Applying Anti-Trust Policies to Increase Trust in a Versatile E-Money System

Markus Jakobsson* Moti Yung†

Abstract

Due to business relationships, alliances, trust, and distribution of liability, *distribution of power* is an important issue in financial systems. At the same time as the security of the scheme is strengthened by this decentralization, the *perception* of the security is also strengthened, which is important from a business point of view. Furthermore, apart from increasing the security, client trust and availability of the system, distribution of power can also increase its *functionality*, as we demonstrate.

We suggest an anti-trust mechanism, namely, a method for distribution of the centralized parties into many modules (potentially controlled by different entities), and apply it to a versatile electronic-money system. The method diffuses a task into distributed modules using recent cryptographic technology; doing so, it achieves increased security, privacy, availability and functionality without introducing any noticeable disadvantage. It uses *Magic Ink Signatures* [29], which are blind signatures that are distributedly generated using a threshold of signers, and where signatures can always be unblinded using (perhaps another) threshold of signers as well. Furthermore, we combine this with recent proactive technology, which enables a stronger adversarial setting. We also suggest techniques for reorganization of data stored and used by various functions, employing secure repository.

The result is an electronic money system that allows user anonymity and its revocation (a notion recently advocated by some works so as to prevent potential criminal actions.) The control over revoking anonymity is given to distributed modules that control a hidden alarm channel. As part of the task diffusion we find ways to simplify and reduce the overall complexity of the system. The revocation ability and distribution of the trust are efficient and allow a large degree of versatility in the functionality of the system (change mechanisms, numerous financial instruments: cash, charge, check, micro-payments, etc.).

*Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92093. Research Supported by NSF YI Award CCR-92-570979 and Sloan Research Fellowship BR-3311. Part of the work was done while visiting IBM T.J. Watson. Email: markus@cs.ucsd.edu

†CertCo. Email: moti@cs.columbia.edu, moti@certco.com

1 Introduction

It is provisioned that in the near future, electronic money systems will be part of the economy, exploiting advances and accessibility of networking technologies to homes, offices, and other locations. Electronic money systems represent a model of highly sensitive systems where trust and protection are much needed, and for which it is vital to win consumer confidence and support. We argue that this has to be gained (a) by not forcing users to put a lot of trust in individual entities for the security of their funds and integrity, and (b) by supplying the user with a practical product. We do so by distributing the power and by introducing mechanisms for removing the privacy in the case of abuse. This revocation capability, both, strengthens the security against fraudulent behavior (which everybody otherwise pays for) and allows for a versatile and efficient scheme.

Originally, perfect anonymity of users of e-money systems was advocated very strongly, granting privacy to users with respect to their purchase patterns. This appealing characteristic of paper money, however, has a dark side: it has recently been pointed out that perfect anonymity also enables "perfect" crimes like money laundry, blackmailing [46], and bank robberies [28]. The reason why we label these crimes *perfect* is that in a setting with perfect anonymity, the skilled perpetrator would be provably untraceable. Thus, we argue that a solution that obtains perfect privacy in the *cryptographic sense*, is far from perfect in the *social or economical sense*, and that we may benefit from a solution in which the anonymity is revocable. On the other hand, giving the revocation power to a single entity enables abuse against individuals, thereby bringing us back to square one.

Lately, several groups have noticed the above problem and suggested schemes ([4, 5, 6, 28, 47, 21]) with computational anonymity that can be removed by cooperation between the Bank and a trustee. The disadvantage of several of these systems are their centralized and highly sensitive functions. We believe that for such a system to be usable and economically viable, it is crucial to increase the overall trust in it. We pursue this issue here; we distribute the sensitive functions into a distributed system where components may be held by different entities (this was typically done for a one-party function in a system, but here we do it for a function that is already a two-party construction). We employ new and recent tools to also enable resistance to strong attacks that are expected over open networks (the Internet.) At the same time, we achieve simplification and allow for a reduction of the storage requirements. We make sure that sensitive actions of producing coins and revoking anonymity are done only under quorum agreement among the distributed entities, giving additional control via the distribution of authority amongst various bodies.

We will base our solution on the system architecture suggested in [28], which protects against the most general attack, namely when a bank is forced to give away its cryptographic keys. We would like to note that while we follow closely the model and some of the functions in [28] (adding distributed functions), the work here requires certain (technical and procedural) innovations that are beyond the simple distribution of the functions in the earlier work.

The system provides protection against direct attacks on the bank and its modules and provides more trusted exposure of criminals by distributing the bank and the trustee functions. This is done in order to protect against an attacker compromising their keys, to ascertain availability, and to protect against corruption. Furthermore, we utilize proactive function sharing methods to achieve security against a strong "perpetually infiltrating" attacker.

At the same time, the system meets the level of versatility in payment methods that is advocated in [28], where various methods and mechanisms of payment are integrated under one basic mechanism. In fact, mechanisms such as divisibility of coins, checks, surety bonds, fairness [26], micro-payments [42], and more, can be implemented virtually for free. Thus, our scheme is most efficient and versatile, and protects against the strongest known attacks, using both cryptography and distributed storage methods. It gives computational anonymity, which can be revoked (only) by cooperation between the various modules (trustees) of the issuing bank and trustees comprising the entity of the Ombudsman[1].

At the heart of our method are a number of techniques and notions: first is the distribution of the Bank and the Ombudsman, and the use of proactive sharing of the Bank/Ombudsman signature function. Then, we allow for a reorganization of memory, permitting one entity (the Bank) to keep storage for another entity (the Ombudsman) in a secure repository. This reorganization and diffusion of function is a strengthening mechanism that protects against insider attacks (which is the most prevalent attack on financial institutes and systems,) and also makes the distribution of the (virtually) storage-free entity – the Ombudsman – inexpensive to perform. We further use cryptographic tools that are applicable under the suggested distributed control. In order to limit the trust assumption to a minimum, we need a *quorum* blind signature generation scheme, and a *quorum* unblinding scheme, where it is always possible for an arbitrary Bank/Ombudsman quorum to perform their tasks. For this, we can use the magic ink DSS signature generation scheme of [29].

1.1 Related Work

Let us review our results in light of related work. Since the introduction of electronic money, in particular the off-line system by Chaum, Fiat and Naor [9], a large number of schemes (e.g., [3, 10, 24, 22, 23, 35, 36, 37]) have been suggested, introducing important mechanisms and properties to the first scheme. Most of these schemes stress perfect anonymity as the ultimate goal in protecting the user's right to privacy. However, as von Solms and Naccache [46] pointed out, such schemes are susceptible to two unpleasant forms of abuse, *money*

[1] Term of Scandinavian origin for consumer representative.

laundering and *blackmailing*. A third, and stronger, attack, *bank robbery*, was introduced in [28]; in a bank robbery, an attacker may get the Bank's key or (in case such attack is too aggressive, and the Bank may refuse to cooperate in it) it may force the Bank to give out coins in a modified withdrawal protocol so that the Bank cannot trace the coin (employing a general secure computation protocol that translate the original withdrawal protocol to one where the result is not visible or computable by the Bank.) Recently, efforts have been made to avoid these types of attacks. The schemes by Brickell, Gemmell and Kravitz [4] and Camenisch, Maurer, Piveteau and Stadler [5, 6, 47] cope with the first two attacks by introducing tracing possibilities into the architecture, and Jakobsson and Yung [28] also prevent the third attack, by introducing signatures that cannot be blinded by the attacker.

Our starting point is the latter payment scheme. We achieve the very same goals w.r.t. attacks and anonymity, but with less assumptions, more robustness and security (as less trust in entities behaving correctly is needed) and for a similar cost (taking into account that we have more entities taking part in the computation.) Just as in the other solutions, the tracing is performed by the cooperation between the issuing Bank and a trustee, the Ombudsman. Given a magic ink signature scheme where the tags are encryptions of the withdrawn coins, we can do away with the Ombudsman database (without increasing the size of the Bank database) by organizing a "repository database" at the Bank. This gives a solution in which we only need to trust the Ombudsman modules not to lose their secret keys. Furthermore, it enables us to distribute the Ombudsman in a very inexpensive way, in order to secure its availability, which is of vital importance to the security of the system. We employ threshold cryptographic schemes [12, 29] (see also [11]), and proactive technology ([19, 20, 38]) to cope with perpetual attacks by a strong mobile adversary. Furthermore, we introduce a simpler signature scheme (i.e., DSS signatures) for the Bank/Ombudsman authentication of coins, decreasing the computational requirements of the participants.

Thus, we obtain a payment architecture where attacks like money laundry, blackmail, and bank robbery can be dealt with, by revocation of the (computational) anonymity of criminal users. The distributed nature of our solution helps in various ways. Foremost, it helps to cope with direct attacks on its secure key memory, which is crucial for tracing of funds (including overspent coins.) Also, it improves the availability of the Ombudsman, and the Bank's trust in the same.

We note that similar, but stronger, compromises in provision of anonymity have been made by Low, Maxemchuk and Paul [30], in order to obtain an extension to the standard credit card system, in which an intermediate degree of anonymity is combined with the ability to trace purchases of credit card type. Other extensions to the existing credit card payment system are [18, 31, 32, 45]; here, however, anonymity is not a primary issue, but backwards compatibility and simplicity are stressed more.

2 System Model

The model follows the one suggested in [28]:

2.1 Participants

The system can be modeled by seven types of (polynomial-time limited) participants/entities : Users (1) who withdraw money and perform payments. Users enjoy computational anonymity. Shops (2) get money from users and deposit it in the bank, while banks (3) manage user accounts, issue and redeem money. The banks are able to alert the shops to engage in a non-standard (on-line) payment procedure. Based on court orders banks may engage in blacklisting and tracing (as a crime prevention mechanism). The ombudsman (4) participates in withdrawals and assists in reacting to court orders. The judge (5) employs enforcement mechanisms, and issues court orders. Finally, there is a certification authority (6) and a key directory (7) for public keys. In more detail, the participants are as follows:

1. Users

   Users withdraw funds from the bank, and make payments to shops. They have certified public keys associated with themselves, allowing the users to identify themselves, and sign agreements. It is the goal of the honest users to transfer funds in return for merchandise or information. Users are concerned with being defrauded of their money, being refused service, having their spending habits monitored, and being falsely accused of a crime. It is the goal of the honest users to transfer funds in return for merchandise or information in a way that does not allow cheating protocol participants to defraud them. We will demand that the honest users enjoy (computational) anonymity.

2. Shops

   The shops receive payments from users. It is important for the shops to be able to verify that they have received information that corresponds to funds, so that when the coins representing funds are deposited, the shops' accounts will be credited with the amount the coins represent. The shops may be interested in tracing purchases, i.e., to match a spent coin to a user identity.

3. Banks

   The bank (who is distributed w.r.t. the ability to sign and trace) manages user accounts, issues money to users and receives spent coins from the shops. The bank may be interested in tracing purchases, and may cooperate with other participants in order to try to do so. Multiple, independent banks will issue and accept coins of distinctive types. The banks are able to issue *alerts*, forcing shops receiving coins issued by the alerting bank for specified time periods to engage in a protocol preventing an attacker to successfully spend money obtained by robbing the bank. Furthermore, each bank after a court order can blacklist coins or have payments traced.

4. Ombudsman

   The ombudsman (who is distributed w.r.t. the ability to sign and trace) is the representative of the honest users. The ombudsman will assist the bank in tracing coins of a suspected attacker upon a court order, but will not assist anybody in any other tracing. We require the ombudsman to be available for the tracing of attackers by the bank, available to the bank during each withdrawal session, and available to shops for verifying the correctness of coins, should the bank request this. Besides availability and non-cooperation with the bank regarding traceability, the ombudsman will not have to be trusted in any other way, neither by the bank nor the users. Specifically, the ombudsman will not be able to frame users or successfully cheat any participant. Efficiently, there may be one ombudsman per bank, but we will treat the ombudsman as one single (although distributed) entity in our discussions. The ombudsman and the bank have a private communication channel that allows them to communicate the status of transactions, e.g., to issue an alert if a bank robbery is launched.

5. Judge

   The judge has the task of resolving conflicts between the above participants after analyzing the corresponding transcripts. The judge will issue court orders forcing the ombudsman to cooperate with the bank to trace coins or user funds specified by the judge. The judge will not have to be trusted in any way by any participant, apart from fulfilling these functions. The judge may be modeled by a multiplicity of independent judges with the same goals and behavior, but we will treat it as a single entity in our discussions.

6. Certification authority

   The certification authority certifies public keys of participants, including keys (of time-limited validity) used by the bank and the ombudsman to produce authentications on coins. The certification may have a hierarchical structure, but, again, we will think of it as a single participant.

7. Key directory

   The key directory stores all currently certified public keys. It is the purpose of the key directory to inform all participants what the currently used public keys are.

The Bank, the Ombudsman, the judge and the certification authority will each have a public key associated with themselves. These public keys (but for that of the certification authority) will be certified by the certification authority. Individual shops will have a (not necessarily certified) public key associated with them.

Remark: It is possible to envision a scenario in which no outside consumer representative is needed, but where we allow a conglomerate of banks to control the ombudsman entity.

2.2 Attackers

An attacker is any coalition of protocol participants who deviate from the specified protocols. Attackers may corrupt any set of the Bank and Ombudsman servers and may depart from the prescribed protocols in any way, including forcing other participants to engage in protocols different from those that are prescribed, in order to obtain spendable money or services for a greater amount than their accounts are billed. An attacker may force any entity to give out its secret keys. It is the goal of the attackers to obtain spendable money or services for a greater amount than their accounts are billed. We distinguish between two different attackers: (1) the *weak* attacker may only corrupt a non-quorum of Bank and Ombudsman servers in each time period (see [20, 19]), as well as any number of users and shops, and (2) the *strong* attacker, who can corrupt *any number* of Bank and Ombudsman servers.

2.3 Time

Time is divided into (possibly overlapping) time periods[2] of publicly known starts and lengths. A withdrawn coin is only accepted by the Bank for credit within its corresponding time frame, specified in the coin. Thus, each coin will have an explicit expiration date, after which it will not be accepted for deposit (and therefore not as payment either). The expiration date of the coin will be part of the coin, and set by the Bank in a non-blindable fashion. It will be readable by all participants, but alterable by nobody.

2.4 Trust model

The following basic assumptions underlie the architecture:

1. The users trust that a quorum of the Bank and the Ombudsman servers will not conspire against them. (A quorum is a set of servers containing at least one bank server and at least one ombudsman server.)

[2] These will be of a length sufficient to strike a good balance between storage requirements and degree of anonymity, two contrasting goals. The time intervals are for simplicity multiples of the refresh intervals in the proactive secret sharing.

2. The Bank trusts a threshold of the Ombudsman servers to be available during withdrawals[3], for traces, and immediately after a bank robbery has taken place.

3. All the participants trust the judge to be honest and fair.

4. All the participants trust the certification authority only to perform certifications of accurate documents and correctly associated pairs of public keys and names of owners.

5. The users trust the Bank not to incriminate them, or to steal their money by annulling their accounts, etc.[4]

EVENTS:
The following events are part of the system description: *open an account*, *withdraw money*, *spend money*[5], *deposit money*, *refresh money*[6], *register new Bank/Ombudsman public keys*, *detect over-spending*, *trace*, *blacklist coins*, *freeze and thaw funds*[7], *alert on/off*[8], *arbitration*, and *certification of public keys*. We will concentrate on those that will have to be changed in order to distribute the Bank and the Ombudsman.

We prove that our solution satisfies the following requirements based on the definitions (to be specified) in a sequence of theorems in the Appendix:

Theorem (main):
The suggested system enables correct transfer of funds. Against the appropriate coalition of attackers it possesses the following security properties: *unforgeability* of coins, *(computational) user anonymity w.r.t. any non-quorum of bank and ombudsman servers*, *traceability and revocability of all coins*[9], *framing-freeness*[10], and *overspending robustness*[11].

[3] Employing a slightly stronger trust model, we do not have to require the Ombudsman to be available during withdrawals.

[4] This assumption can easily be avoided to the price of an inefficient system in which each transaction has to be signed by the Bank as well as the users, and signatures exchanged using methods for simultaneous exchange of secrets.

[5] After a successful bank robbery, shops will have to deposit coins before accepting them as valid.

[6] Replacing coins that are soon to expire by spending the coins and withdrawing the corresponding amount anew.

[7] This corresponds to a temporary blacklisting of coins.

[8] This is used after a successful bank robbery to inform the shops that the on-line payment/deposit protocol must be used.

[9] If and only if the Bank and the Ombudsman cooperate.

[10] Neither the Bank (only applicable if the user signs each withdrawal) nor the Ombudsman can falsely incriminate a user.

[11] If an overspending is performed then no coalition of shops, users, the Bank and the Ombudsman with access to all spending transcripts can generate a new transcript for the coin with public key y, that will be accepted by the Bank when deposited.

3 Definitions

We give informal definitions of concepts from the literature, and refer the reader to previous work for more information.

Definition 1: Cryptosystem
A cryptosystem has the following components:

- A *security parameter* k.
- A *message space*, $\mathcal{M}_k = \{0,1\}^k$, to which the encryption algorithm may be applied.
- A p-time *key generation algorithm* $\mathcal{KG}$ producing a random pair (PK, SK) of matching public and secret keys on input 1^k.
- A p-time *encryption algorithm* $\mathcal{E}$. This is a probabilistic polynomial time algorithm which given a message m and a public key PK outputs an encryption $\overline{m}$ of m with respect to PK.
- A p-time *decryption algorithm* $\mathcal{D}$. This is a polynomial time algorithm which given E, $\overline{m}$, and a matching pair of public and secret keys, (PK, SK) outputs m.

Security:
A cryptosystem is *secure* if there does not exist a poly-time decrypting mechanism $\mathcal{D}$ who for infinitely many k succeeds to decrypt a random encrypted message $\overline{m}$ with some non-negligible probability (in k).

Probabilistic Encryption: (see [15] for a formal definition.)
We say that E is a probabilistic encryption there is no poly-time algorithm $\mathcal{A}$ that can distinguish between pairs $(m, E(m))$ and (m, r) for random strings $r \in E(\cdot)$.

Definition 2: Signature Scheme [13, 17] (by exposition of [2])
A digital signature scheme has the following components:

- A *security parameter* k, a *message space* and a *key generation algorithm* as above.
- A *signing scheme* $\mathcal{S} = (S_S, S_R)$, where S_S and S_R are probabilistic p-time algorithms run by the signer vs. the receiver of a signature s on a message m. The signer knows the pair (PK, SK) of matching public and secret keys, and the receiver knows PK.
- A *verification algorithm* $\mathcal{V}$. This is a polynomial time algorithm which given s, m, and PK outputs `true` if s is a valid signature for the message m with respect to the public key PK, and `false` otherwise.

Security:
A signature scheme is *secure* if there is no p-time forger $\mathcal{F}$ who, for infinitely many k, succeeds with a non-negligible probability to forge a signature s on a given message m so that $\mathcal{V}(s, m, PK) = \texttt{true}$.

Security against Adaptive Chosen Message Attacks: [16]
A signature scheme is *secure against adaptive chosen message attacks* if there is no p-time forger $\mathcal{F}$ who, for infinitely many k, after given access to a signature oracle a polynomial number of times, succeeds with a non-negligible probability to forge a signature s on a given message m such that $\mathcal{V}(s, m, PK) = \texttt{true}$, and m was never given to the oracle.

History-Free:
A signature scheme is *history-free*[12] if the signature $\mathcal{S}$ creates on a message is not a function of previous messages signed.

Blind Signature: [7]
A *blind signature* scheme is a pair $S = (S_S, S_R)$ that allows the receiver to obtain a valid signature (m', s') that is statistically uncorrelated to the transcript seen by the signer during the protocol.

Transparently Blindable Signature: [28]
A signature function $S = (S_S, S_R)$ is *transparently blindable* if there exists a blind signature scheme (S_S, S'_R) giving signatures with the same distribution as S does, and the signer's view is indistinguishable for the two protocols, i.e., the signer cannot tell whether he signs regularly or blindly.

Blindfoldable Signature: [28]
The signature scheme $S = (S_S, S_R)$ is *blindfoldable* if there exists a blind signature protocol (S'_S, S'_R) giving signatures with the same distribution as $S = (S_S, S_R)$ does.

Dual Verification Signature Scheme: [28]
A *dual verification signature scheme* is a seven-tuple $(k, \mathcal{M}_k, \mathcal{KG}, S, V_1, V_2, t)$, such that $(k, \mathcal{M}_k, \mathcal{KG}, S, V_1)$ is a signature scheme with publicly verifiable signatures, and $(k, \mathcal{M}_k, \mathcal{KG}, S, V_2)$ is a signature scheme where a signature can only be verified by interaction with a authenticator. Here, (m, s) may be a correct message-signature pair w.r.t. one of the schemes, but not the other. We call t the triggering condition; this decides whether V_1 or V_2 shall be used for verification of a signature.

Definition 3: Challenge Semantics [28]
The challenge semantics of a coin describes the functionality of the coin by assigning different meanings to different bits of the challenge. It is not possible to alter the challenge semantics of a coin once it has been spent.

[12] A signature scheme that is not history free may have problems if used in anonymous e-cash schemes, as shown in [39].

4 Solution

After giving an overview of the scheme, we briefly present the building blocks and their properties, and then introduce the protocols for withdrawing a coin, spending a coin, depositing it, and finally, tracing it if needed.

4.1 Overview

In order to defend against bank robberies, i.e., attacks where a perpetrator either obtains the bank's and ombudsman's secret keys, or obtains coins in a perfectly blinded way (through the use of general secure computation protocols,) we will use a *dual verification signature scheme* where signatures can be publicly verified in the general case, but can be required to be verified interactively with the Bank or Ombudsman in case of an attack. The non-publicly verifiable part of the signature scheme will be implemented using secret lists tagging all valid signatures. However, to guarantee anonymity of the honest user, a threshold of Bank and Ombudsman servers have to cooperate for both individual tracings and for the production of a "valid-list" from such a tag list.

Each coin corresponds to a pair of a secret and public key, authenticated by the Bank/Ombudsman signature during the withdrawal protocol. Using the secret key, the owner of the coin can sign a message, i.e., the challenge given by the shop in the spending protocol. Given the public key of the coin, the shop (as well as other entities at a later point) can verify that this signature is valid, which corresponds to the coin being spent. We can use *challenge semantics*, i.e., special forms of challenges, to implement different functionalities of the coin, such as divisibility, fairness, anonymous micro-payments, etc. All of these functional expansions will be possible to implement without any additional cost. This can be done since the possibility to trace a coin does not depend on the number of challenges responded to (as in schemes implementing perfect anonymity,) but only on agreements on what constitutes a proper spending and what does not (e.g., overspendings.)

4.2 Building Blocks

Coin Signature Scheme:
Each coin will be represented by a secret key / public key pair (x, y), such that for some f, $y = f(x)$, and the keys are associated with a signature scheme S_{Coin} that is *existentially unforgeable*[13], and that has a *public verification algorithm*, V_{Coin}. A lot of schemes satisfy the requirements on the coin signature schemes, e.g., RSA [43], ElGamal [14], DSS [33], and related schemes like [1, 44].

[13] In order to heuristically produce existentially unforgeable signatures using message-recovery schemes, the standard method is to apply a one-way function unrelated to the signature scheme to the message to be signed before the signature is calculated (see also [40]).

Bank/Ombudsman Signature Scheme:
The Bank and the Ombudsman calculate in a distributed fashion a signature $s = S_{B/O}(y)$ on the public key y associated with a properly withdrawn coin and a tag *tag* on the signature. The signature scheme is not *transparently blindable*, is *history-free*, and has a *dual verification scheme* where all the valid message-signatures pairs under V_2 also are valid under V_1. The non-public verification algorithm V_2 is implemented by a list of valid tags to which a potential signed message can be matched to for verification. For security reasons, we want the signature generation to be a distributed function, calculating a signature from shares of the message using shares of the secret. We use so called *magic ink* [29] generation of DSS [33] signatures, e.g., a distributed signature generation employing computational blinding, such that the blinding can be removed by a quorum using an unblinding algorithm. A forgery of such signature can be proven if the majority of signers is honest.

Probabilistic Encryption Scheme:
Let (E_{B_i}, D_{B_i}) denote the probabilistic encryption vs. decryption algorithms of the ith server of the Bank, using the Bank's public key for encryption and its secret key for decryption. Similarly, (E_{O_i}, D_{O_i}) are the public key probabilistic encryption and decryption algorithms of the ith server of the Ombudsman. Any public key encryption scheme can be employed for these. Furthermore, let (E_K, D_K) denote a symmetric key probabilistic encryption/decryption scheme with key K; DES [34] can be used for this. The schemes will be augmented using standard methods [15] for making the encryption probabilistic.

4.3 Protocols

4.3.1 Withdrawing a Coin

A coin is represented by a pair (x, s), where $s = S_{B/O}(f(x))$. In order to perform the withdrawal of a coin, the following protocol is executed:

1. The withdrawer, whom we will call Alice, selects x uniformly at random from the set of possible secret keys, and calculates $y = f(x)$. She proves her identity to the Bank. Alice establishes shared session keys with each server of the Bank and the Ombudsman, e.g., by randomly choosing such keys and encrypting these using the public keys of the intended receivers. (Which can be the same as the public key shares of the signature scheme.)

2. Using the magic ink signature generation scheme, Alice and a quorum of bank and ombudsman servers compute an output so that Alice gets a Bank/Ombudsman signature $s = S_{B/O}(y)$ on y, and the bank and ombudsman servers get a tag *tag*, linked to the signed message, but not linkable by less than a quorum of bank and ombudsman servers. The *tag* is stored in a secure database.

4.3.2 Spending a Coin

A coin (x, s), where x is the secret key of a signature scheme, S_{Coin} with public key y, is spent in the following way:

1. The spender of the coin, Alice, sends (y, s) to the payee, Shop, where $y = f(x)$, and Shop verifies the validity of the message-signature pair (y, s).
2. Shop sends the challenge c to Alice. The challenge may to some part be of a predetermined form, using challenge semantics to implement functions like divisibility etc. (see [28],) and to the other part random and set by Shop.
3. Alice sends the answer $a = S_{Coin}(c, y)$ to Shop, who saves the transcript (y, s, c, a).

4.3.3 Depositing a Coin

A spent coin is deposited by forwarding the transcript (y, s, c, a) to the Bank. The Bank verifies that s is the Bank/Ombudsman signature on y and that a is the y coin signature on c. The Bank further verifies that the same transcript has not been deposited before, and then credits the depositor's account. The Bank then saves the transcript in a database containing all transcripts of non-expired deposited coins, and checks for overspendings of the coin, by summing up the total spendings done for the coin with public key y.

4.3.4 Tracing a Coin

When a coin needs to be traced, the Bank will either (case 1) have as input an identity *id* of a user, and will want to know what coin(s) (y, s) this person withdrew in a certain time interval, or (case 2) have the description (y, s) of a coin and want to know the identity *id* of the withdrawer, or, finally (case 3) determine whether a certain coin (y, s) was withdrawn during a specific withdrawal session. We refer to [29] for technical details.

4.3.5 Alert

After a successful bank robbery, the Bank will alert all shops that they now must use the on-line verification protocol for Bank/Ombudsman signatures, i.e., deposit each coin before it is accepted as a payment. In order not to have to involve the ombudsman for each payment, and in order to be able to use proxies, the Bank and the Ombudsman will produce a list from the list of tags, to which signed messages can be compared for validity check. The list will be produced in a way that does not compromise user privacy (see [27].) Before any coin is accepted, the Bank or a Bank proxy will verify that the coin is in the list of valid coins.

5 Versatility

We noticed in [28] that the representation of a coin by a dual signature, and the separation of two entities Bank and Ombudsman in the system may increase not just the security but also functionality. The distributed nature of the system here keeps this functionality and the system can be made versatile by the use of *challenge semantics*, i.e., the use of some of the bits of the challenge to denote the functionality of the coin. Examples of such a use is to let some of the bits represent the amount spent, thereby implementing *divisibility* trivially and with no extra cost. Similarly, *checks* can be constructed (by assigning a value higher than the value of the coin,) as can *credit* purchases (by using challenge semantics to indicate what account is to be charged.) Furthermore, conditionals can be part of the non-random field, allowing us to implement *surety bonds*, i.e., payments that will only be performed under certain conditions. Challenge semantics can also be used to implement a *fair exchange* [26, 28]. Micro-payments can be implemented by signing the last hash value in a chain of such values; later, the spender can gradually pay by sending over hash preimages to the merchant. Since only one signature generation is needed per chain, and this is relatively light-weight, this scheme is well suited to implement anonymous micro-payments.

6 Efficiency

First, the computational and communicational costs for withdrawing and storing a coin do not depend on the number of times it can be spent. For other schemes offering k-spendability, these costs are linear in k. Using the proposed magic ink DSS scheme, we get the following: Apart from the counter and independently of the value of k, where k is the number of times the coin was spent, the user has to store only one public key (which is between 512 and 1024 bits according to DSS specifications) and one DSS signature, i.e., 320 bits. This is significantly less than the amount stored in most other electronic cash schemes. It is *particularly* competitive for large values of k. Likewise, there is no extra cost associated with divisibility of coins or checks, or with other extended functionality obtained by the use of challenge semantics.

Second, when a user spends a coin, the transcript (y, s, c, a) will have to be transferred first between the user and the shop, then between the shop and the bank, where it will be stored until the coin expires. This adds another 320 bits, plus the length of the challenge, which acts like a contract. (This can be appropriately compressed (resp. expanded) before its usage, using a hash function (resp. a pseudo-random generator).) This, too, compares very well to other schemes.

References

[1] G. Agnew, R.C. Mullin, S. Vanstone, "Improved digital signature scheme based on discrete exponentiation," Electronics Letters, v. 26 1990, 1024-1025.

[2] M. Bellare, S. Micali, "How To Sign Given Any Trap-Door Permutation," Journal of the ACM, Vol. 39, No. 1, Jan 1992, pp. 214-233

[3] S. Brands, "Untraceable Off-line Cash in Wallets with Observers," Proceedings of Crypto '93, pp. 302-318

[4] E. Brickell, P. Gemmell, D. Kravitz, "Trustee-based Tracing Extensions to Anonymous Cash and the Making of Anonymous Change," Proc. 6th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 1995, pp. 457-466

[5] J. Camenisch, U. Maurer, M. Stadler, "Digital Payment Systems with Passive Anonymity-Revoking Trustees," Computer Security - ESORICS 96, volume 1146, pp. 33-43

[6] J. Camenisch, J-M. Piveteau, M. Stadler, "An Efficient Fair Payment System," 3rd ACM Conference on Computer and Communications Security, 1996, pp. 88-94

[7] D. Chaum, "Blind Signatures for Untraceable Payments," Advances in Cryptology - Proceedings of Crypto '82, 1983, pp. 199-203

[8] D. Chaum, H. Van Antwerpen, "Undeniable Signatures," Crypto '89, pp. 212-216

[9] D. Chaum, A. Fiat and M. Naor, "Untraceable Electronic Cash," Advances in Cryptology - Proceedings of Crypto '88, pp. 319-327

[10] CitiBank and S. S. Rosen, "Electronic-Monetary System," International Publication Number WO 93/10503; May 27 1993

[11] A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung, "How to Share a Function Securely," STOC 94, pp. 522-533

[12] Y. Desmedt and Y. Frankel, "Threshold cryptosystems," Crypto '89, pp. 307–315.

[13] W. Diffie and M. E. Hellman, "New Directions in Cryptography," IEEE Trans. Info. Theory IT-22, Nov. 1976, pp. 644-654

[14] T. ElGamal "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms," Crypto '84, pp. 10-18

[15] S. Goldwasser and S. Micali, "Probabilistic Encryption & How To Play Mental Poker Keeping Secret All Partial Information," Proceedings of the 18th ACM Symposium on the Theory of Computing, 1982, pp. 270-299

[16] S. Goldwasser, S. Micali and R. Rivest, "A 'Paradoxical' Solution to the Signature Problem," 25th Annual Symposium on Foundations of Computer Science, 1984, pp. 441-448

[17] S. Goldwasser, S. Micali and R. Rivest, "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks," SIAM Journal of Computing 17(2), April 1988, pp. 281-308

[18] IBM Research, "iKP - A Family of Secure Electronic Payment Protocols', The First USENIX Workshop on Electronic Commerce, New York, July 1995

[19] A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, M. Yung, "Proactive Public Key and Signature Systems," ACM Comp. and Comm. Security '97, pp. 100-113

[20] A. Herzberg, S. Jarecki, H. Krawczyk, M. Yung, "Proactive Secret Sharing, or How to Cope with Perpetual Leakage," Crypto '95, pp. 339-352

[21] Y. Frankel, Y. Tsiounis, M. Yung, "Indirect Discourse Proofs: Achieving Efficient Fair Off-Line E-Cash" Asiacrypt '96, pp. 286-300

[22] M. Franklin and M. Yung, "Towards Provably Secure Efficient Electronic Cash," Columbia Univ. Dept of C.S. TR CUCS-018-92, April 24, 1992

[23] M. Franklin and M. Yung, "Blind Weak Signatures and its Applications: Putting Non-Cryptographic Secure Computation to Work," Eurocrypt '94

[24] N. Ferguson, "Extensions of Single-term Coins," Crypto '93, pp. 292-301

[25] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, "Robust Threshold DSS Signatures", Eurocrypt '96.

[26] M. Jakobsson, "Ripping Coins for a Fair Exchange," Eurocrypt '95, pp. 220-230

[27] M. Jakobsson, "Privacy vs. Authenticity," PhD Thesis, University of California, San Diego, Department of Computer Science and Engineering. See: `http://www.cse.ucsd.edu/users/markus/`.

[28] M. Jakobsson and M. Yung, "Revocable and Versatile Electronic Money," 3rd ACM Conference on Computer and Communications Security, 1996, pp. 76-87

[29] M. Jakobsson and M. Yung, "Distributed 'Magic Ink' Signatures," Advances in Cryptology - Proceedings of Eurocrypt '97, 1997.

[30] S. H. Low, N. F. Maxemchuk and S. Paul, "Anonymous Credit Cards," The Second ACM Conference on Computer and Communications Security, Nov. 1994, pp. 108-117

[31] G. Medvinsky and B. C. Neuman, "Netcash: A design for practical electronic currency on the Internet," The First ACM Conference on Computer and Communications Security, Nov. 1993, pp. 102–106.

[32] G. Medvinsky and B. C. Neuman, "Requirements for Network Payment: The NetCheque^{TM} Perspective," Compcon '95, pp. 32-36

[33] National Institute for Standards and Technology, "Digital Signature Standard (DSS)," Federal Register Vol 56(169), Aug 30, 1991

[34] NBS FIPS PUB 46, "Data Encryption Standard," National Bureau of Standards, U.S. Department of Commerce, Jan 1977

[35] T. Okamoto and K. Ohta, "Disposable Zero-Knowledge Authentication and Their Applications to Untraceable Electronic Cash," Advances in Cryptology - Proceedings of Crypto '89, 1990, pp. 481-496

[36] T. Okamoto and K. Ohta, "Universal Electronic Cash," Advances in Cryptology - Proceedings of Crypto '91, 1992, pp. 324-337

[37] T. Okamoto, "An Efficient Divisible Electronic Cash Scheme," Crypto '95, pp. 438-451

[38] R. Ostrovsky and M. Yung, "How to withstand mobile virus attacks," Proc. of the 10th ACM Symposium on the Principles in Distributed Computing, 1991, pp. 51-61.

[39] B. Pfitzmann and M. Waidner, "How to break and repair a "provably secure" payment system," Crypto '91.

[40] D. Pointcheval, J. Stern, "Security Proofs for Signature Schemes," Eurocrypt '96, pp. 387-398

[41] R. Rivest, "The MD5 Message Digest Algorithm," RFC 1321, Apr. 1992

[42] R. Rivest, A. Shamir, "PayWord and MicroMint–Two Simple Micropayment Schemes", Manuscript.

[43] R. Rivest, A. Shamir, L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," Communications of the ACM, v. 21, n. 2, Feb 1978, pp. 120-126

[44] C. P. Schnorr, "Efficient Signature Generation for Smart Cards," Crypto '89, pp. 239-252

[45] M. Sirbu and J. D. Tygar, "NetBill: An Internet Commerce System Optimized for Network Delivered Services," Compcon '95, pp. 20-25

[46] S. von Solms and D. Naccache, "On Blind Signatures and Perfect Crimes," Computers and Security, 11 (1992) pp. 581-583

[47] M. Stadler, J-M. Piveteau, J. Camenisch, "Fair Blind Signatures," Advances in Cryptology - Proceedings of Eurocrypt '95, 1995

Appendix :Claims and Sketches of Proofs

Theorem 1:
The system achieves *unforgeability*, i.e., a set of users, shops, and bank and ombudsman servers, not including a quorum of the latter, are not able to perform payments for a value exceeding $\mathcal{V}$, which are later accepted by an honest bank as valid, after engaging in withdrawal protocols withdrawing funds for a value of $\mathcal{V}$.

Lemma 1a:
For each spendable coin (x, s), the Bank cooperated to produce one signature.

Proof of Lemma 1a:
Assume the contrary. For each valid coin, we need a Bank/Ombudsman signature on the public key of the coin. Since a quorum by assumption must contain at least one Bank server, it must then be the case that it is possible to construct a valid coin given only the Ombudsman's share(s) of the secret needed to produce a signature. Since both the Bank and the Ombudsman shares are selected uniformly at random, and therefore, the Ombudsman share of the secret is statistically uncorrelated to the secret key, this would by a simulation argument imply that valid coins could be produced without any secret knowledge. □

Lemma 1b:
It is only possible to produce a new correct challenge-response pair (c, a) for a coin with public key y if the corresponding secret key x is known, even if a polynomial number of correct pairs $\{(c_i, a_i)\}$ have been seen, where the c_i's are set by the attacker.

Proof of Lemma 1b:
Assume the contrary. This means that, given the public key y of the coin, it is possible to produce a message-signature pair (c, a) not earlier seen after seeing a polynomial number of such correct pairs. This is impossible since the coin signature scheme, (S_{Coin}, V_{Coin}), is assumed to be existentially unforgeable. □

Lemma 1c:
For each deposited coin, the Bank will find out its corresponding public key y and the value of the spending, and will know the total value of all the spendings of each coin that has been deposited.

Proof of Lemma 1c:
When a coin is spent and deposited, its public key y must be sent, by the specification of the protocol. The correctness of the public key will be authenticated by the combined Ombudsman and Bank signature s on it. The corresponding secret key, x, must for each spending be used to produce a signature a on the challenge c of the spending. Only coins of this format can be accepted by the Bank for credit when deposited. Using a simple book-keeping procedure, the Bank will know the total value of the spendings, each one indexed by its coin public key y. □

Proof of Theorem 1:
By Lemma 1c, each deposited coin must have a valid Bank/Ombudsman signature on its public key in order to be accepted. By Lemma 1a, at least one bank server must be involved in the generation of this signature. By Lemma 1b, the withdrawer needs to generate a new signature for each new payment and deposit transcript. According to Lemma 1c, the Bank can tell payment transcripts of different coins from each other by their public key y, and will know the total sum spent by its owner for each coin y. Therefore, it is not possible to overspend coins without being detected – the scheme achieves unforgeability. □

Theorem 2:
The system achieves impersonation safety, i.e., if there is no attacker consisting of a quorum of bank and ombudsman servers, or if transactions can be legally challenged by taking the case to a judge, then no coalition of users, shops, bank and ombudsman servers can succeed in charging an honest user more than his withdrawals total.

Proof of Theorem 2:
According to the assumptions, the signature scheme used to sign the challenge is existentially unforgeable, and the user employs a sound identification scheme to identify herself before withdrawals.
Therefore, if only a user $A \in S_1$ has the representation (x, s) of a coin, then only A can produce valid payment transcripts of that coin. Thus, in order to produce a valid payment transcript for the coin associated with A, the attacker S_2 needs to know the corresponding representation (x, s). Since this cannot be obtained by S_2 during a normal withdrawal (or this would give a method to break the encryption scheme employed), and only A can authenticate herself as A, then a quorum of ombudsman and dishonest bank servers must be involved in the attack, and sign a public key y of a coin claimed to be associated with

A, but known by the bank servers not to be.
Furthermore, if the user sends a signed acknowledgment of having received the withdrawn coin after having obtained a valid pair (x, s), but not before that, and only A can produce this signature, then if the bank does not have this signature on a withdrawal, then the bank does not have a complete withdrawal view (making any claims of improper use invalid,) which would be recognized by a third party, such as a judge. □

Theorem 3:
The system achieves overspending detection, i.e., if a set S_1 of attackers performs an overspending attack in which a value $\mathcal{V}$ is withdrawn and a value $\mathcal{V}_1 > \mathcal{V}$ is spent, then a quorum of the bank and ombudsman servers will be able to establish (a) the sum of the overspending, i.e., $\mathcal{V}_1 - \mathcal{V}$, and (b) the identity of at least one member of S_1.

Proof of Theorem 3:
The bank will not accept a deposit unless it is of the proper format, (y, s, c, a), where $s = s_{B/O}(y)$, and $a = s_y(c)$, where the former is the bank/ombudsman signature on y, and the latter the signature, using the secret key corresponding to y, on c. Since the bank/ombudsman signature scheme by assumption is existentially unforgeable, if the bank receives such a deposit for a public key y, the same public key must have been previously signed by a bank/ombudsman quorum. The value of the spending, which will be associated with the public key y, will be obvious to the bank given the valid deposit transcript, since the challenge specifies this amount and the coin signature scheme is assumed to be existentially unforgeable, too. Therefore, if a coin y is overspent, the bank will know this, as soon as a value exceeding the legal value has been deposited for y. Furthermore, the bank will be able to lower-bound the amount of the overspending at this time; the precise value of the overspending will be known at the end of the expiration time (for deposits) of the coin. □

Theorem 4:
The system achieves overspending robustness, i.e., if a set S_1 of attackers perform an overspending attack in which a value $\mathcal{V}$ is withdrawn and a value $\mathcal{V}_1 > \mathcal{V}$ is spent, then a set S_2 of users, shops, bank and ombudsman servers, disjoint from S_1, cannot make an honest quorum of bank and ombudsman servers running the overspending detection protocol output a sum of overspendings $\mathcal{V}_2 - \mathcal{V} > \mathcal{V}_1 - \mathcal{V}$ and only identities of participants in S_1.

Proof of Theorem 4:
For each time the user spends a coin, he will have to give a signature on a challenge using the secret key of the coin. As long as only a polynomial number of signatures are given, we have that, according to the definition of an existentially unforgeable signature scheme, that it will not enable an adversary to sign a new

message. □

Theorem 5:
The system achieves traceability, i.e., any bank and ombudsman quorum can, regardless of the actual withdrawal protocol used, and regardless of whether an "illegal untraceability" attack is executed, perform the following actions: (a) given a coin description, find the corresponding withdrawal session, (b) given a withdrawal session, find the corresponding coin description, and (c) given a coin description and a withdrawal session, verify whether these correspond to each other. All of these actions are performed in a way that does not give any set of participants a non-negligible advantage (apart from the advantage gained by knowing the *result* of the above calculation) in establishing any other coin-session relationship.

We refer to [29] for the proof.

Theorem 6:
The system achieves revocability, i.e., any traced coin can be permanently (resp. temporarily) made unspendable by blacklisting (resp. freezing).

Proof of Theorem 6:
Since, by Theorem 5, *any* coin can be traced at any time after its withdrawal, it will be possible for the Bank to construct a list of coins (or rather, their corresponding public keys y) not to be accepted (whether temporarily or permanently) and distribute this (in an authenticated form) to all the shops. Similarly, it will be possible for the Bank to remove coins from this list (corresponding to thawing of funds) by broadcasting a signed list of coins on the black list to be taken off the latter (or just an updated version of the black list, not containing descriptions of the thawed coins.) □

Theorem 7:
The system achieves anonymity, i.e., the probability for any coalition of participants not containing a quorum of bank and ombudsman servers to determine whether a particular coin description corresponds to a particular withdrawal session is not non-negligibly better than that of a guess, uniformly at random, given all the available pairs of descriptions of coin descriptions and withdrawal session descriptions, and all the already computed relations available.

We refer to [29] for the proof.

Theorem 8:
The system achieves framing-freeness, i.e., it is not possible for a set S_2 of dishonest users, shops, bank and ombudsman servers, not including any member of a set S_1 to produce a set of transcripts, that, if a tracing is performed with these as input, the output would identify a member of S_1 with non-negligible

probability if (a) the withdrawer has a public key associated with him, and he signs withdrawals, or (b) there is no quorum of ombudsman and dishonest bank servers.

Whereas the requirement 'impersonation safety' is not implying or implied by the requirement 'framing-freeness', the proof that the system satisfies impersonation safety also proves that it satisfies framing-freeness. We therefore refer to the proof of Theorem 2.

Theorem 9:
The system achieves refundability, i.e., if S_2, a set of users, shops, bank and ombudsman servers, not including any member of a set S_1, makes only a value $\mathcal{V}_1 < \mathcal{V}_2$ be accepted as valid by an honest set of bank servers, after members of S_1 spend funds with a value $\mathcal{V}_2 \leq \mathcal{V}$ from the said withdrawals, then members of S_1 can prove that the attack took place, resulting in the identification of at least one of the members of S_2.

Proof of Theorem 9:
If a user $A \in S_1$ does not obtain a valid signature s on the public key y during a withdrawal, then he can prove this by presenting the session keys to the bank and ombudsman servers, who will then be able to identify the cheater. This is the provable forgery nature of the signing protocol in [29].
If a user $A \in S_1$ has a valid pair (x, s), and is blocked from spending this coin to its full value, then he can complain to the ombudsman or a third party. Since the coin signature scheme is assumed to be existentially unforgeable, and only A can know (x, s), then the bank will not be able to produce valid payment transcripts of the coin to a value exceeding that spent by the user (see the proof of Theorem 2,) and the bank will be identified as a cheater.
After a bank robbery, the bank will have the a list of descriptions of all properly withdrawn coins (but only these). This is guaranteed by the robustness of the the batch blinding protocol. Therefore, the bank can not be misled by a cheating ombudsman not to accept coins that are valid, or accept them to a lesser value. Thus, if a participants blocks some valid payments from being accepted, the identity of the cheater will be known. □

Remark: It is clear that the shop cannot change the semantics of a coin, or it would be able to change the challenge in the basic system, thus obtaining two valid coins (or more) to deposit, instead of one. Since the extensions are only based on the semantics (which is a convention how to set the challenge, agreed upon by all participants) the security of the extended system is unchanged.