-level” to refer to these levels. For example, a coin of $200-$1000 corresponds to level 1 and a coin of $50-$199 corresponds to level 2. According to the value-level of the coins, more or less resources of the ACS network will be used to check the coins, and hence with a proportional probability the double-spender is caught in real-time. Checking tiny value coins, e. g. 1-10 cents, will take no resource of ACS network as the shop does not contact any ACS and deposits the coins at the end of day (as in the Chaum off-line model). Meanwhile checking big value coin, such as a $1000 coin, will take maximum resources and the effect is equivalent to on-line checking in the on-line model.

The rest of the chapter is structured as follows. In the next section we will introduce the conceptual model of the proposed system and will provide some notations. System specification will be given in the section after the next, with a numerical example for illustration. Finally, we will evaluate the system and present the chapter’s conclusion.
III.2 CONCEPTUAL MODEL

III.2.1 THE ANTI CHEATING SERVER

In our system the bank does not rely on a single central host. Rather, the workload is distributed among multiple Anti Cheating Servers (ACS), each of which is located in a region, having connections to all the shops in that area. At each ACS, a blacklist of multiple-spent coins is kept. Whenever a shop receives a coin from a customer, it transfers the coin in real-time to the local ACS, which will check the coin against the blacklist to find out if any other cheating has occurred. All the just-double-spent coins will be discovered by Check-at-the-end-of-the-day as in the Chaum off-line model. Moreover, for very high value coins, the on-line checking is performed on the whole database of spent coins instead of checking against the blacklist to stop any double-spending of big value coins.

In general, the second time spending of the same coin may be successful but after that, exchanging information among the ACSs will disclose the cheating and will insert the coin into a blacklist, which is kept in all ACSs, ensuring that further attempts of a greedy cheater will be stopped!
III.2.2 **Hierarchical Network of ACS**

The network of ACS is organized as a hierarchy with multiple levels of ACS. The lowest level includes all the ACS which have direct connection to the shops. These ACS locations are at the lowest branches of the bank which are, say, of **district level**. All the shops in the *district* are connected to the *district ACS*\(^{11}\). The higher levels are at state and the national levels, and at the very top level, is just the central host.

While the central host maintains a *history database* that stores all the spent coins of the whole system life period, each ACS at lower level maintains a local database which stores spent coins of the day and is reset to empty at the beginning of everyday. The *off-on-line* principle is supported by the following points:

- The micro coins are not submitted in real-time from the shops to the district ACS (off-line submission).
- Small value coins are submitted immediately (on-line) to the district ACS.
- Higher value coins, are submitted to district ACS first, then if they are not found in the blacklist, they are forwarded from the district ACS to the ACS at the higher levels, depending on their value.
- Biggest value coins can go as far as the top level ACS.

\(^{11}\) The *district ACS* denotes the ACS at the district level
At each level, coins are checked against the coins submitted earlier in the day in the region (district, state ...) corresponding to the level of that ACS. In this way, double-spending in the region of districts, state, ... will be detected and updated into the blacklist. Now suppose a cheater has stolen a coin and wants to spend that coin many times. If the value-level of the coin is such that it is stored at state level, she can spend her coin at most twice in a state and then she has to travel to the next state for the next spending of the coin, but of course this cheating will eventually be disclosed by Check-at-the-end-of-the-day.

III.2.3 Using hash value

The size of a coin in off-line schemes is usually not less than 100 bytes. This is not only to protect against cryptanalysis, but also to allow storing information about the user’s identity inside the coin. The coin submission requires even more data because it includes information produced in the challenge-response phase at the shop when the customer goes through the payment process. We propose to transfer and manipulate digital coins in full and in compressed form both. This allows more efficient storage and checking of the coins. We will use hash functions to produce digests of coins. A hash function is a function that maps an arbitrary length sequence to a fixed length sequence (usually a much shorter). The output of a hash function, or hash value, is similar to a fingerprint of the input and although takes much less space, it can be used to represent the original string. A collision happens when the hash...

III-48
values of two different inputs are the same. For our purpose, a simple hash function such as division hash, is sufficient.

The submitted coin is stored in full at the district ACS, where it is submitted by the shop; after that it is hashed into a digest and then forwarded to a higher level ACS. All the transfer and computations are performed on the hash values (of course, IDs and tracing information will be added in transfer). At a high level ACS, if a collision is found, to determine whether it is by accident or a true double-spending, the high level ACS can ask the district ACS the full information.

Here is an example that illustrates the usefulness of using hash value. Codes of 8 bytes (or 64 bits) length can present $2^{64}$ different numbers. Let say, a fixed coin value (denomination) can not appear in the shopping over a city of millions people, more than 1 million ($<2^{20}$) times in a day. The probability of at least one collision, according to ‘Birthday paradox’, can be estimated as:

$$P(m,k) = 1 - e^{-\frac{k(k-1)}{2m}} = 1 - e^{-\frac{2^{20}(2^{20}-1)}{2 \times 2^{64}}} < 1 - 1/e^{2^{25}} = 1 - 1/(1 + 2^{-25}) < 2^{-25}$$

That is, 8 bytes for hash value provides enough protection against accidental collision, while the size of e-coins is usually in hundreds of bytes.
We note that the size of the hash value (output of the hash function) should be different for different value-levels of coins. One can see that coins of different values have different frequency of occurrence (and hence different probability of accidental collision of hash value). The small value coins occur more often than the big value coins. Consequently, the size of the hash value space for small value coins should be larger than those for bigger value coins.

### III.2.4 Definitions and Notations

The system can be seen as an extension of the Chaum model with the shop contacting the local ACS in real time to have verification of a submitted coin. We will omit details of the Chaum model such as how to blind a coin or how to withdraw a coin. Instead, we concentrate on the architecture and the work of the network of ACS, which is the new feature of the new framework and extends the original Chaum model.

The ACS network can be seen as a tree of nodes consisting of ACS. The structure of the network and relevant definitions are given below:

- The leaves of the tree are *district ACS*, which are located at the district branches of the bank. All the shops in the district have a live connection to the corresponding district ACS.
• The central host is located at the root of this tree network.

• State ACS is a non-leaf node of this tree network. I-ACS denotes an ACS at level I.

• NUM-LEVELS denotes the number of the levels of this tree. Obviously, NUM-LEVELS is a key parameter of the system.

• History database denotes the database managed by the central host and stores all the coins spent during the lifetime of the system.

• District database denotes a database managed by a district ACS, which stores all the coins submitted during a day from all the shops having connection to that district ACS.

• Blacklist denotes the list of coins which have been detected as double-spent at a certain time. Some double-spent coins may be found at a later time, for example, small value coins that are double-spent only twice in two different states. These coins will be eventually caught by Check-at-the-end-of-the-day. Each district ACS has a copy of this blacklist and every time a new coin is detected as a black coin it will be sent throughout all the district ACS to update their copies of the blacklist.

• Ancestor list of a district ACS denotes a list of ancestor nodes of that district ACS. A node named A, is an ancestor of a node named B if cutting the tree at the point just above node A results in a tree with A being the root and B still being an internal node. Each district ACS keeps its own ancestor list.
• Corresponding to the tree structure of the network of ACS, the coin value also is divided into many levels (coin value classification). The number of coin value levels is $NUM-LEVELS + 1$. The network of ACS can also be seen as a tree of $NUM-LEVELS + 1$ levels, if shops are also counted as nodes of the network.

• $I$-coin denotes a coin at level $I$. For example, 1 cent may be a $(NUM-LEVELS+1)$-coin while $1000$ may be a $1$-coin. The level of a coin determines the level of the ACS tree network to which the coin finally reaches during a purchase.

• $Check-at-the-end-of-the-day^{12}$ denotes the checking process at off-peak time when all the coins have been submitted to the central host and are checked against the whole history database. Because of the coin value classification, $Check-at-the-end-of-the-day$ for different coin value levels is performed at different time intervals (shorter intervals for higher value coin).

• $EndDayCheckTimingFrame[1..NUM-LEVELS+1]$ is a system parameter which denotes the list of the values of different end-of-day checking timers, corresponding to coin levels from 1 to $NUM-LEVELS+1$.

• $Hash-Function-List[1..NUM-LEVELS+1]$ is a system parameter. $Hash-Function-List [I]$ corresponds to the coin value level $I$

---

12 The phrase ‘end-of-day’, is used to imply the off-peak time for periodically checking as is not necessarily at the end of the day.
Figure 7. The conceptual model

III.3 THE SYSTEM

III.3.1 SPECIFICATION

Here we describe the algorithms which are used when:

a. The customer submits a coin (I-coin) to the shop.

Depending on the value level I, one of the following procedures will be performed:

a.1. Procedure for tiny value coins (I=NUM-LEVELS + 1)

a.2. Procedure for small-medium value coins (NUM-LEVELS ≥ I > HIGH-LEVEL)
HIGH-LEVEL is a system parameter, indicating the least level of coin value at which a coin can be considered of high value.

a.3. Procedure for high value coins: \( \text{HIGH-LEVEL} \geq I \geq 1 \)

b. The Check-at-the-end-of-the-day is performed.

The following is the specification of these 4 procedures. The procedure for small-medium value coins is presented first and then the others because they refer to the former.

Each procedure includes a number of steps. Each step is described informally (by natural language) and then more formally, using a pseudo-code style. In the pseudo-code description we follow the common style in programming languages. For example,

\[
\text{X\rightarrow Y: info} \quad \text{Message info is transferred from X to Y}
\]

\[
\text{if, then, else, stop, for...} \quad \text{keywords}
\]

\[
\text{Open, Check, Store in, ...} \quad \text{commands}
\]

\[
\text{CANCEL, OK, NIL} \quad \text{defined constants}
\]

\[
\{ \text{text} \} \quad \text{comment}
\]

\[
\text{INFO-REQUEST (coin-ID)} \quad \text{message INFO-REQUEST with parameter coin-ID}
\]
III.3.1.1 Procedure for small - medium value coins (A)

1. When the customer submits a coin to the shop, the shop starts a new communication session with the district ACS and transfers the submitted coin to that ACS.

C {The customer}

\[ C \rightarrow S: cs \]

\{ cs - coin submission = coin + customer's response to the shop's challenge \}

S {The shop}

\textit{Open} comm-session

\{comm-session: a communication session to DA\}

\[ S \rightarrow DA: cs \]

2. The ACS checks the coin against the blacklist. If it finds that the coin is already in the blacklist then it will alert the shop to cancel the purchase and catch the cheater-customer. Otherwise the shop is allowed to proceed with the purchase.

\{2.1.\}

DA {district ACS}

\textit{Check}(cs, blacklist)

\textit{If} 'found' \textit{then}
DA $\rightarrow$ S: CANCEL

\{ message to cause the shop cancel the purchase\}

Close comm-session

Stop

{2.2.} Else

DA $\rightarrow$ S: OK

\{ message to cause the shop accept the purchase\}

Close comm-session

End-If

3. Depending on the coin value, a hash function is selected and the hash value of the coin is produced. The ACS checks the coin against the district database. If ‘found’, the coin will be broadcast to every district ACS as a black coin and all the district ACSs will update their blacklist copies. Otherwise, the submitted coin is stored in the district database with the coin ID (coin ID can be generated by using a counter, which increases by one every time a coin is submitted to the district database) and the hash value.

DA

Hash = Hash-Function-List[LevelOf(cs)]

Check (Hash(cs), DA-database)
If 'found' then

DA $\rightarrow$ All DAs: IS-BLACKCOIN(cs)

Else

Store (coin ID, hash(cs), cs) in DA-database

End-If

4. If the coin is not of district level, an ancestor ACS at appropriate higher level (state, nation ...) is determined, then the triple, coin hash value, coin ID and the district ACS’s ID, is forwarded to that ACS.

DA

higher-level = LevelOf(cs)

If higher-level=NUM-LEVELS then

Stop

End-If

SA = Ancestor-List[higher-level]

{ Ancestor-List[1..NUM-LEVELS] indicates the ancestors of the district database,
Ancestor-List[1] is the central host, Ancestor-List[NUM-LEVELS-1] is the father state ACS, Ancestor-List[NUM-LEVELS] is the district ACS itself }

DA $\rightarrow$ SA: forward-info = (hash value, coin ID, DA’s ID)
5. At that higher-level ACS, say the state ACS, the forwarded coin is checked against the state database. If ‘found’ then the state ACS will ask the corresponding district ACS for full information, which will determine whether this is a true double-spent coin or not. If yes, alarms will be sent to every ACSs to update their blacklists. Otherwise the state database is updated with the new information.

5.1

SA {state ACS}

Check (hash(cs), SA-database)

If ‘found’ then

Extract DA’s ID from forward-info {of collision-making coins}

SA → DA1 : INFO-REQUEST (coin-ID)

SA → DA2 : INFO-REQUEST (coin-ID)

{for information of colliding coins in full-form (ie. cs)}

DA1 → SA : cs1

DA2 → SA : cs2

5.2

If cs1[coin]=cs2[coin] then

SA → All DAs : IS-BLACKCOIN(cs1)

Stop

End-If
III.3.1.2 Procedure For tiny value coins (B)

The shops do not transfer the tiny coins in real-time, but temporarily keeps them in their own database. At Check-at-the-end-of-the-day, all shops transfer the coins to their district ACSs, which are then forwarded to the central host.

C

\[ C \rightarrow S: cs \]

S

Check (cs, shop-database)

If ‘found’ then

\[ S \rightarrow C: CANCEL \]

Else

Store cs in shop-database

\[ S \rightarrow C: OK \]

End-If

Stop
III.3.1.3 Procedure for big value coin (C)

The process is similar to the process for small-medium value coins, with the following modifications.

1. Same as step 1 of Procedure for small-medium value coins, called Procedure A.

2. Same as step 2.1 of Procedure A.

3. Same as step 3 of Procedure A.

4. Same as step 4 of Procedure A.

5. 

5.1 {Same as step 5.1 of Procedure A.}

SA {state ACS}

Check (hash(cs), SA-database)

If 'found' then

Extract DA's ID from forward-info {of collision-making coins}

SA → DA1 : INFO-REQUEST (coin-ID)

SA → DA2 : INFO-REQUEST (coin-ID)

{for information of colliding coins in full-form (ie. cs)}

DA1 → SA : cs1

DA2 → SA : cs2

5.2. {different from step 5.2. of Procedure A}
If \( cs_1[\text{coin}] = cs_2[\text{coin}] \) then

\[ SA \rightarrow \text{All DAs} : IS\text{-BLACKCOIN}(cs_1) \]

\[ SA \rightarrow DA \rightarrow S : CANCEL \]

\[ DA : \text{Close comm-session} \]

Stop

End-If

End-If

Store forward-info in DA-database

\[ SA \rightarrow DA \rightarrow S : OK \]

Close comm-session to S

Stop.

### III.3.1.4 Procedure for Check-at-the-end-of-day (D)

For level I of coin value, there is a corresponding I-timer for check-at-the-end-of-the-day. These timers are common for all ACSs. When an I-timer strikes, all district ACSs transfer the part of their district database that only includes I-coins to the central host which accomplishes the Check-at-the-end-of-the-day for this level I of coin value. After that, parts of the blacklist, which stores information about the I-coins and I-ACS databases can be reset to empty.

For all \( I = 1 \), \text{NUM-LEVELS}
When timer(EndDayCheckTimingFrame[I]) strikes

All DAs → Central-Host : DA-database(I)

For All DA: DA-database(I) = NIL

For All DA: blacklist(I) = NIL

For All I-ACS : I-Database = NIL

End

End

For every shop, when timer of Check-at-the-end-of-the-day for tiny value coins strikes, the shops must transfer their database of tiny coins to the corresponding district ACSs, which in turn forward the data to the central host.

For all S

When timer (EndDayCheckTimingFrame[NUM-LEVELS+1]) strikes

S → DA→ Central-Host: shop-database

shop-database = NIL

End

End

III.3.2 System Parameters

The system has the following parameters:
• NUM-LEVELS, an integer, defining the number of levels of the coin value, the number of levels of tree of ACSs and the number of hashing functions employed in the system. This is a key parameter determining the system scale and the workload distribution.

• HIGH-LEVEL, an integer, defining the separation between medium and high value coin, allowing the high value coins to be treated differently: on-line checking at the ACSs at the level corresponding to their value level.

• LevelOf(), a function, which takes a coin value as input and returns the level of that coin value as output; it determines the detailed ranges of coin value for each level.

• Hash-Function-List[1..NUM_LEVELS+1], an array, whose elements are hash functions employed in the system and defines which hash function is coupled with which level of coin value.

• EndDayCheckTimingFrame[1..NUM_LEVELS+1], an array, whose elements are check-at-the-end-of-day timers for each coin level.

These system parameters together with the geographical distribution of ACSs are the design decision to be made, determining the security and economy of the system. ACSs should be distributed based on the population density, which is assumed to determine the business activities. Given the geographical map of a country, the population map, and shopping statistics, one can decide how many levels (NUM-
LEVELS) of coin values should be used and the detailed installation of the ACS network can be worked out.

An important feature of the system is that a cheater who holds an I-coin must spend that I-coin no more than twice in an I-region of an I-ACS, otherwise she would be caught. So the cheater has to travel I-region by I-region to spend the coin. This leads to the main design principle of ACS network: The average I-region should be large enough to make the cost of traveling between two I-ACSs more than the benefit of spending that I-coin two more times.

The employed hash functions highly affect the economy of the system. Hash functions must be chosen such that their output size satisfy the following condition: the probability of two different coins, bearing the same value, and having the same hash value be adequately small.

The parameter EndDayCheckTimingFrame also affects the economy of the system. This parameter should be chosen considering the severity of the assumed loss for each level of coin value. Small coins of say 5 cents can be delayed for quite long time, say 2 weeks, while big coins of $1000 or more should be off-line checked more often (say every 2 hour). We note that the occurrence frequency of the 5 cents coins is far
greater than that of the $1000 coin, and so a well-designed

EndDayCheckTimingFrame will produce a more economical solution.

III.3.3 Example

A city has a population of 5 millions people. Assume there are 50,000 shops (high bound), that is one shop for every one hundred people. Assume the shops are using e-cash system. Further assume:

- A coin size is 100 bytes.
- A payment requires on average transfer of 5 coins to the central host (e.g. the amount of $135.50 can be paid by 1 coin of $100 + 1 coin of $20 + 1 coin of $10 + 1 coin of $5 + 1 coin of 50 cents). This costs 500 bytes storage of data.
- Each shop has on average 100 purchases/hour.

If this e-cash system is based on the on-line model, the storage of data sent to the central host in one day is:

$$0.5 \text{ Kbytes} \times 100 \text{ (purchases)} \times 24 \text{ (hour)} \times 50,000 \text{ (shops)} = 60 \text{ (Gb)}.$$ 

This is just for one day. After one year no single host can easily manage this database.

In our system, one can employ 50 district ACS, each of which working for one thousand shops. Every day the amount of stored data in a district ACS is:

$$0.5 \text{ Kbytes} \times 100 \text{ (purchases)} \times 24 \text{ (hour)} \times 1,000 \text{ (shops)} = 1.2 \text{ (Gb)}.$$
This can be stored in a typical PC hard disk (the district database is reset to empty after each day). The ACS can be just a normal PC. The hardware requirement is trivial in here.

III.3.4 POSSIBLE ATTACKS

III.3.4.1 Organization cheating

The enemy can be a hostile organization or a well-organized group of vandals. The enemy can deploy many agents in many different region, say states. Then after successfully stealing some e-coins at the state level or lower, these e-coins can be electronically sent to all the members of the group who will spend them just twice and hence will not be caught. This attack is beyond the real-time defense of the system and can be considered as a potential danger, although the cheating will be eventually disclosed by the Check-at-the-end-of-the-day. However, to successfully damage the system, the number of stolen coins must be very large (note that the very high value coins are still safe because of on-line checking at the central database). This can not be done just by some coins accidentally picked up, but requires systematized robbery which will put the enemy into a risky situation with serious penalty. Hence, this kind of attack would be very rare.

It is commonly admitted that perfectly secure systems only exist in theory. To obtain a practical system one must sacrifice some of the security. The important point for
the designer of a system is to design the system and its parameters such that it effectively limits the risk on the participants’ asset. In our case, the system parameter EndDayCheckTimingFrame[1.. NUM_LEVELS+1] should be chosen carefully to limit the double spending of high value coins. Moreover, such parameter can be dynamic: when a series of cheating is detected in certain regions, backup resource on the ACS network can be used to get the EndDayCheckTimingFrame down in these regions, say the coins at the state level be end-day checked every 2 hour instead every 12 hour; hence the cheaters can be caught.

III.3.4.2 Common boundary effect

The advantage of the system can be directly seen from the off-on-line principle, namely:

*The amount of system resource used to check double-spending of a coin of a certain value is proportional to the coin value.*

This is achieved by the introduction of a hierarchy of ACS. However this hierarchy, as shown above, causes a weakness in the system. A cheater who holds a stolen I-coin can spend it up to 2*N times (with a small traveling cost) around the common boundary of N I-regions. If the knowledge of such black area is made public, the loss could be even more severe.
A possible solution to this problem is to require an I-ACS to transfer all its information about a new submitted I-coin to its neighboring I-ACS.

**III.3.5 Extension for Shopping over the Internet**

The framework can be easily implemented for ordinary shopping activities with the customer coming to the shop and using some kind of electronic wallet to pay. The electronic wallet can be a dumb-card (like a prepaid phone card) or a sophisticated smart-card with its own display and keyboard for protection against possible cheating by the shopkeeper at the point-of-sale ([GSTY96]).

However, the framework can not be directly applied to shopping over the Internet. In shopping over the Internet, the geographical factor becomes less important as the customer can visit shops at any distance. The idea of the shop being tied to an ACS is hence less effective, although in practice the shop delivers the goods on the next day and after the *Check-at-the-end-of-the-day* that allows the shop to detect double-spent coins and catch the cheater before the delivery. In the case of shopping over the Internet, the ACS can be tied to the user through her long-term IP address. The idea is drafted in the following.

An ACS is established for each Internet domain. When the customer visits a shop and buys some goods or service, the shop will obtain the IP address of the machine which...
the customer is using to access the Internet\textsuperscript{13}. Using this IP address the shop can learn the Internet domain to which the user machine belongs, and hence can contact the right ACS which is tied to all the IP addresses of that domain. From now on, the system is the same as for ordinary shopping. The network of ACS is built up according to the domain structure of the Internet. For example, if the framework is implemented in Australia, the central ACS will be ACS.au. A customer accessing the Internet at the domain cs.uow.edu.au, will have her coins submitted to ACS.cs.uow.edu.au, ACS.uow.edu.au, ACS.edu.au and ACS.au, depending on the coin value. Consequently, the cheater who wants to multiply spend a stolen coin must manage to obtain multiple Internet accounts belonging to different low-level domains. The cheater can not avoid checking by high-level domain such as com.au, edu.au ..., which means the cheater can not double-spend high value coins. This clearly reduces the chance of the cheater and the presumed loss of the issuer bank, maintaining an acceptable level of risk.

\textbf{III.4 CONCLUDING REMARKS}

We have proposed a framework for extending previous e-cash schemes achieving a higher level of security against multiple-spending.

\textsuperscript{13} The IP address of the sender machine is added in every IP packet by IP protocol to allow the receiver machine to answer.
The system has the basic structure of the old systems but includes, besides the 3 traditional entities, the user, the shop and the bank, a new entity - the ACS (Anti Cheating Server). The complete set of protocols for withdrawal, payment and deposit, is extended with the procedure for the shop contacting the local ACS.

The development of the ACS hierarchical network, the use of hash value and the classification of coin values, result in strong and efficient protection against double-spending. A malicious attacker with even expert knowledge could not effectively cheat in the new system. The new system is also efficient from the point of view of minimizing the workload. For example, a cheater has no way of double-spending a $1000 coin but it is allowed to multiple-spend a coin of 50 cents at most twice in a district. However in this case the traveling cost to other districts would be more than 50 cents and hence the attackers is effectively stopped. The higher the value of the coin is, the less is the probability of being minted and appearing in public. The higher value coins result in more work in the system than the smaller value coins but they appear less frequently.

Extending Chaum's off-line e-cash scheme using this framework will offer a high level of security against multiple-spending which can even be compared to an on-line
model. Meanwhile there is no need for special hardware and so the system is easier to implement!

This framework can also be applied to the ‘wallet with observer’ model. This provides an alternative for those who are using smart-card technology and now seek higher security after recently announced attacks on smart-cards.

It is commonly believed that different e-cash schemes should be used for different classes of coin values; that is, micro-payment for low value, off-line e-cash for middle value and on-line e-cash for high value. This framework helps to optimize the coin value denominations and allows enlargement of the range of coin values used in a given system.
IV. E-CASH WITH ‘MIRROR WALLET’

In this chapter, we propose a new architecture that is highly efficient and provides all the desirable properties of an e-cash system such as revocable anonymity and loss recovery. The proposal is based on mirror wallets that are network objects that store customer’s transactions and are transparent unless an illegal activity is committed by a customer. The system allows a trade-off between customer privacy and legal traceability and incorporates the notion of off-on-line, introduced in [NgRa97], to achieve a trade-off between protection against double-spending and system efficiency.

Our proposal is versatile and provides features such as coin divisibility, overspending detection and electronic check. Finally, we consider, and succeed to control the potential risk of trust abuse where an insider in the trusted third-party (auditing server) is corrupted.
**IV.1 The basic model with new notions**

In non-anonymous e-cash systems, a coin consists of a public part ($p_{\text{coin}}$) and a secret part ($s_{\text{coin}}$). The customer pays the shop by handing over $p_{\text{coin}}$ and then uses $s_{\text{coin}}$ to generate the response to the shop’s challenge. $p_{\text{coin}}$ can be revealed to public but only the customer can use it for payment because of the secret $s_{\text{coin}}$. For anonymous e-cash, $p_{\text{coin}}$ is also kept secret before the payment. By using blind signature in withdrawal the bank can not learn $p_{\text{coin}}$.

We devise the notion of *mirror wallet* which is a database object storing values of $p_{\text{coin}}$. Mirror wallets are created when customers start using the system (in the Wallet Setup protocol) and are stored at a trusted third party called *auditing server*. Two different wallet serial numbers (WSN), $b_{\text{WSN}}$ and $c_{\text{WSN}}$, are generated by the bank and the customer (user), respectively, to identify the wallet in communicating with the auditing server. If the bank does not know $c_{\text{WSN}}$ then it can not trace the customer’s payment.

The auditing server is involved in the withdrawal protocol, however, the involvement is non-interactive and off-line. Having finished the normal withdrawal process, the customer and the bank both send their withdrawal views to the auditing server (including the blinding factor from the customer). This can be done off-line but needed to be before the customer’s spending of the coin. The auditing server will
check the customer’s report of withdrawal \((p\_coin\ \text{inside})\) against the bank’s report and if it does not match, alarm is sent to the bank with \(b\_WSN\) to identify the dishonest customer. Otherwise it stores the combined withdrawal report into the corresponding mirror wallet.

During the payment, the customer sends the \(p\_coin\) to the shop with \(c\_WSN\) encrypted with the auditing server’s encryption key. After verifying the bank’s signature on the \(p\_coin\) and the customer’s knowledge of \(s\_coin\), the shop forwards the payment view to the auditing server, who will check the coin against the \textit{blacklist}\ (list of \textit{black coins} that are either double-spent or suspicious as originated from criminals) and against the corresponding mirror wallet. We separate these two parts of checking (by the shop and by the auditing server) and introduce an auditing phase which can be dealt with separately. Then normal method of payment is off-line and on-line payment is used (with auditing process done directly in real-time) only if the shop is alarmed to do so (for tracing illegal coins).

Revoking anonymity is easily achieved in this model. We provide a \textit{coin tracing} and \textit{owner tracing} protocol. Coin tracing is used against crimes such as blackmailing: the blackmailed customer authorizes the bank to contact the auditing server and then get all the un-spent coins from the mirror wallet. After checking against the deadline of the blackmailed withdrawal, suspicious coins are blacklisted: the auditing server
updates the blacklist and all the shops are alarmed to use on-line payment. Owner tracing is used against crimes such as money laundering: The bank sends the suspicious deposited coins to the auditing server that checks them against the mirror wallet database. Once a mirror wallet is found matched, the corresponding $b_{WSN}$ is sent to the bank to identify the owner of suspicious coins.

The notion of auditing server with mirror wallet is similar to the notion of ombudsman in [JaYu96] but is different and independent. The involvement of the auditing server is non-interactive and off-line while the ombudsman takes an interactive role in the withdrawal. Moreover withdrawal views from all customers are unordered in the ombudsman’s database, while are ordered by mirror wallets in the auditing server’s database and help to speed up the checking process and provides a precious property: wallet loss recovery. The cost to be paid for all of this is stronger trust on the auditing server, who will be able to have all the customer transactions revealed if the bank, or the shop, accidentally or deliberately reveals the customer name in even one transaction. However, as mentioned earlier, this is a starting model which is furthered developed in the subsequent sections to reduce trust on the servers and provide protection against collusion attacks.

**IV.2 The Scheme**

In the following, we present a realization of the above model.
IV.2.1 Notations, Message Formats and Cryptographic Components

Players are denoted by C (customer, user), S (shop), B (bank), AuS (auditing server).

A message numbered $i$ sent from $Src$ to $Dest$ in phase $T$ is denoted by $m_i^T(Src, Dest)$. Phase $T$ can be, $W$ for withdrawal, $P$ for payment, $A$ for auditing or $D$ for deposit.

We often write the equations $y = f(x)$ or $y = f(x, \rho)$ in another form

$$y = \{x\}^f \quad \text{or} \quad y = \{x\}^f$$

To emphasize on the data (messages $x$, $y$) rather than the function. Also $y = \{x\}_{K_A}$ and $y = \{x\}_{K_A}$ denote a message encrypted with the public key and with the secret key of player $A$, respectively.

IV.2.1.1 Partial blinded signature scheme - How to date blind signature

As mentioned in chapter 2, a blind signature scheme consists of a 4-tuple of functions

$$(Bl(m, \rho), Sig(m), UnBl(m, \rho), Ver(m, m'))$$

that satisfies

$$\forall M \forall \rho \quad Ver(m, \{\{m\}^Bl_{\rho}\}^{Sgn}_{\rho})^{Unbl}_{\rho} = 1.$$  

Partial blind signature schemes, introduced in [AbFu96], is a special blind signature scheme in which a part of the message is kept unprocessed during the blinding phase and is readable by the signer. This part of the message is usually some useful common information such as time-stamp (in applications such as E-cash, it helps to set up parameters such as coin expiration). The scheme guarantees that none of the
parties (the user and the signer) can cheat by altering this common plaintext information without being detected by the other. Accordingly, the 4-tuple of functions is

\[ (Bl(m, c, \rho), \text{Sig}(m, c), UnBl(m, \rho), \text{Ver}(m, c, m')) \]

which satisfies:

\[ \forall M \forall c \forall \rho \quad \text{Ver}(m, c, \{ \{ m \}_{c, \rho}^{Bl}, \{ \text{Sig} \}_{c}^{Unbl} \}) = 1 \]

with \( m \) is blinded part and \( c \) is the clear part (common information).

For the sake of brevity, we will refer to this component as [P-Blind Signature].

**Example:** [P-Blind Signature] can be employed to realize the e-cash model with multiple deposit servers in II.2.2, that is to bind a customer’s (C) deposits to a particular deposit server (D). This can be done by using [P-Blind Signature] in the withdrawal protocol such that the \( c \) above consists of the identifier of the deposit server D. Hence, in the payment phase the shop knowing \( c \) can deposit the e-coin to D properly.

**IV.2.1.2 Collaborative generation of a pseudo-random number**

This problem is often stated as how two parties can collaborate in tossing a coin.
A method for achieving this goal is to use Diffie-Helman key agreement protocol with extension to protect against *man-in-the-middle attack*. We will refer to this component as [Co-Tossing].

**IV.2.1.3 Coin construction**

The coin consists of 2 components: secret and public - denoted by $s_{\text{coin}}$ and $p_{\text{coin}}$ respectively: coin = ($s_{\text{coin}}$, $p_{\text{coin}}$). The user presents $p_{\text{coin}}$ to the shop and proves to the shop (by challenge-response method) that she knows $s_{\text{coin}}$. This can be done as follows:

+ The user has an RSA signature scheme with modulus $N=P*Q$; $P$ and $Q$ are large primes and are kept secret.

+ The user generates a random number $s$ such that $\gcd((P - 1)(Q - 1), s) = 1$ and will keep it as the secret $s_{\text{coin}}$. He then computes $p$ as the public key corresponding to the secret key $s$: $p = F(s, p, q)$.

+ The user contacts the bank during the withdrawal phase and asks for the bank's signature on $p$. Now $p_{\text{coin}}$ is the signed $p$.

+ The shop challenges the user by sending a random number $r$. The user responds by $y = \{r\}_{s_{\text{coin}}}$ which the shop can verify by using $p$ in the $p_{\text{coin}}$.

Clearly the shop can not frame the user by forging the user’s transactions because only the user can generate the correct $y$ for a randomly chosen $s$ such that the pair $(r, y)$ can be verified by using $p$ in the $p_{\text{coin}}$. 

IV-78
We will refer to this component as [Coin-Construction].

**IV.2.2 Protocols**

We assume that the communication channels between any two parties are authenticated and cannot be eavesdropped. The bank has a digital signature scheme with the key pair \((K_b, K^*_b)\). The auditing server has a public-key encryption scheme with the key pair \((K_{AuS}, K^{'AuS})\).

**IV.2.2.1 Wallet Setup**

1. C and B collaboratively generate a random \(r_1\) ([Co-tossing]) then each computes
   \[ b_{WSN} = f\left(\langle user\_ID \rangle, r_1\right) \]
   where \(f\) is a one-way collision-free hash function.

2. B sends a *request-to-open-new-mirror-wallet* to AuS with \(b_{WSN}\). AuS creates a new mirror wallet \(m\_wallet\) linked to \(b_{WSN}\).

3. C contacts AuS and shows \(b_{WSN}\). C and AuS collaboratively generate a random \(r_2\) ([Co-tossing]). Each computes \(c_{WSN} = f\left(b_{WSN}, r_2\right)\).

The Wallet Setup is finished with B end up knowing \(b_{WSN}\) linked to the customer bank account. AuS knows the link \((m\_wallet, c_{WSN}, b_{WSN})\) just for internal use and knows nothing about the customer’s real ID information. The customer knows \(c_{WSN}\) to contact the auditing server later on.
IV.2.2.2 Withdrawal

Withdrawal has two distinct stages. In stage A, the customer works with the bank to generate a coin. In stage B, both the bank and the customer report their withdrawal views to the auditing server who will verify and store the complete view into the mirror wallet associated with the customer.

Stage A. Withdrawal of the coin:

In this stage the customer works with the bank to produce the coins.

1. C generates a unique random number $s$ ($s_{coin}$) and computes $p = \{s\}_{N, r, Q}^F$ using the function $F$ introduced in [Coin-Construction].

C sends a request-to-withdraw message to the bank. The message consists of

\begin{align*}
\text{C\text{-}B}: m_1^w (C, B) &= (\text{bld\_p, c\_inf, customer\_ID}) \\
& \text{with } p = \{s\}_{N, r, Q}^F \\
& \text{bld\_p} = \{p\}_{c\_inf, \rho}^{bl}
\end{align*}

Here $p$ corresponds to $m$ and $c\_inf$ corresponds to $c$ in [P-Blind Signature], respectively. $\rho$ is the blinding factor.
2. B receives $m^w_1(C, B)$, debits the customer’s account by the value, checks if $c_{inf}$ has been correctly constructed, then signs $bld_p$ and sends the signed message to C:

$$B \rightarrow C: m^w_1(B, C) = (sig_{bld_p})$$

with $sig_{bld_p} = (bld_p)^{sig^p}_{K^s_B}, K^s_B$ is the secret-key of the bank.

3. C receives $m^w_1(C, B)$ and then un-blinds and verifies $sig_{bld_p}$:

$$sig_p = (sig_{bld_p})^{UnBl}_{c_{inf}, p}$$

$$Verify(p, c_{inf}, sig_p) = 1$$

Now the customer has the $coin = (s_{coin}, p_{coin}) = (s, p|sig_p|c_{inf})$.

Stage B. Report to the auditing server

4. B sends its view of the withdrawal transaction to the AuS:

$$B \rightarrow AuS: m^w_1(B, AuS) = (b_{WSN}, bld_p, sig_{bld_p}, c_{inf})$$

AuS finds $m_{wallet}$ linked with $b_{WSN}$, then the triple $(bld_p, sig_{bld_p}, c_{inf})$ is linked to $m_{wallet}$.

5. C sends AuS the $p_{coin}$ and the blinding factor:

$$C \rightarrow AuS: m^w_1(C, AuS) = (c_{WSN}, p, sig_p, c_{inf}, p)$$

AuS finds $m_{wallet}$ linked with $c_{WSN}$, verifies customer’s report by checking against bank’s report:
If any of the equations above is not satisfied, AuS will inform B who will work out either:

+ that is a B’s error (rarely) Or
+ C did not follow the protocol correctly

Otherwise, AuS stores $(p, \text{sig}_p, c_{\text{inf}})$ into $m_{\text{wallet}}$.

IV.2.2.3 Payment

1. $C \rightarrow S$: $m_1^p(C,S) = (p, \text{sig}_p, c_{\text{inf}}, \text{enc}_c\text{_WSN})$

where $\text{enc}_c\text{_WSN} = \{c\text{_WSN}, r\}_{k_s}$; $r = \text{random}$

Having $c_{\text{inf}}$, $S$ checks if the coin has not expired yet ($c_{\text{inf}}$.time-stamp) and bears a proper value($c_{\text{inf}}$ value). Then $S$ verifies $(p, \text{sig}_p)$:

$\{\text{sig}_p, p\}^\text{Ver}_{k_s} = 1$

2. $S$ challenges $C$ by sending a random $r_s$ :

$S \rightarrow C$: $m_1^p(S,C) = r_s$

3. With the secret $s$ ($s$_coin) $C$ signs $r_s$ and sends the result to $S$:

$C \rightarrow S$: $m_2^p(C,S) = \{r_s\}^\text{sign}_s$

IV-82
4. S receives $t = m_2^p(C, S)$ and verifies $\{t, r_s\}_p^{\text{ver}} = 1$. If OK, then:

+ S continues the auditing protocol (presented next) if B has alerted it to do so (see blacklisting protocol), and otherwise

+ S accepts the payment.

**IV.2.2.4 Auditing**

1. S sends the payment view to the AuS

\[ S \rightarrow \text{AuS: } m_4^A(S, \text{AuS}) = (p, \text{sig}_p, c_{\text{inf}}, \text{enc}_c_{\text{WSN}}, r_s, t) \]

2. AuS extracts $c_{\text{WSN}}$: $c_{\text{WSN}} = \{\text{enc}_c_{\text{WSN}}\}_{k_{\text{AuS}}}$. AuS maintains a blacklist with items of format $m_4^A(S, \text{AuS})$. AuS extracts $p$ and checks against the blacklist, if ‘found’ then informs S to cancel the purchase and the required law enforcement actions can take place; otherwise lets S accept the payment.

\[ \text{If } \text{Check\{blacklist, } p\} = 1 \text{ then} \]

\[ (\text{ACS} \rightarrow S: \text{CANCEL}) \]

Else

\[ \text{ACS} \rightarrow S: \text{OK} \]

End-If

3. AuS finds the $m_{\text{wallet}}$ associated with $c_{\text{WSN}}$, then uses $p$ for keying to check against the coin against $m_{\text{wallet}}$. If the result is:
+ 'not found' or 'found in the list of spent coins', then AuS informs B by giving it b_WSN. B blacklists the coin and uses b_WSN to disclose the dishonest customer’s identity.

+'found in the list of un-spent coins', then AuS marks it as spent coins with \((r_s, t)\) recorded for a proof of spending.

Note: AuS can periodically check and eliminate the expired coins from both lists of spent and un-spent coins.

**IV.2.2.5 Deposit**

1. S redeems coins to B at off-peak time by sending payment views:

   \[ S \rightarrow B: m_i^B(S, B) = (p, \text{sig}_p, c_{\text{inf}}, \text{enc}_c\_WSN, r_s, t) \]

2. B verifies the pairs \((p, \text{sig}_p)\) and \((r_s, t)\) and credits S’s account by the amount in \(c_{\text{inf}}\):

   \[
   \{\text{sig}_p, p\}_K^\text{ver} = 1
   \]

   \[
   \{t, r_s\}_p^\text{ver} = 1
   \]

**IV.2.2.6 Wallet loss recovery**

If C looses his wallet, he can recover all the remaining coins in the wallet using the following protocol.

1. C contacts B for loss recovery. B sends b_WSN to AuS
2. AuS finds the \( m\text{-}\text{wallet} \) linked to \( b\_\text{WSN} \) then sends the list of un-spent coins to B:

\[
\text{AuS} \rightarrow \text{B}: \{(p, \text{sig}_p, \text{c\_inf})\}_{i=1}^{N}
\]

Also AuS delete \( m\_\text{wallet} \) in its database.

3. B verifies the pair \((p, \text{sig}_p)\) and then credits back the corresponding amount to C’s account.

4. C goes through the \textit{Wallet Setup} phase to rebuild the wallet.

\textbf{IV.2.2.7 Blacklisting}

1. B and AuS agree on a case to blacklist a suspicious or double-spent coin.

2. AuS updates the \textit{blacklist} with that coin.

3. B alerts all S to do on-line auditing (connect to AuS to audit coins in real-time in payment)

\textbf{IV.2.2.8 Tracing coin}

B sends \( b\_\text{WSN} \) to AuS.

AuS finds the \( m\_\text{wallet} \) linked to \( b\_\text{WSN} \) \textit{and} blacklists the list of un-spent coins.

\textbf{IV.2.2.9 Tracing the owner}

1. B sends AuS the deposit view of a suspicious coin

\[
\text{B} \rightarrow \text{AuS}: (p, \text{sig}_p, \text{c\_inf}, \text{enc\_c\_WSN}, r_s, t)
\]

2. AuS extracts \( c\_\text{WSN} \): \( c\_\text{WSN} = \{\text{enc\_c\_WSN}\}_{k_{\text{AuS}}} \)
AuS checks its store of links \((m_{\text{wallet}}, c_{\text{WSN}}, b_{\text{WSN}})\), if found sends \(b_{\text{WSN}}\) to \(B\).

3. B checks its customer account database and finds the one linked to the \(b_{\text{WSN}}\).

**IV.2.3 SECURITY**

Here we state the main security features of the system and a brief justification of each feature.

*Given that an auditing server is trusted,*

*+ to always correctly follow the protocols,*

*+ not to reveal its database of mirror wallet to any other party unless required by the law,*

*+ to cooperate with the bank in the lawful tracing suspicious cases and to recover customer’s wallet,*

*then the scheme achieves,*

*+ double-spending detection: the auditing server keeps withdrawal reports in mirror wallets and marks a coin spent after the first spending;*

*+ framing-freeness: the shop can not frame the customer because it does not know \(p_{\text{coin}}\) and so can not generate \((r, t)\);*

*+ anonymity w. r. t. the bank: by using blind signature, the bank does not know \(p_{\text{coin}}\) before payment;*
+ anonymity w. r. t. the auditing server as the auditing server knows only $b_{WSN}$ and $c_{WSN}$, pseudonyms of the customer;

+ unlinkability w. r. t. the bank: the bank does not know $c_{WSN}$, which is the only key to link customer payment views together;

+ anonymity revocability: directly seen from the protocols; and

+ wallet loss recoverability: directly seen from the protocols.

**IV.2.4 Efficiency**

+ Most previous e-cash proposals suffer from the problem that the bank has to maintain a large database of deposited coins to be able to check double spending and search through this database which is a time consuming task. In our scheme the time to check a coin against such database is nearly zero because the database is distributed among customer mirror wallets. This efficiency is equivalent to those in [CPS94, LoMa94].

+ The involvement of the auditing server is non-interactive and it is off-line during withdrawal and, off-line during payment if no suspicious case is reported. Overall, the on-line processing in the system is minimal.
V. EXTENDED SCHEME WITH DISTRIBUTED ‘MIRROR WALLETS’ AND ANONYMOUS AUDITING SERVERS

This chapter is the continuation of chapter 4. Here we consider the weaknesses of the scheme presented in the previous chapter and develop an extended scheme to overcome these weaknesses.

V.1 THE EXTENDED MODEL

V.1.1 DISTRIBUTED MODEL WITH ANONYMOUS COMMUNICATION

In chapter 4 we showed that using mirror wallets is very helpful. However it introduces some new problems. Firstly, it causes the linkability of the customer’s transaction w. r. t. the auditing server. That is all the customer’s transactions can be linked together through the mirror wallet. Therefore, if just one transaction is disclosed with customer’s name known to the auditing server then all other transactions can be linked to that customer’s name. An example situation is when a
shop discloses the customer’s name to a person who works at the auditing server.
Secondly, the auditing server becomes a very attractive target for banks, or other attackers: an attacker who knows $b_{WSN}$ or $c_{WSN}$ of his enemy’s wallet can break into the auditing server and find the proper mirror wallet to trace his enemy’s business. Finally, we note a potential threat to the system when a staff is bribed to sell auditing information to a bank or an attacker.

To prevent these potential weaknesses, we develop an extended model with multiple auditing servers. At the Wallet Setup, an auditing server will be selected and authorized to create and maintain the customer’s mirror wallet. This selection is made fair to both the customer and the bank, i.e., the selection is random to both sides, by using the protocol in [Co-tossing]. Therefore, neither the customer nor the bank can choose an auditing server at their advantage. It helps to protect the auditing server against the customer’s attempt to double-spend without being detected and against the bank’s to record all the customer’s transactions.

We establish anonymous communication channels between the auditing server and the other parties. This is achieved using intermediate servers called ACSs (Anonymous Communication Server) and using aliases. The ACS officer who establishes and maintains the ACSs, generates an encrypted list of network addresses of all the auditing servers by using a secret key encryption scheme. This secret key
encryption scheme is known to all the ACS. Bank, shops, customers communicates with the auditing servers through the encrypted addresses in this list, used as aliases. A message sent to an auditing server must pass through one of the ACS, who will decrypt the alias address and forward the message to the destination auditing server. Aliases (encrypted addresses) can be periodically changed by encrypting real addresses with salt values that are changed. This will protect against gradual leaking of addresses. If the number of auditing servers is large, it is practically impossible to guess the true address of an auditing server from its alias.

([LoMa94] also proposed an electronic payment scheme that uses an intermediate machine to provide anonymous channels.)

An important advantage of this distributed model is the distribution of trust and hence lowering trust level on a single server. This makes the system more practical compared to the basic model.

In the new model, the database at an auditing server is only a small part of that used in the basic model. This has two advantages. Firstly, the risk to selling auditing information by a staff member is reduced. In the new model, he can not find out who likes to buy this auditing information. Secondly, if an auditing server is compromised, only a small part of the customer population is put at risk of loosing their privacy.
A distributed model does not only help to improve system robustness but also scalability. In the basic model, when the customer population grows, the auditing server becomes a bottleneck flooded with huge flow of transactions. In distributed model, one can set an upper bound on the number of customers per auditing server and introduce new auditing server when the number of customers passes that bound.

**V.1.2 A CONTINUUM BETWEEN ON-LINE AND OFF-LINE PAYMENT**

Most existing anonymous e-cash systems are on-line systems while the more advanced model – off-line e-cash – is also extensively studied (chapter II). Compared to off-line payment that relies on detecting the double-spender after the fact, on-line payment provides a prior restraint against double-spending. In the former the real-time checking process effectively stops cheating attempts, while in the latter the delay window in checking process could let the cheater to make many purchases with the same e-coin. The cheater may have stolen the e-coin from another customer, in which case the bank will catch the honest customer! Using this method a collaborative group of cheaters may succeed in hundreds of purchases in a single day. We call this attack a MMS (mega-multiple-spending) attack. One may argue that off-line e-cash systems should not be used for high value purchases. Such purchases must be reserved for on-line systems. However if a MMS attack is launched, the loss would not be easy to tolerate.
‘Wallet with embedded observer’ is an innovative off-line paradigm in that one manages to achieve a mechanism for prior restraint against double-spending with the embedded observer device acting as an electronic guardian ([ChPe92, Bra93b, CAFE94, Fe94]). Nevertheless, this mechanism is not perfectly immune to double-spending attacks because breaking such tamper-resistant device is just a matter of cost and time. Especially, given that the compromised wallet is full of coins, an MMS attack can really damaging.

Bypassing tamper-resistance combined with MMS produces a very strong attack that although hard to implement but not impossible. Here we give a solution to this problem in the extended model. In chapter 3, we proposed the off-on-line approach that effectively combines the security of on-line and efficiency of off-line systems. The basic idea is to provide different types of processing for different coins based on their value. In the rest of this section we will show how to incorporate this feature into the model described in the earlier sections.

In our proposed solution, we provide smooth change between the processing for different grades of coin value. In other words, we achieve a continuum between on-line and off-line payment.
Firstly, we distinguish three stages of checking process of coins:

A. Verifying the bank signature on the coin ($p\_coin$) and verifying the knowledge of the customer about $s\_coin$.

B. Checking the coin against the blacklist of double-spent coins and suspicious coins.

C. Checking the coin against the mirror wallet.

Secondly, we let the ACS take part in the process of checking. Recall that in the basic model, stage A is performed in payment phase, processed by the shop, while stages B and C are in the auditing phase, processed by the auditing server. Here, stage B is separated from auditing and processed by the ACS. The blacklist is stored at ACSs instead of the auditing servers. Therefore, the three stages A, B & C are processed at separate places (shop, ACS and auditing server respectively).

Thirdly, we establish a set of rules to provide a continuum of checking between on-line and off-line payment, based on the principle that “the higher value coin must have less delay in going through stages B and C (if it has passed the earlier stage)”:

+ For small value coins (say < $1) only stage A in real-time (B,C is off-line)
+ Middle value coins (say ≥ $1) have stages A and B in real-time (C is off-line), that is the shop contacting an ACS in real-time at payment to check coins against the blacklist.

+ High value coins (say ≥ $100) have all the checking stages (A, B, C) in real-time. That is after the ACS checks a coin against the blacklist, it immediately contacts the auditing server for checking against the corresponding mirror wallet. All these 3 stages are made on-line before letting the shop accept the payment.

+ The delays between stages A and B (if B is made off-line) and between stages B and C (if C is made off-line) are inversely proportional to the coin value. For example, coins of values between $1 and $99 are all processed on-line in stages A and B, and off-line in stage C. However, the delay between payment and auditing (phase C) are varied, say 5 minutes for $20 coins but 2 minutes for $50 coins.

This mechanism achieves an ideal trade-off between the efficiency and security:

+ The low value purchases are more frequent than high value purchases and in the proposed system require a higher degree of checking (towards the off-line extreme on the continuum introduced above). This results in a very economical system.

+ The MMS attack is restrained and the loss, if attack occurs, is controllable: the more valuable coins are checked quicker in stages B and C and therefore if the attacker wants to deploy MMS with high value coins he has very little time to succeed, and he is forced to use low value coins if he wants longer period.
Finally, there is a specific local ACS for each small location (small town, district, suburb) where every shop in the area maintains a live communication with it. Each ACS has live channels to all the auditing servers. There is an upper bound on the number of live connections with shops per ACS. Whenever the shopping in an area grows, new ACS are introduced to avoid a bottleneck in the system. This results in that the ACSs are always available for communications of various types discussed in (3).

Generally, the ACS has two roles:
+ to be the intermediate machine to anonymize the auditing servers.
+ to maintain the blacklist and check against this blacklist with coins submitted from the shop. Often, this checking is on-line as for coins of not very low value, but this is affordable due to the limited scope of the trading in a small local area around the ACS. Besides, the work of auditing server is to discover illegal coins and, if found, alarm all the ACSs to update the blacklist. In short, the work of ACSs and auditing servers combined replaces the work of the auditing server in the basic model and the alarm channels between banks and shops are no longer needed.
Because of this dual role, ACS can stand in two ways: for Anonymous Communication Server and for Anti Criminal Server (an incredibly nice coincidence!).

**V.2 THE EXTENDED SCHEME**

**V.2.1 NEW SCENARIO WITH EXTENDED FEATURES**

The system consists of Customers (C), Shops (S), Banks (B), ACSs and auditing servers (AuS). There are two trusted parties:

+ the judge who enforces the law and sends the instructions or court orders to revoke anonymity of illegal transactions and, who settle cases of complaints and disputes between players.

+ the ACS officer who sets up and controls the ACS servers.

The ACS officer establishes local ACS servers in each small area. Auditing servers are trusted softwares that are installed by the ACS officer at locations approved by ACS officer. The number of auditing servers should be large to minimize the risk and loss incurred if an auditing database is compromised. Auditing servers are hidden from banks, shops and customers by the method mentioned earlier with the help of ACS.

Extending the protocols in the basic scheme to the distributed case, we have:
In wallet setup, an anonymous auditing server is chosen to audit the customer's transaction transactions. The customer can authorize the ACS officer to make the selection for him otherwise (for VIP customers who are more sensible with privacy) a fair selection can be made by collaboration of the customer and the bank (based on [Co-tossing]).

In withdrawal and payment phase, the bank, the customer and the shop communicate with the corresponding auditing server through local ACS using an alias (as the AuS address encrypted by the ACS key).

The payment and auditing phases follow the rules of the continuum of on-line & off-line introduced in section V.1.2, with checking against blacklist at the local ACS.

In coin tracing and owner tracing protocols, the bank cooperates with the auditing server as in the basic scheme, except that the bank has to contact the ACS officer beforehand to show the court order.

Blacklisting is done by the auditing server by alarming all the ACS to update the blacklist with double-spent coins, or coins, that are traced by the court orders (in cases of blackmailing or bank robbery)

In the following we will present the new features of the extended scheme.
### V.2.2 Wallet Setup

<table>
<thead>
<tr>
<th>Customer (C)</th>
<th>Bank (B)</th>
<th>ACS officer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Randomly choosing $AA_c = {addr_{AS_i}}_{i=1}^{MaxNoAuS}$ in one of the two ways:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The ACS officer randomly chooses an item $AA_c$ in the list ${addr_{AuS_i}}_{i=1}^{MaxNoAuS}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The customer and the bank collaboratively and randomly choose $AA_c$ in the above list.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>B</th>
<th>Auditing server (AuS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboratively generate a random number $r_1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>([Tossing coin])</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{WSN} = f(&lt;user_ID&gt;, r_1)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$c_{WSN} = f(b_{WSN}, r_2)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Collaboratively generate a random number $r_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>([Tossing coin])</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$c_{WSN} = f(b_{WSN}, r_2)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### V.2.3 WITHDRAWAL

<table>
<thead>
<tr>
<th>Customer (C)</th>
<th>Bank (B)</th>
<th>Auditing server (AuS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_{\text{inf}} = AA_c[v\text{al}ue</td>
<td>t\text{ime_stamp}$</td>
<td>$p = {s}<em>{k</em>{p,q}}$</td>
</tr>
<tr>
<td>$m^w_1(C,B) = (b_{ld_p}, c_{\text{inf}}, \text{Customer_ID, amount})$</td>
<td>$\text{Verify}(p, c_{\text{inf}}, \text{sig}_bld_p) = 1$</td>
<td>$\text{coin} = (s, p</td>
</tr>
<tr>
<td>$\Delta t$ (reporting to AuS can be done off-line)</td>
<td>$m^w_1(B, AuS) = (b_{WSN}, b_{ld_p}, \text{sig}<em>bld_p, c</em>{\text{inf}})$</td>
<td>$m^w_1(B, AuS)$</td>
</tr>
<tr>
<td>$m^w_1(C, AuS) = (c_{WSN}, p, \text{sig}<em>p, c</em>{\text{inf}}, p)$</td>
<td>$m^w_1(C, AuS)$</td>
<td>$\text{receipt} = {m^w_1(C, AuS)}<em>{\text{sign}}^k</em>{\text{ACS}}$</td>
</tr>
</tbody>
</table>

$\text{Storage (p, sig}_p, c_{\text{inf}})$
and $(b_{WSN}, c_{WSN})$
V.2.5 AUDITING

<table>
<thead>
<tr>
<th>ACS</th>
<th>Auditing Server (AuS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;address of AuS&gt; = {AAC}<em>{K</em>{ACS}}</td>
<td></td>
</tr>
<tr>
<td>c_WS N = {enc_c_WS N}<em>{K</em>{ACS}}</td>
<td></td>
</tr>
<tr>
<td>m^A(ACS,AuS) = (\text{p,}\text{sig_p,c_inf,c_WS N,r_s},t)</td>
<td></td>
</tr>
</tbody>
</table>

\[ m^A(ACS,AuS) \rightarrow m\text{-wallet} = \text{Find(database, c_WS N)} \]
\[ \text{Verify(m_wallet, p, UNSPENT)} = 1 \]
\[ \text{to-blacklist} \]

V.2.6 ROBUSTNESS AGAINST THE MMS ATTACK

**Theorem 1.** If all other parties follow the protocols correctly then the cheating customer can not multiple-spend more than \( L_{d\text{-spend}} \) times with a particular coin or she will be caught. \( L_{d\text{-spend}} \) is a controllable threshold.

**Proof.**

Let \( x \) denote the coin value of a coin. \( t_{ACS} = t_{ACS}(x) \) denotes the delay between the time an ACS receives a coin of value \( x \), and the time it sends this coin to the proper AuS for auditing. \( t_{AuS} = t_{AuS}(x) \) denotes the time delay between an AuS receives a coin and the time it really audits this coin. \( T_{black} \) denotes the maximum time that it may take to blacklist a coin when an AuS discovers a black coin. Let \( t_{d\text{-spend}} \) denote
the minimum time between two consecutive double-spending tries with a particular coin.

After the cheater spends this coin the first time, $t = t_{ACS} + t_{AuS}$, the coin will be marked spent in the corresponding mirror wallet. After the cheater spends the coin for the second time, at $t = t_{ACS} + t_{AuS}$ units of time later the corresponding AuS will discover that the coin has been already spent and blacklist the coin. Then at $t = t_{ACS} + t_{AuS} + T_{black}$ units of time after the second attempt, the coin is blacklisted and from that time on any further attempt will cause the cheater to be caught. We have $L_{d-spend} = (t_{ACS} + t_{AuS} + T_{black})/t_{d-spend} + 1$, and adding one is to include the first attempt that is truly accepted).

There must exist a constant $c_{d-spend}$ that is the lower bound for $t_{d-spend}$, because it always takes some time to travel between shops, selecting the goods, queuing and processing transaction. Because of all of that $L_{d-spend}$ is controllable: it can be made as small as required by designing proper $t_{ACS}$, $t_{AuS}$, which depend on the resource supplied for the ACS service (the number of ACS and AuS, speed of each machine, bandwidth of the network communication, etc.).
Of course $L_{d-spend}$ depends on $x$ as well as $t_{AuS}$ and $t_{AuS}$. Assume $T_{black} \ll t_{d-spend}$ when $x$ is large enough, then $t_{ACS} = t_{AuS} = 0$ (online auditing) and no double-spending can be made successfully.

V.2.7 CUSTOMER COMPLAINTS

Theorem 2. If all the authorized parties and the shop correctly follow the protocols, then customer complaints for being wrongly blacklisted are always solvable.

Proof.

There are only two situations that the customer coins can be blacklisted (apart from the cases of criminals in which the bank and the ACS officer have a court order to do so):

1. The customer double-spent a coin and the auditing server blacklisted it: in this case we note that the auditing server only blacklisted the coin when it had received the two messages $m^A_{1}(ACS,AuS) = (p, sig_p, c_{\inf}, c_{\text{WSN}}, r_5, t)$ with the same $p$ and different pairs of $(r_5, t)$. Given this information, the judge can always tell that the customer is lying because only the customer could produce such pair $(r_5, t)$. If the ACS officer blacklisted the coin it must have kept this proof, otherwise the customer wins the case and the ACS officer has to pay for that.
2. The customer had not followed the step 5 in the withdrawal protocol, namely reporting the withdrawal view to the auditing server, and then the auditing server blacklisted the coin (after the customer payment) as it did not find the coin in the corresponding mirror wallet. If the customer can show that she had really followed the step 5 in the withdrawal protocol by showing the receipt as an ACS signature on $m_i^w(ACS, C)$ then the judge regards the customer as innocent, otherwise the case is clearly that the customer tells lie.

**V.3 Security & Efficiency**

The extended system achieves all the security properties of the basic system in section V.2.3. Moreover, it effectively improves the system robustness:

+ The distributed nature and the mechanism for anonymous communication helps to control the risk of a trusted party being attacked from inside. In the basic system, it would be a disaster if an expert staff in the auditing server is corrupted, while the loss in the distributed system is small.

+ The MMS attack is no longer a threat. The loss from this attack can be controlled.

*The efficiency is improved:*

+ The system achieves a trade-off between security and economy as discussed in the section V.1.2.

+ Computation time is effectively reduced due to the distributed nature of the auditing servers.
The alarm channels between banks and shops (costly for shops) are no longer used.

**V.4 CONCLUDING REMARKS**

A new architecture for electronic-cash systems is introduced that provides a new kind of trade-off between customer privacy and legal traceability. Customer transactions are stored in a network object, the so called *mirror wallet*, which is hidden from all other parties unless an illegal activity is committed by the customer. The notion of *mirror wallet* also helps to provide the property of wallet loss recoverability.

Another trade-off is between preventing double-spending (MMS attack) and system efficiency by combining on-line and off-line payment ([NgRa97]).

The architecture is versatile. Functionality can be extended to include coin-divisibility, overspending detection, electronic check, multi-currency support, etc., when applying the method mentioned in 2.1.7.

The high performance of the system is due to the combined techniques from the field of cryptography and distributed system (e.g. the organization of the network of ACSs and auditing servers). This can be seen as the unique feature of our work compared to the other previous works in e-cash.
This combination allows us to obtain the desirable e-cash properties but avoid the highly difficult and complex mathematical systems, such as the well-known work of Brands' based on the presentation problem (see [Bra93]).

Providing legal traceability through auditing server is similar to the notion of ombudsman in [JaYu96, JaYu97b] or the blinding office in [M'Ra96]. However our proposal does not suffer the interactive and on-line involvement of the trusted third-party in withdrawal phase. Moreover, we have also considered and succeeded to control, the potential risk of trust abuse: collusion attack when an insider in the trusted third-party (auditing server, ombudsman) is corrupted. Another advantage is that the checking phase at the auditing server is much faster than that of ombudsman in [JaYu96] by using mirror wallets.
VI. CONCLUSION

This thesis deals with the topic of analyzing and designing electronic cash systems. We study this on the background of applied cryptography and distributed systems. In e-cash systems, to achieve properties such as anonymity and off-line payment, strong cryptographic protocols are used. However, deployment of the participants in e-cash systems, their functionality and the flows of data between them, are aspects of distributed systems. Combining the two viewpoints helps us to come up with some new results.

Our main focus has been in decentralizing the process of checking coins, which is usually handled by the bank host only. Our solution is to distribute the check process and the data storage among a number of intermediate servers added into the system.

In the e-cash framework provided in chapter 3, a hierarchical network of Anti-Cheating Servers (ACS) is introduced, which allows distributing the checking workload. The denomination of e-coins together with the differential on-line e-coins
checking helps us to optimize the system efficiency, achieving a continuum between on-line and off-line payment.

The idea of *distributing* is exploited much further in chapters 4 & 5 with the introduction of the notions of mirror wallets and auditing servers. The checked space reduces from the whole database of coins ever produced to a small part of it which is anonymously linked to a specific user. Obviously, this strongly reduces the computation workload of the system. The extended scheme in chapter 5 also provides many useful functions that are tracing coins or owner and wallet loss recovery.

Studying the active field of e-cash systems is hard but joyful. The *distributed systems* idea has directed us to new e-cash schemes based on new extended models, using a number of different existing cryptographic tools. We hope to be able to further extend these ideas in future.
REFERENCES

[AbFu96] M. Abe and E. Fujisaki. How to Date Blind Signature. Advances in Cryptology - Asiacrypt’96, Lecture Notes in Computer Science, Springer-Verlag, pp. 244-251


California, San Diego 1997. URL http://www-cse.ucsd.edu/users/markus


