

Trustee Tokens: Simple and Practical Anonymous Digital Coin Tracing

Ari Juels

RSA Laboratories
Bedford, MA 01730, USA
E-mail: ari@rsa.com

Abstract. We introduce a trustee-based tracing mechanism for anonymous digital cash that is simple, efficient, and provably secure relative to its underlying cryptographic primitives. In contrast to previous schemes, ours may be built on top of a real-world anonymous cash system, such as the DigiCashTM system, with minimal modification to the underlying protocols. In addition, our scheme involves no change to the structure of the coins. On the other hand, our scheme requires user interaction with a trustee, while many other such systems do not. This interaction occurs infrequently, however, and is efficient both in terms of computation and storage requirements. Our scheme also achieves more limited security guarantees in the presence of malicious trustees than many other systems do. While this is a disadvantage, it represents a tradeoff enabling us to achieve the high level of practicality of our system.

Keywords: anonymity, blind digital signatures, coin tracing, digital cash, e-cash, trustee-based coin tracing

1 Introduction

Anonymous digital cash, known informally as *e-cash*, is a form of digital currency that provides anonymity to users with respect to both merchants and banking institutions, thereby affording a heightened assurance of consumer privacy. Since David Chaum first proposed the idea in 1982 [9], it has been a major focal point of academic research in electronic commerce. Researchers observed early on, however, that if anonymity in payment systems is unconditional, it may be exploited to facilitate crimes like blackmail and money laundering [28]. This observation spurred research into the idea of making anonymity in payment systems conditional, and, in particular, revocable by a third party or *trustee* under court order. This notion, known as *trustee-based coin tracing*, was developed by Brickell, Gemmell, and Kravitz [5] and independently by Stadler, Piveteau, and Camenisch [27]. A National Security Agency report has since declared the availability of such tracing in e-cash systems vital to the security interests of the United States [19]. The importance of traceability in e-cash systems has motivated the proposal of many new trustee-based coin tracing schemes, among them [6, 7, 8, 12, 13, 15, 17, 16, 22, 23].

Several trustee-based coin tracing schemes have offered notable innovations in functionality and flexibility. In general, however, these improvements are achieved at the cost of diminished simplicity and practicality. In this paper, we introduce a simple and highly efficient trustee-based tracing mechanism that may be added on top of anonymous cash schemes based on blind RSA signatures. Rather than seeking to offer new functionality with respect to other tracing schemes, our scheme does the opposite: it trades off some functionality against a higher degree of simplicity and practicality. Thus, while our scheme has limitations with respect to some previous ones, it also has several important advantages:

- **Practicality** Unlike previous schemes, ours can be incorporated straightforwardly on top of a commercially implemented on-line anonymous e-cash scheme, namely the DigiCashTM scheme [1, 26] (also referred to as Chaumian e-cash [9]). Our scheme involves no change to the structure of the coins or the spending or deposit protocols, and can be easily applied to off-line e-cash variants as well.
- **Efficiency** Our scheme imposes minimal computational overhead on the underlying withdrawal scheme for the user – essentially just several modular multiplications and a MAC. Most other tracing schemes carry overhead for the user amounting to several modular *exponentiations* per transaction.
- **Provability** In contrast to other schemes in the literature, our system is provably secure with respect to underlying cryptographic primitives. We state theorems treating both the anonymity and non-forgability properties of our scheme in Section 4.
- **Simplicity** Our scheme is conceptually very simple.

Like most other schemes, ours can support both tracing of the identity of a user from a coin, known as *coin tracing*, and generation of a list of all coins belonging to a given user, known as *owner tracing*. Both of these operations require very little computation and database access.

The tracing mechanism we propose in this paper has two shortcomings with respect to other schemes. First, our scheme requires user registration with a trustee upon set up of the user’s account (and possibly again later, if the user spends a large number of coins). While some systems, such as, e.g., [17], require on-line participation of trustees, others, like [12], do not. As a result of this interaction between user and trustee, our system requires storage of a small amount of authorization data for withdrawals, which many other systems do not. A second drawback to our scheme is its limited privacy guarantees when multiple trustees are used. We require the use of what amounts to a *trusted dealer* (see, e.g., [4]) upon user registration. This is discussed in Section 5.2.

The idea behind our scheme is quite simple. Before making a withdrawal at a bank, a user contacts a trustee. The user shares with the trustee secrets used to generate a coin x and a blinding factor r . The user receives from the trustee what is called a *trustee token*. A trustee token is a piece of information entitling the user to withdraw an anonymous coin generated using r and x . It is

essentially a short (say, 10-50 bit) proof to the coin issuer, i.e., bank, that the coin in question can be traced by the trustee. By requesting many trustee tokens in advance of coin withdrawals, and batching trustee tokens so that they apply to multiple coins, the user can achieve a very low frequency of interaction with the trustee.

1.1 Previous Work

As mentioned above, trustee-based tracing schemes were first elaborated independently by Brickell et al [5] and Stadler et al [27]. Brickell et al describe two schemes. The first is based on a blind Schnorr-like signature scheme and requires interactive proofs between trustees and the bank. The second is based on blind RSA signatures and makes use of a cut-and-choose protocol, resulting in a scheme that is flexible, but has large coin sizes and computational requirements.¹ Stadler et al introduce several schemes, the most practical of which makes use of a blind signature scheme based on that of Chaum and Pedersen [11]. In their scheme, the user requests a pseudonym and registration information from a trustee. The user presents this registration information to the bank, and also incorporates it into the coins she withdraws. Although use of a pseudonym for multiple withdrawals can lead to linkage of user identity across coins, this problem can be addressed in part by having the user register multiple pseudonyms.

Jakobsson and Yung [15] introduce the notion of “challenge semantics”, enabling flexible determination of coin value, so that coins can be invalidated in case of, e.g., a bank robbery. Their scheme is capable of addressing stronger attack models than many others and a wider range of commercial settings. It is also adaptable to use with any underlying digital signature scheme. On the other hand, their scheme requires on-line participation of a trustee in both coin withdrawal and coin spending. The “Magic Ink” construction of the same authors makes use of blind DSS signatures [17]. In this scheme, signing and anonymity revocation can be conducted by differing quorums of trustees. Trustees are again, however, fully on-line, and the scheme is also rather computationally intensive for most operations. In [16], Jakobsson and Yung show how to combine the benefits of Magic Ink signatures with those of challenge semantics.

Camenisch, Piveteau, and Stadler [8] introduce a slightly different approach to trustee-based tracing. They propose a system, based on blind Schnorr signatures, in which a user transfers funds from a non-anonymous to an anonymous account, and a trustee is capable of linking the two accounts. The chief disadvantage of this approach is that once the two accounts are linked, anonymity is eliminated.

Camenisch, Maurer, and Stadler [6] demonstrate a system, based on blind Schnorr signatures, in which the trustee is wholly off-line. Their system is quite complex, and involves well over a dozen modular exponentiations by the user at each coin withdrawal. A system with very similar properties was introduced

¹ Often overlooked in the literature is the fact that the RSA-based system of Brickell et al is very likely the first with fully off-line trustees.

independently by Frankel et al [13]. Davida et al [12] improve on the system in [13], reducing the computation required in the withdrawal protocol, as well as the database search requirements in owner tracing. Their withdrawal protocol, however, still requires over a dozen modular exponentiations by the user.

Most of the above schemes rely on discrete-log based blind signature schemes. The exceptions are those in [5] and [15], which can make use of blind RSA signatures. Both schemes, however, involve changes or additions to the underlying structure of the coins, and have the inefficiencies mentioned above. In contrast, our scheme may also be used in conjunction with blind RSA signatures, but does not require any modification to the underlying coin structure. This is the reason why our scheme may be quite practically adopted in conjunction with DigiCashTM and like e-cash systems.

Trustee participation in our scheme is minimal, limited to interaction between the user and trustee upon account set-up and perhaps on an infrequent basis afterward. Our scheme does not suffer the communications and computational overhead of a scheme like that in [17], but, on the other hand, has a small amount of overhead not present in, e.g., [12]. Additionally, our scheme requires storage of trustee information, but this information is not sizeable, particularly with respect to the large coin sizes of many trustee-based tracing schemes.

The chief advantage of our scheme is its overall efficiency. The user requires minimal computation on coin withdrawal – as much as one hundred times less than in schemes like [6, 12]. Computational and storage requirements for the bank are also comparable to or smaller than in most other schemes. Tracing is also highly efficient in our scheme, more so than in most others. In the case of coin tracing, for instance, our scheme requires no database lookups, which most other schemes do.

Although papers describing schemes with off-line trustees do not discuss the issue at any length, many such trustee-based tracing schemes allow for multiple trustees or a distributed trustee in a strong privacy model. A notable exception is that of Stadler et al [27]; our scheme is similar in this regard. When deployed with multiple trustees, our scheme essentially requires distribution of owner tracing information through a trusted dealer, as discussed in Section 5.2.

The basis of our scheme is in fact quite similar in flavor to that of Stadler et al [27]. The key idea behind both schemes is to use trustee registration prior to coin withdrawal. While the Stadler et al scheme makes use of a digital signature where ours uses a MAC, ours can, of course, be adapted to use a digital signature, as discussed in Section 4.1. The Stadler et al scheme differs crucially from ours in that the trustee registration is part of the coin structure. Hence their scheme is bound to, rather than added on top of the underlying signature scheme. It may be used with a special-purpose discrete-log based blind signature scheme, but not straightforwardly with blind RSA signatures.

We stress in general, however, that our scheme does not really bear direct comparison to most previous trustee-based tracing schemes. Rather, it seeks to strike a different balance, sacrificing some flexibility and privacy guarantees in favor of heightened efficiency and practicality.

1.2 Organization

The remainder of this paper is organized as follows. Section 2 gives notation and definitions. We describe the details of our trustee-based tracing scheme in Section 3, and discuss security issues in Section 4. In Section 5, we discuss the efficiency of our system and also describe a means of incorporating multiple trustees.

2 Background

2.1 Notation

An anonymous digital cash scheme involves the following participants. The first three - namely the Bank, the Trustee, and the User - will be central to the description of our scheme in this paper.

- The *Bank* is a financial institution that issues anonymous digital coins and manages accounts. The Bank publishes an RSA modulus $N = pq$, whose factorization it alone knows.
- The *Trustee* (which we denote by T) is a trustworthy device or agency responsible for tracing coins on presentation of a valid court order. In our scheme, the Trustee holds a secret key SK_T for some public key encryption algorithm; it publishes the corresponding public key PK_T . The Trustee also holds a symmetric key w , which it shares with the Bank.
- The *User* (whom we denote by U) is any entity that withdraws money from the Bank. In our scheme, the User possesses a unique identifier or account number denoted by ID_U , and also a secret s_U associated with ID_U and used to prove the User's identity. The Trustee knows a binding of ID_U to the User's real-world identity, although the Bank may transact with the User on an entirely anonymous basis using only account information.
- The *Government* is any law enforcement body that is authorized to request the tracing of coins on presenting a valid court order to T .
- The *Merchant* is any party with whom the User spends money. Our trustee-based tracing scheme does not involve any modification to the underlying digital cash protocols involving the Merchant, i.e., the spending and deposit protocols. We therefore do not have cause to discuss the role of the Merchant at any length in this paper.

We use f to denote a secure one-way or hash function, and $MAC_w(m)$ to denote a MAC (Message Authentication Code) computed using a symmetric key w . (See [21] for definitions and a discussion of MACs.) $E_{PK_X}(m)$ will indicate the encryption (under some appropriate asymmetric cipher) of the message m using the public key PK_X . We let PS denote an indexed pseudo-random generator (although a chained generator can easily be adapted to our schemes as well), and $PS_A(i)$ stand for the output of this generator with secret seed A on index i . We write $\{X_i\}$ to mean the set of values X_i over all appropriate values of i .

The symbol $\|$ will denote concatenation of strings, \oplus , the XOR operation, and \in_R , uniform random selection from a set.

There are three security parameters in our scheme, denoted by k_1 , k_2 , and k_3 . The parameter k_1 is the length of the seed to the pseudo-random number generator PS , and thus, as we shall show, specifies the level of security on the User's anonymity. The parameter k_2 specifies the length of the digital signature modulus N used by the Bank for signing coins, and thus the hardness of existential forgery in our scheme. The parameter k_3 specifies the length of the trustee tokens or MACs used in our scheme. This is essentially equivalent to the level of security on the Trustee's ability to trace the User's coins.

We write $1/2 + 1/\text{poly}$ to denote a probability greater than or equal to $1/2 + 1/k^c$, where c is some constant and k is the pertinent security parameter. For example, in the proof of Theorem 1, the security parameter in question is k_1 , the pseudo-random seed length. Using somewhat rough notation, we write $< 1/2 + 1/\text{poly}$ to indicate a probability which is asymptotically less than $1/2 + 1/k^c$ for any constant c .

2.2 Definition of Blindness

Informally, a digital cash scheme is *blind* or *anonymous* if the Bank is unable to determine, either at the time of withdrawal of a coin, or later, upon examining circulating or deposited coins, which coin was withdrawn by which user. Chaum [9] first put forth the notion of blind signatures in connection with payment schemes, demonstrating an RSA-based signature scheme, described in Section 2.3 of this paper, that is unconditionally blind. A number of papers, e.g., [24], have described a weaker notion of blindness informally in terms of a lack of statistical correlation between the view of the signer at the time of signing and the set of produced signatures. A more formal definition of computational blindness, proposed in [18], may be described in terms of the following experiment. The User produces two messages m_0 and m_1 of length polynomial in k_1 . The User sets a bit b uniformly at random. In two arbitrarily interleaved (and presumed blind) digital signature protocols, she presents the documents m_0 and m_1 to the Bank in an order specified by b , i.e., in the order $\{m_b, m_{1-b}\}$. In this interaction, she obtains from the Bank signatures $s(m_0)$ and $s(m_1)$ on the two messages. The User presents the message/signature pairs $(m_0, s(m_0))$ and $(m_1, s(m_1))$ to the Bank. The Bank then attempts to guess the bit b . If no polynomial-time algorithm exists which enables the Bank do so with probability $1/2 + 1/\text{poly}$ (over its own coin-flips and those of the User), then we say that the digital signature scheme is blind (or secure with respect to anonymity).²

² As noted in [18], by standard hybridization arguments this definition is as general as one involving polynomially many withdrawals by polynomially many users.

2.3 Blind RSA Signatures

Blind digital signatures were introduced by Chaum [9] as a means of implementing anonymous digital cash. As explained above, the anonymous digital cash schemes of Chaum et al, as in [9] and [10], make use of blind RSA signatures. In these schemes, the Bank publishes a public modulus $N = pq$, for which it alone knows the factorization; it creates RSA signatures in this modulus. Depicted in Figure 1 is Chaum's protocol enabling the User to obtain a blind RSA signature (or coin) with public exponent 3 from the Bank. This coin takes the form $(x, f^{1/3}(x))$. All computations here are mod N .

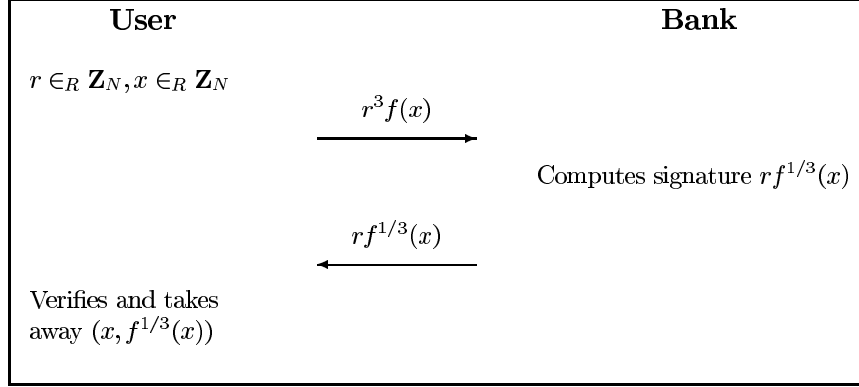


Figure 1. Blind RSA Signature Protocol.

Note that this protocol is unconditionally blind, i.e., the blindness does not rely on computational assumptions or statistical arguments.

2.4 DigiCash™

There are several DigiCash™ variants, not all of which have been implemented commercially. In this paper, we describe trustee-based tracing as it applies to a commercially implemented, on-line version of DigiCash™.

In the version of DigiCash™ (Chaumian e-cash) we consider in this paper [1, 26], a coin consists of an RSA signature by the Bank on the hash of a message x . If, for instance, the Bank uses a public (encryption) exponent of 3, then a valid coin would assume the form $(x, f^{1/3}(x) \bmod N)$. To distinguish among different denominations in this scheme, the Bank uses different public exponents, e.g., $(x, f^{1/3}(x) \bmod N)$ might indicate a \$.50 coin, while $(x, f^{1/17}(x) \bmod N)$ indicates a \$1 coin. The full system works as follows. The User with identifier ID_U authenticates herself in a secure manner to the Bank. She then withdraws coins using the blind RSA signature protocol described above, the Bank deducting the corresponding funds from her account. To spend a coin with the Merchant, the

User simply transmits it to the Merchant. To prevent double spending of a coin, the Merchant verifies on-line with the Bank that the coin is still valid. While this description overlooks some details, the reader may find a complete explanation of the DigiCash™ protocols in [26]. As explained there, newer variants of the DigiCash™ system have begun to make use of a redundancy function f with message recovery. This coin structure can be accommodated with only minor modifications to the scheme we present here.

Note that there are digital cash systems other than DigiCash™ that rely on the use of RSA signatures, e.g., the “X-cash” scheme described in [14]. Our trustee token scheme may be applied equally well to the anonymous variants of such systems.

3 Our Scheme

3.1 Key Ideas

Before interacting with the Bank in our system, the User obtains from the Trustee a set of trustee tokens $\{M_i\}$. Recall from Section 1 that a trustee token is essentially a short (say, 10-50 bit) proof that the blinded information the User is presenting to the Bank has been seen and its correctness verified by the Trustee. This proof is presented to the Bank when the User withdraws a coin. It reveals no information to the Bank about the coin the User obtains. Note that for the sake of simplicity, we assume in our presentation that one trustee token is used for each coin withdrawal. As we show later, however, a single token can in fact be used for multiple coins.

Our scheme makes use of an enhancement to improve communications and storage efficiency. On performing a withdrawal from the Bank, the User generates her coin and blinding data pseudo-randomly from random seeds R and S . In her interaction with the Trustee, it therefore suffices for the User just to transmit R and S : the Trustee is then able to generate all of the desired trustee tokens. The seeds R and S constitute all of the data required by the Trustee to perform owner tracing against the User. Note that R and S are given as separate seeds for notational convenience. In practice, they may be combined into a single seed.

Recall that in the anonymous withdrawal protocol described above, to obtain a coin $(x_i, f^{1/3}(x_i) \bmod N)$, the User sends to the Bank the blinded quantity $r_i^3 f(x_i) \bmod N$. In our protocols, the quantity x_i will contain the User’s identifier ID_U encrypted under the public key of the Trustee. This will facilitate coin tracing by the Trustee. We let $x_i = E_{PK_T}(ID_U \parallel s_i)$, where $s_i = PS_S(i)$. Hence x_i may be computed using S and ID_U . The integer r_i is generated from the random seed R . In particular, $r_i = PS_R(i)$.

We are now ready to present the trustee token and coin withdrawal protocols. Note that all computations are performed $\bmod N$. To simplify notation, we omit explicit indication of this fact.

3.2 Protocols

Trustee token withdrawal In the trustee token withdrawal protocol, the User proves her identity to the Trustee. She then reveals the secret seeds R and S used to generate the pseudo-random data for withdrawals, and indicates the number j of tokens she wants. The Trustee computes a set of trustee tokens $\{M_i\}_1^j$ on these seeds for $1 \leq i \leq j$. He sends these tokens to the User. The protocol should take place over an authenticated and encrypted channel to prevent compromise of R and S . Figure 2 depicts the trustee token withdrawal protocol.

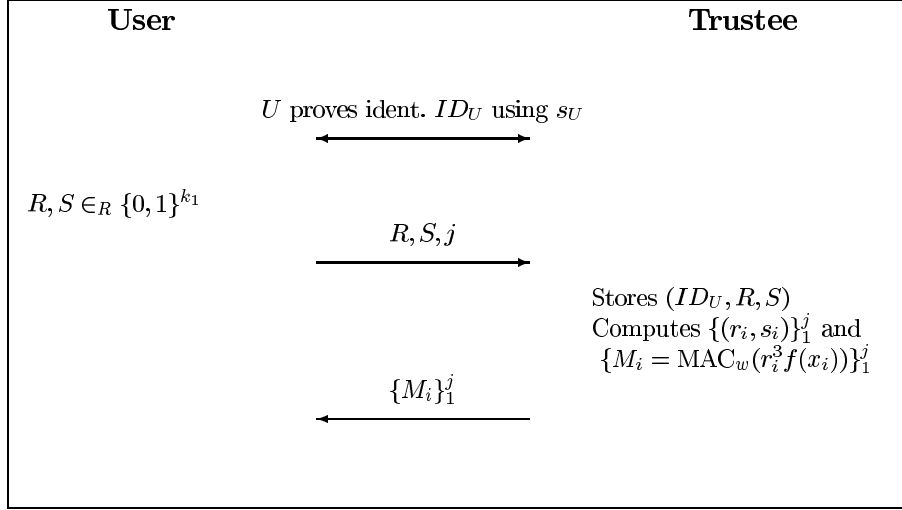


Figure 2. Trustee Token Withdrawal.

Observe that the User can request and store a large number of tokens, since tokens are small (again, say, 10-50 bits apiece). These tokens may be used for future withdrawals without the need for additional contact with the Trustee until the User exhausts her supply. Note also that although we present the token withdrawal protocol as an interaction with an on-line trustee, this need not be the case. For example, trustee tokens can be requested using a secure store-and-forward system, or, alternatively, loaded on a smart card by the Trustee.

Coin withdrawal protocol The coin withdrawal protocol in our scheme is essentially the same as in the underlying Chaumian e-cash protocol. The only difference is that the Bank verifies, by means of a valid trustee token, that the User's withdrawal request has been authorized by the Trustee. To prevent theft of the User's coins, the coin withdrawal protocol should take place over an authenticated and encrypted channel (on which even the Trustee cannot eavesdrop). Figure 3 depicts the coin withdrawal protocol.

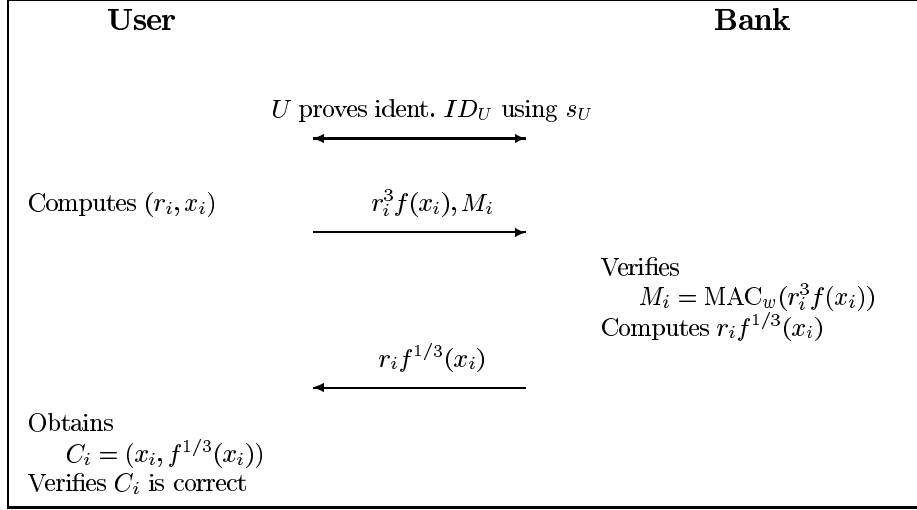


Figure 3. Coin Withdrawal Protocol.

We do not present coin spending and deposit protocols for our scheme, since these are exactly as in the underlying scheme.

Off-line e-cash As mentioned above, our scheme can be applied fairly straightforwardly to off-line variants of Chaumian e-cash, such as [10]. Off-line schemes involve the User's embedding tracing information in coins that gets revealed when a coin is double-spent. To employ trustee tokens, then, in an off-line scheme, it suffices for the User to generate this tracing information from the seed S . Note that our tracing scheme presents the possibility of making off-line systems more efficient by allowing the Trustee to verify coin information incorporated to prevent double-spending. We leave further details to the reader.

Tracing Both coin and owner tracing are straightforward in our system. To trace a coin C_i , the Government presents it (along with a valid court order) to the Trustee. Since $x_i = E_{PK_T}(ID_U || s_i)$, the Trustee may extract ID_U from a coin C_i simply by performing a decryption with its secret key SK_T . To perform owner tracing, the Government presents ID_U (along with a court order) to the Trustee. The Trustee then uses S to compute all $\{x_i\}$. This is sufficient to identify all coins withdrawn by the User ID_U .

Observe that owner tracing in our scheme has an interesting property. In contrast to many other schemes, it is possible for the Trustee to identify not only all coins withdrawn by the User and subsequently deposited, but also all coins in current circulation, and even some coins to be withdrawn by the User in future transactions.

4 Security

We discuss four aspects of the security of our scheme. These are:

1. **User anonymity** The Bank should not be able to extract any information from its interactions with users that reveals which coins have been withdrawn by whom. This property was described more formally above in terms of the definition of blindness given above from [18]. As we shall show, the security of user anonymity in our system is dependent on k_1 , the security parameter, i.e., seed length in bits, of the pseudo-random number generator. (An acceptable level of security may be achieved by letting k_1 be about 80.)
2. **Non-forgability of coins** The users of the system and the Trustee, even in collaboration, should not be able to mint coins without express participation of the Bank. The hardness of forgery in our system is determined by security parameter k_2 , equal to the length of the modulus N . (The security parameter k_2 may safely be set at 1024.)
3. **Traceability** The Trustee should be able to perform both coin tracing and owner tracing with high probability. The security of tracing in our system is determined by security parameter k_3 , equal to the length of the trustee tokens in bits, and by security parameter k_2 . (For most purposes, it would be acceptable to let k_3 be 10-50 bits.)
4. **Inability of Trustee to steal User's cash** Although the Trustee should be able to link the User to her coins, he should not be able to steal her coins or make withdrawals from her account. It is common practice in the literature not consider efforts by the Bank to steal the User's money: it is assumed that the Bank, which has control of the User's account in any case, must be trusted in this respect.

We are able to prove that our system is secure in all four of the above senses, relative to underlying cryptographic primitives. Due to constraints of space, we provide only proof sketches.

In the underlying blind RSA signature scheme, anonymity is unconditional, i.e., information theoretically secure. In particular, use of the blinding factor r ensures that the Bank receives no information about withdrawals. In our system, however, we introduce a pseudo-random number generator PS to enhance efficiency. We can prove the security of anonymity relative to that of PS .

First some definitions. Let PS_S denote the output (of an appropriate length) of PS given seed S . Let A be a polynomial time algorithm that outputs a 0 or a 1. Let $EX_Z[A(M)]$ denote the expected output of algorithm A given input M over uniform random choices of Z , i.e., the probability that $A(M) = 1$. By a definition equivalent to those in [20], the pseudo-random generator PS may be said to be broken if a polynomial time algorithm A may be found such that $|EX_S[A(PS_S)] - EX_R[A(R)]| = 1/poly$. This notion may be used in a formal proof of the following theorem.

Theorem 1. *If the Bank is able to break the anonymity in our scheme, then it can break the pseudo-random generator PS .*

Proof (sketch): User anonymity is information theoretically when the User employs purely random inputs. If user anonymity can be broken when the User employs a pseudo-random number generator, then it is possible to use the trustee token scheme to distinguish between random and pseudo-random inputs. This is equivalent to breaking the pseudo-random number generator PS . ■

We address the issue of forgeability in the following theorem.

Theorem 2. *It is as hard in our scheme for the User and Trustee to forge coins collaboratively as it is for the User to do so herself in the underlying blind digital signature scheme.*

Proof (sketch): Formal proof of this theorem depends upon the fact that the User can simulate the establishment of a secret key between Trustee and Bank. In carrying out the withdrawal protocol in the underlying scheme, the User can then simulate both the role of the Trustee and the role of the Bank in processing trustee tokens. Thus, in our scheme, the Trustee can provide no additional information useful to the User in committing forgery. ■

Our next theorem regards the assurance of the Trustee that the User cannot cheat and evade tracing of her coins.

Theorem 3. *Suppose that the User is able to produce a coin which cannot be traced by the Trustee. Then the User was successful at either forging a coin or forging a MAC.*

Proof (sketch): If the User is able to produce an untraceable coin C , then either the User forged C , or the User withdrew C from the Bank. If the User withdrew C , and C is untraceable, then the User must have provided the Bank with an invalid MAC. ■

By Theorem 2, it is presumed that the User cannot forge a coin in polynomial time with more than a negligible probability. Therefore the probability that a given coin C held by a cheating User is untraceable is roughly equal to the User's ability to forge a MAC. Under reasonable assumptions (see, e.g., [2, 3]) this is about 2^{k_3} , where k_3 is the length of the trustee tokens in our scheme. Thus, a 10-bit trustee token should yield a probability of less than 1/1000 of the User being able to evade tracing for a given coin. For most law-enforcement purposes, this should be adequate, particularly as law enforcement officers are likely to have multiple coins available for tracing in most scenarios. Moreover, the User can only verify the correctness of a MAC by interacting with the Bank. Her efforts at cheating are thus likely to be detected by the Bank before she obtains an untraceable coin. For very high security applications, it may be desirable,

however, to use MACs of up to, say, 50 bits. (Note that while MACs of 32 or even 64 bits are typical in most financial transactions, the risks in such cases are much greater, involving the potential for many millions of dollars to be misdirected.)

Finally, we consider the ability of the Trustee to steal the User's money. Although the Trustee has access to the secrets of the User employed to generate coin data, the Trustee does not have access to the User secret s_U . Therefore, the Trustee cannot impersonate the User and withdraw money from the User's account without the Bank's collusion. This yields the following theorem.

Theorem 4. *It is infeasible for the Trustee to steal money from the User's account without the collusion of the Bank or the User.*

Proof (sketch): The Bank only withdraws money from the User's account if authorized to do so through a channel authenticated by means of the User's secret s_U . This channel is authenticated so that no eavesdropper, even the Trustee, can steal the User's coins. ■

4.1 Untrustworthy Trustee

In our exposition above, we assume the trustworthiness of Trustee, and that the MAC held jointly by the Trustee and Bank is well protected. In some scenarios, it may be desirable to make weaker security assumptions. We can achieve this – at the cost of computational and storage efficiency – with more extensive record keeping and use of digital signatures by the Trustee.

Compromised MAC If an attacker manages to seize the MAC shared by the Bank and the Trustee, and the theft goes undetected, he can present false Trustee authorization to the Bank. This would enable him to withdraw untraceable coins. There are many possible defences against this attack, representing a spectrum of tradeoffs between efficiency and security. One possible countermeasure would be to refresh the shared MAC on a frequent basis. Another option would be to have the Trustee digitally sign tokens using a public key signature algorithm, rather than MACing them. While this variant on our scheme provides stronger security guarantees, is rather less efficient in terms of the computation required to produce a token.

Failed tracing In the above variants of our scheme, if the Government presents a coin C to the Trustee and the coin cannot be traced, it is unclear whether the Trustee or the Bank permitted withdrawal of an untraceable coin. This can be remedied by having the Trustee digitally sign all tokens, and requiring the Bank to keep track of all coin withdrawal transcripts for each User and the Trustee to keep track of the number of token withdrawals. If untraceable coins surface, then the Bank and Trustee records can be compared, and a determination made as to whether the cheating party is the Bank or the Trustee.

Framing by Trustee In our scheme as described above, the Trustee can frame the User. In particular, the Trustee can generate a value x based on a pair of values (r_i, s_i) not yet employed by the User (or based on false seed pair (R', S')), withdraw from the Bank a coin C_i based on x_i , spend the coin on some illicit purchase, and then claim that the User was responsible, adducing the registered seed pair (R, S) as evidence.

It is possible to prevent attacks of this nature as follows. We have the User digitally sign (R, S) . We then have the Bank record all blinded values presented for signing, as well as the number of withdrawals by the User. The Bank rejects any withdrawal request based on a previously presented blinded value. Now if Trustee attempts, under a false identity, to withdraw a coin C_i (based on a User's seed pair (R, S)) when C_i has already been withdrawn, he will be stopped. If he withdraws a coin C_i not already withdrawn by the User, then the User is able to prove, on revealing (R, S) and adducing the Bank's counter as evidence, that she was not responsible for the initial withdrawal of C_i . Hence, framing of the User would require collusion between both the Bank and the Trustee.

Note that framing of the User by the Bank is infeasible, as the Bank has no knowledge of S .

5 Efficiency and Extensions

As explained above, our trustee-based tracing scheme adds very little computational overhead to the underlying coin withdrawal protocol. The Bank must compute a MAC which it would not otherwise compute, but this requires negligible effort in comparison with generation of the signature on the coin. The User must compute the values r_i and x_i from a pseudo-random generator, but these values would likely be computed in some pseudo-random fashion in any case. In fact, if the Bank uses a signature exponent of 3, the User need only compute two pseudo-random values, a hash, six modular multiplications, and a modular inversion per withdrawal (including verification that it has a valid coin). In contrast, the scheme in [12], for example, requires fifteen (160 bit) modular *exponentiations* on the part of the User at the time of withdrawal - and even more if the scheme is to permit owner tracing.

The token withdrawal process requires no computationally intensive cryptographic operations - just a few hashes and computations of MACs. The storage requirements for trustee tokens are also minimal. The Trustee must store a pseudo-random seed for each User (perhaps 80 bits). In the scheme described above, the User must store perhaps 10 bits for each trustee token. A coin in Chaum's scheme consists of roughly 1000 bits [26]. Hence, the storage of a collection of trustee tokens will not be difficult on a device capable of storing anonymous digital cash. In fact, at 10 bits per token, 1K bytes of memory is enough to store more than 800 trustee tokens.

5.1 Improving Efficiency

For the sake of simplicity, we have assumed in the above description of our scheme that one trustee token is used for every withdrawal. It is possible to improve the storage efficiency of trustee tokens substantially by making a single trustee token good for multiple withdrawals, at the cost of some linkage of User identity across coin. Suppose, for instance, that the User always withdraws coins in multiples of ten. We let $M_j = MAC_w(r_{10j}^3, f(x_{10j}), r_{10j+1}^3 f(x_{10j+1}), \dots, r_{10j+9}^3 f(x_{10j+9}))$. The token M_0 is good for the first ten withdrawals, M_1 for the next ten withdrawals, etc. In fact, it is not even necessary for the User to withdraw coins in groups of ten. If she wishes to use a trustee token to withdraw fewer than ten coins, she need only send an appropriate number of blank withdrawal requests, i.e., a sequence of $r_i^3 f(x_i)$ for which she does not wish to receive the corresponding coins. If enough batching is performed, it may be efficient to use digital signatures instead of MACs, eliminating the need for secret key establishment between the Trustee and Bank.

5.2 Multiple Trustees

Our scheme as described has only a single trustee. It is possible, however, to achieve k -out-of- n tracing with any number n of trustees by sharing the secret SK_T among the n trustees in an initialization phase. It is also necessary, however, to share R , S , and ID_U for each user after the corresponding trustee tokens have been distributed. Sharing may be performed using any of a number of threshold and/or proactive secret-sharing techniques, such as, e.g., [25].

While flexible, this method of incorporating multiple trustees into our scheme achieves weaker security guarantees than in many other systems, such as [12]. In [12] and related schemes, secrets enabling compromise of the User's identity are never revealed to the trustees, except during tracing. In the scheme we describe above, the secrets R and S are revealed by the user, and must then be shared among trustees. These secrets, which enable owner tracing (but not coin tracing), are thus vulnerable to attack during the token withdrawal protocol. Hence the entity handling the user secrets acts essentially like a trusted dealer. This is generally acceptable if sufficient controls on handling of user secrets are set in place. A more elegant and robust approach to incorporation of multiple trustees into the proposed scheme would nonetheless represent a desirable advance.

Acknowledgements

The author wishes to express thanks to Markus Jakobsson, Burt Kaliski, and Berry Schoenmakers for their helpful comments and suggestions.

References

1. DigiCash, Inc. Web site. <http://www.DigiCash.com>, 1998.
2. M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology - CRYPTO '96*, pages 1–16. Springer-Verlag, 1996. LNCS No. 1109.
3. M. Bellare, R. Guerin, and P. Rogaway. XOR MACs: New methods for message authentication using finite pseudo-random functions. In Don Coppersmith, editor, *Advances in Cryptology - CRYPTO '95*, pages 15–28. Springer-Verlag, 1995. LNCS No. 963.
4. D. Boneh and M. Franklin. Efficient generation of shared RSA keys. In Burton S. Kaliski, Jr., editor, *Advances in Cryptology - CRYPTO '97*, pages 425–439. Springer-Verlag, 1997. LNCS No. 1294.
5. E.F. Brickell, P. Gemmell, and D. Kravitz. Trustee-based tracing extensions to anonymous cash and the making of anonymous change. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 457–466, 1995.
6. J. Camenisch, U. Maurer, and M. Stadler. Digital payment systems with passive anonymity-revoking trustees. In *Computer Security - ESORICS '96*, pages 31–43. Springer-Verlag, 1996. LNCS No. 1146.
7. J. Camenisch, U. Maurer, and M. Stadler. Digital payment systems with passive anonymity-revoking trustees. *Journal of Computer Security*, 5(1):254–265, 1997.
8. J. Camenisch, J.-M. Piveteau, and M. Stadler. An efficient fair payment system. In *3rd ACM Conference on Computer Communications Security*, pages 88–94. ACM Press, 1996.
9. D. Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology - CRYPTO '82*, pages 199–203. Plenum, 1982.
10. D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In Shafi Goldwasser, editor, *Advances in Cryptology - CRYPTO '88*, pages 319–327. Springer-Verlag, 1988. LNCS No. 403.
11. D. Chaum and T. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92*, pages 89–105. Springer-Verlag, 1992. LNCS No. 740.
12. G. Davida, Y. Frankel, Y. Tsiounis, and M. Yung. Anonymity control in e-cash systems. In Rafael Hirschfeld, editor, *Financial Cryptography '97*, pages 1–16. Springer-Verlag, 1997. LNCS No. 1318.
13. Y. Frankel, Y. Tsiounis, and M. Yung. Indirect discourse proofs: Achieving fair off-line e-cash. In M. Y. Rhee and K. Kim, editors, *Advances in Cryptology - Proceedings of ASIACRYPT '96*, pages 286–300. Springer-Verlag, 1996. LNCS No. 1163.
14. M. Jakobsson and A. Juels. X-cash: Executable digital cash. In Rafael Hirschfeld, editor, *Financial Cryptography '98*. Springer-Verlag, 1998. To appear.
15. M. Jakobsson and M. Yung. Revocable and versatile e-money. In *3rd ACM Conference on Computer Communications Security*. ACM Press, 1996.
16. M. Jakobsson and M. Yung. Applying anti-trust policies to increase trust in a versatile e-money system. In Rafael Hirschfeld, editor, *Financial Cryptography '97*, pages 217–238. Springer-Verlag, 1997. LNCS No. 1318.
17. M. Jakobsson and M. Yung. Distributed Magic-Ink signatures. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97*, pages 450–464. Springer-Verlag, 1997. LNCS No. 1233.

18. A. Juels, M. Luby, and R. Ostrovsky. Security of blind digital signatures. In Burton S. Kaliski, Jr., editor, *Advances in Cryptology - CRYPTO '97*, pages 150–164. Springer-Verlag, 1997. LNCS No. 1294.
19. L. Law, S. Sabett, and J. Solinas. How to make a mint: the cryptography of anonymous digital cash. Technical Report 96-10-17, National Security Agency, 1996. Available at <http://www.ffhsj.com/bancmail/bancpage.html>.
20. M. Luby. *Pseudorandomness and Cryptographic Applications*. Princeton University Press, 1996.
21. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
22. D. M'Raihi. Cost-effective payment schemes with privacy regulation. In M. Y. Rhee and K. Kim, editors, *Advances in Cryptology – Proceedings of ASIACRYPT '96*, pages 266–275. Springer-Verlag, 1996. LNCS No. 1163.
23. D. M'Raihi and D. Pointcheval. Distributed trustees and revokability: A framework for internet payment. In Rafael Hirschfeld, editor, *Financial Cryptography '98*. Springer-Verlag, 1998. To appear.
24. D. Pointcheval and J. Stern. Provably secure blind signature schemes. In M. Y. Rhee and K. Kim, editors, *Advances in Cryptology – Proceedings of ASIACRYPT '96*, pages 252–265. Springer-Verlag, 1996. LNCS No. 1163.
25. T. Rabin. A simplified approach to threshold and proactive RSA. In Hugo Krawczyk, editor, *Advances in Cryptology - CRYPTO '98*, pages 89–104. Springer-Verlag, 1998. LNCS No. 1462.
26. B. Schoenmakers. Basic security of the ecashTM payment system. In Bart Preenel et al., editors, *Computer Security and Industrial Cryptography: State of the Art and Evolution, ESAT Course*, pages 338 – 352, 1998. LNCS No. 1528. Corrected version available on-line at <http://www.win.tue.nl/~berry/papers/cosic.ps.gz>.
27. M. Stadler, J.M. Piveteau, and J. Camenisch. Fair blind signatures. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology - EURO-CRYPT '95*, pages 209–219. Springer-Verlag, 1995. LNCS No. 921.
28. B. von Solms and D. Naccache. On blind signatures and perfect crimes. *Computers and Security*, 11(6):581–583, 1992.