Money Conservation via Atomicity in Fair Off-Line E-Cash

Shouhuai Xu * Moti Yung ** Gendu Zhang *** Hong Zhu †

Abstract. Atomicity and fault tolerance issues are important and typically open questions for implementing a complete payment scheme. The notion of "fair off-line e-cash" (FOLC) was originally suggested as a tool for crime prevention. This paper shows that FOLC schemes not just enable better control of e-cash when things go wrong due to "criminal suspicion" and other "regulatory/legal" issues, but it can also assure atomicity which takes care of conservation of money in case of failures during transaction runtime. The added protocols are very efficient and quite simple to implement. This kind of piggybacking atomicity control over "anonymity revocation" makes good sense as both actions are done by off-line invocation of the same trustees (**TTP**s). The resulting solution is a comprehensive yet efficient solution to money conservation in electronic cash transactions based on FOLC schemes. The adopted *recovery* approach makes the involved participants (customer, bank, merchant) sure that they can "re-think" the transactions when things go wrong, implying the atomicity of the transactions. We also take an optimistic approach achieving *fair exchange* costing only 2-round of communicational complexity (trivially the lower bound) with no additional **TTP** involvement since FOLC already employs such a party.

Keywords Atomicity, Payments Systems, E-cash, Conservation of Money, E-Commerce Transactions Processing, Recoverability, Fair Off-Line E-Cash (FOLC)

1 Introduction

Any system should be robust enough to enable the completion of given tasks in the presence of faults and attacks. This is especially important in the context of e-cash based applications where certain messages actually represent real money. Task completion is not a major problem in face-to-face physical money transactions. In the context of account-based [ST95] and token-based [BGJY98] *on-line* e-commerce systems, where there is an *on-line* trusted third party (**TTP**), the

* Department of Computer Science, Fudan University, Shanghai, P. R. China. shxu@fudan.edu.cn
** CertCo, NY, NY, USA. moti@cs.columbia.edu
*** Department of Computer Science, Fudan University, Shanghai, P. R. China. gdzhang@fudan.edu.cn
† Department of Computer Science, Fudan University, Shanghai, P. R. China. hzhu@fudan.edu.cn

issue is relatively easy to handle. On the other hand, in off-line e-cash based systems the issue becomes subtle. It is so because off-line e-cash schemes provide user anonymity, which implies that identifying parties via anonymity revoking for the sake of achieving *atomicity* or assuring *fair exchange* is totally unacceptable. This issue has never been dealt with in full generality and was only partially addressed by [CHTY96] (though with somewhat poor efficiency due to the use of generic methods which cause a heavily overloaded *on-line* **TTP** or log center) in the setting of (unconditionally anonymous) off-line e-cash based systems. The goal of the current paper is to present a much more efficient and complete solution to this problem. The approach we take is exploiting the properties of fair off-line e-cash (FOLC) schemes proposed in [FTY96, JY96].

We notice that the inefficiency of [CHTY96] is due to the adopted *two-phase commit* idea which is well-known in the area of database systems. While that tool is a powerful general purpose mechanism for atomicity in any transaction system, it is unnecessary in the context of FOLC based systems. As we will see, the efficiency of our solution comes from this observation: we need no "on-line" **TTP**, yet the functions implemented in [CHTY96] (say, fair exchange) are nevertheless provided. Our solution is comprehensive since we take the entire system into consideration. Moreover, we take an optimistic approach to *fair exchange* in the cost of only two-round communication (which is a trivial lower bound) while still without on-line **TTP**. Even in the occasional (and indispensable) invocation of the off-line **TTP** to handle a dispute, it can at times be casted by the party which is anyway responsible for such invocation due to an anonymity revocation request which is part of FOLC.

1.1 The Problem

Why hasn't off-line e-cash been adapted in practice so far? Many answers are possible, let us review some of the technical difficulties. Jakobsson and M'Raihi [JM98] have noticed that *conditional anonymity* and *card-ability* (whether they are efficient enough to be implemented using smart card) are two difficulties. Here we argue that *the assurance of conservation of money* (a kind of robustness or survivability) in e-cash transactions is another difficulty. Without promising money conservation, arguably nobody will dare to take the risk to use the systems as money can be lost and uncertainty prevails. This is justified by the analogous efforts for atomicity (and reduction of uncertainty about transaction completion) in generic distributed transactions in database systems.

Let us see some possible scenarios in off-line e-cash based systems. In the withdrawal protocol, what happens if the network crashes while the customer, **U**, is awaiting the e-coins? **B**, the bank, may have debited **U** [1] while **U** obtaining no coin. So, his [2] money is lost. On the other hand, **U** may have received the

[1] If **B** debits **U** after receiving the acknowledgment from **U**, then **U** may get coins without being debited (say, **U** will not send the acknowledgment). Even if special measures are taken, similar problems still exist.

[2] In this paper, customers are male, shop and **TTP** are female.

e-coins while claiming not to have received it! Thus, the money of the bank may be lost. We see that atomicity is needed to avoid both over-debiting and double-withdrawal.

In a purchase transaction, both the customer and the shop may agree on the price and electronic goods [3] after some negotiation. Seeing the encrypted e-goods from the shop, the customer optimistically sends his e-coins to her and waits for (her share of) the decryption key (see section 4 and 5). What is the dilemma for them if the network crashes (or when the customer's PC malfunctions, or even when there is a denial-of-service attacker) while neither **U** is acknowledged by the shop (**S**) for his e-coins nor **S** is acknowledged by **U** for her share of the decryption key [4]? Due to the complexity of the causes, there are two possible states for this transaction.

- **S** has received the e-coins. Since **U** does not know the state of the transaction, what can **U** and **S** do? (1) **S** will deposit the e-coins. If **U** spends these coins at another time, he is bound to face an accusation of double-spending in the scheme of [FTY96] (alternatively, possibly over-spending in the scheme of [JY96][5]). So, **U** better dare not spend the coins and, effectively, his money is lost. (2) **S** will not deposit the e-coins as she has received no acknowledgment for the key from **U**, effectively she sells her e-goods without income (a denial-of-service attacker may gain benefit from it as well). As a result, her money is lost.
- **S** has not yet received the e-coins so she trivially will not deposit them. However, **U** still dare not spend his e-coins at another transaction because he fears taking the risk of being accused a double-spender or an over-spender.

Furthermore, even if **U** knows that **S** will not deposit the coins if she has not received the acknowledgment for the key, though now he will not face an accusation of double-spending (or over-spending) when he spends the e-coins again, his ID may still be revealed in some schemes (say, [FTY96]) by colluding shops.

While, intuitively, the broken transaction can be recovered through a special protocol, how about if the transaction information (including the URL, price, contract) is lost due to either hardware or software crashes of the customer's poor PC? He can not recover it even if he wants to.

[3] In this paper we focus on e-goods while the ideas can also be adapted to the setting of physical goods.

[4] We note that the reliability mechanism at the transport layer (say, TCP) will not help us in this case. The reason being that the verification for the validity of either the e-coins or the decryption key (therefore the decrypted e-goods) is executed at the application layer, so the acknowledgment must also be from the peer application layer entity. Indeed, the acknowledgment for the valid e-coins is always the transmission of (the share of) the decryption key itself in practice. On the other hand, the underlying connection itself may have been released due to a timeout.

[5] For example, if we assume that the avaliable value of a coin is \$4 while its denomination is \$10 and the failure transaction is of \$2, if he spends at another transaction \$3, then he may be accused of being an over-spender and his anonymity is revoked.

In the deposit, the shop should be assured that when she sends the e-coins, she is bound to be credited correspondingly. What happens if the network crashes before **B** acknowledging **S**? **S** will not be sure if, and how much, her account has already been credited correctly.

To summarize, numerous mishaps and problems occur when atomicity of a payment related transaction is not assured.

1.2 Our Solution

In this paper we present an efficient solution to the above problem via *withdrawal atomicity*, *payment-delivery atomicity* [6], and *deposit atomicity* in FOLC schemes [FTY96, JY96] based transactions. We assure that while the security and anonymity of the original schemes are still maintained, our solution is efficient in that there needs no (additional) on-line **TTP** even in the dispute handling protocol [7] for fair exchange.

This paper, therefore, shows that FOLC schemes do not just enable better control of e-cash when things go wrong due to "criminal suspicion" and other "regulatory/legal" issues. Rather, the same parties that take care of this can also assure atomicity and conservation of money in case of failure during the transactions runtime. This kind of piggybacking atomicity control over "anonymity revocation" makes good sense as both actions are done by off-line invocation of the same **TTP**s. This implies that FOLC schemes ease atomicity in e-cash systems.

It has also been noted in [W98] that the similarity between database transactions and commercial ones, unfortunately, always misleads people to simple minded implementations. While the traditional destructive *prevention-and-recovery* approach (say, two-phase commit [BN97]) to *atomicity* in database community is powerful enough, it provides poor efficiency in our e-cash context (as we mentioned above). Instead, our simple *recovery* philosophy makes it much more efficient than the traditional approach. We comment that just as mentioned in [CHTY96], the basic properties of transaction systems which accompany atomicity: CID (which stands for: Consistent, Isolated, and Durable, refer to [BN97] for details), are naturally satisfied in e-cash transactions and therefore are omitted in the rest of this paper.

This paper also shows that the well known problem of *fair exchange* (refer to [ASW98]) is not so hard to implement in the context of "exchange fair off-line e-coin for e-goods". In our solution, the *fair exchange* need only two rounds of communication in real-life. That is, one sends the money, and the other sends the

[6] As we focus on a total solution to e-goods based e-commerce, we incorporate delivery into payment. In the context of physical goods, the payment atomicity is naturally implied by the solution of this paper. As for fair exchange or atomic delivery, it is obviously out of the reach of the e-system by itself.

[7] On one hand, according to [BP89], pure fair exchange is unachievable and dispute handling must come to rescue in electronic commerce. On the other hand, the off-line **TTP** in this paper can be performed by any Trustee for conditional anonymity in the original schemes.

e-goods (or the share of the decryption key). It is even better than the 3-round protocol most recently proposed by Bao et al. in [BDM98]. Furthermore, 2-round communicational complexity is the trivial lower bound for any fair exchange protocol as, even in the real world with physical cash, it is still necessary. The trick is that the (off-line) bank plays a key role, i.e., certifying the reciept of money.

1.3 Outline

In section 2 we briefly review the related work. The basic ideas, definitions, and assumptions used in this paper are presented in section 3. Our atomicity providing extensions to the electronic coin schemes of [FTY96, JY96] are proposed respectively in sections 4 and 5. We conclude in section 6.

2 Related Work

2.1 E-Cash schemes

The basic electronic cash system consists of three (probabilistic polynomial-time) parties: bank **B**, user or customer **U**, and shop or merchant **S** , and three main procedures: withdrawal, payment, and deposit. Users and shops maintain their respective accounts with the bank, while:

- **U** withdraws electronic coins (digital cash) from his account, by performing a withdrawal protocol with the bank **B** over an authenticated channel.
- **U** spends a coin by participating in a payment protocol with a shop **S** over an anonymous channel.
- **S** performs a deposit protocol with the bank **B**, to deposit the user's coin into her account.

The original electronic cash aimed at offering some level of user anonymity during a purchase with the intention to emulate electronically the properties of physical cash exchange. The first generation of electronic cash systems are proposed in e.g., [CFN88, OO91, FY93, B93, F93]. These schemes provide anonymity in information theoretic sense through the useful primitive called blind signature [C82] or its variants. Realizing that the complete anonymity can be abused to commit "perfect crimes" (money laundering, blackmailing, bribery taking, even bank robbery [vSN92, JY96]) a way to avoid perfect anonymity was sought out. In [BGK95, SPC95, CPS96, JY96, J97, FTY96, T97, CFT98], conditional anonymity (alternatively, fair off-line digital cash, revokable anonymity) schemes with various extensions have also been proposed. It seems that while the first generation of e-coin schemes focuses on privacy, the second one pays much attention to balance privacy and robustness (against abuse), and the notion of trustees who may revoke anonymity was put forth.

The *indirect discourse proofs* technique was used by Frankel, Tsiounis, and Yung in [FTY96] to offer the Bank the confidence that the Trustee will be able to

trace in case tracing is needed. Further such schemes are in [DFTY97,dST98,FTY98]. Another recent extension is presented in [T97, CFT98] where the divisibility properties from [OO89, OO91, O95] are incorporated. At the same time, Jakobsson and Yung presented a versatile electronic cash scheme in [JY96], and its *magic ink signature* version in [JY97, J97]. This latest scheme tries to cope even with the strong bank robbery attack by introducing trustees outside the bank system.

2.2 Previous Approaches to Atomic E-Commerce Transactions

Tygar [T96] is the one who first noticed the problem of *atomicity*, then [CHTY96] is the first work aimed at supporting atomicity for electronic commerce transactions based on off-line e-cash schemes. Though the authors used a heavily overloaded inefficient on-line **TTP** (called log center), this is nevertheless the first solution realizing the three classes[8] of atomicity proposed in [T96], namely:

- **Money atomicity**: Transactions feature atomic transfer of electronic money, i.e., the transfer either completes entirely or not at all;
- **Goods atomicity**: Transactions are money atomic and also ensure that the customer will receive goods iff the merchant is paid;
- **Certified delivery**: Protocols are goods atomic and also allow both the customer and merchant to prove exactly what was delivered. If there is a dispute, this evidence can be shown to a judge to prove exactly what goods were delivered.

This problem has been paid attention to also in the easier settings of on-line schemes. For example, atomicity for on-line account-based transactions is addressed in [ST95], whereas atomicity for on-line token-based transactions is given in [BGJY98].

Following are a few remarks regarding the early work:

1. The solution proposed in [CHTY96] is weakened by the fact that they use an impractical variant of the blind coin from [C82, CFN88] (first generation e-cash) as their building block. It is well known that these schemes are subject to the potential trouble of money laundering, blackmailing and bribery committing attack (to overcome which FOLC was invented).

2. Furthermore, the solution proposed in [CHTY96] is not double-withdrawal robust. For example, the customer claims after receiving the e-coins that he has never seen them. Due to the perfect anonymity of the underlying blind signature, the bank is unable to trace the e-coins. Therefore, the money of the bank is lost.

3. It is already mentioned in[CHTY96] that the "ripping coin" idea proposed by Jakobsson in [J95] is not *money atomic*. This is because each of the half protocols themselves may be interrupted, leaving the digital cash again in an ambiguous state (alternatively, in a blocking state, using the terms of the area of transaction processing [BN97]).

[8] The last one is partially implemented.

4. The ideas proposed in [BBC94] against lost of money due to wallet loss or malfunctioning can be used as an orthogonal building block, and be added to our solution. That is, we concentrate on the protocol transactions and we do not consider the possibility of money lost resulted from crashes of the e-coin wallets. We assume this kind of fault tolerance is provided.

3 Preliminary

3.1 Basic Ideas

As mentioned above, the users of an electronic commerce system must be assured (in addition to security and privacy) that each transaction executes either completely or not at all (i.e., *atomicity*) implying that their money will not be lost due to the system itself. As any electronic commerce application has to involve some delivery process, we incorporate delivery of e-goods into payment in this paper resulting in an efficient payment-delivery scheme. We attempt to realize a comprehensive atomicity (including *withdrawal atomicity*, *payment-delivery atomicity*, and *deposit atomicity*) as non-atomicity may occur at any phase due to any malfunctioning within the system or even an external (denial of service) attack. The basic philosophy used in our solution is *recovery* rather then *prevention-and-recovery* of the database community (as in [CHTY96]) for the sake of efficiency.

While we use a simple "resolve protocol" for *payment-delivery atomicity*, a "dispute handling protocol" for fair exchange in any protocol (even in the real world [BP89]) is also adopted. We still present a secure (against denial-of-service attack) "refund function" in case the customer needs to either refund his e-coins or exchange them for fresh e-coins in certain e-cash schemes.

Definition 31 *Let τ_0, τ be global time, Δ is some (implicitly or explicitly) predesigned period. Let us assume that an off-line e-coin based transaction begins at τ_0, whereas $\tau = \tau_0 + \Delta$. A withdrawal protocol is said to have withdrawal atomicity if the customer's account is debited iff he receives the corresponding e-coins within τ. A payment-delivery protocol is said to have payment-delivery atomicity if the payee receives the coins iff the payer receives the e-goods (namely, the decryption key) within τ. A deposit protocol is said to have deposit atomicity if the payee is credited with certain amount as well as the bank receives the corresponding e-coins within τ.*

Remarks:

1. We focus on the "practical value" of a coin rather than its "face value". Namely, the customer in a withdrawal protocol may get some additionally legal coins (in a mathematical sense), but he dare not spend them as he is bound to face the *tracing of owner* (refer to **Theorem 2** and **4**). We note again that, as mentioned before, the withdrawal protocol in [CHTY96] is not double-withdrawal robust. Another case, in a payment-delivery protocol, the shop may "obtain" some coins and will not deposit them. It is said that she does not receive

the coins as the "practical value" of the coins is zero though in a mathematical sense they are valid. It is interesting to note that its mirror exists in the setting of physical cash in the real world.

2. Obviously, if we consider only a payment protocol, the corresponding definition for atomicity can be easily adapted from the above one for payment-delivery. This is omitted in the rest of this paper due to space limitation.

3. The *money atomicity* defined in [T96] is trivially implied by our definitions. (Thus, we will not explicitly discuss it.) Consequently, the assurance for the conservation of money and fair exchange (i.e., *goods atomicity*) in e-cash based transactions is indirectly implied by our definitions. In the context of this paper, the so-called *certified delivery* is directly implemented by the transaction atomicity and fair exchange in a "destructive" way which means that, though the customer can abuse the complaint facility, he can obtain no additional useful advantage (e.g., e-goods) whereas the shop has to be honest since the bank can present the records of her coin income. This suggests that carefully designed protocols can prevent (rather than resolve after the fact) some abuses in protocols for certain complicated problems (e.g., certified delivery) at least in certain settings (say, e-coin vs. e-goods).

3.2 Assumptions

All the computational complexity assumptions for security of the transactions are naturally inherited from the underlying e-cash schemes [FTY96, JY96], therefore, they are claimed respectively in the context while necessary or omitted while it is clear from the context.

Additionally, we assume that there exist anonymous communication channels for anonymity or privacy. Such a channel is typical in the literature, say, [CHTY96, FTY96, T97, CFT98, JY96, J97, JM98][9].

We assume that a transaction may be broken in an arbitrary fashion. It is also assumed that the money of the customer will not be lost due to the crash of his PC, as he has a fault tolerant e-wallet of [BBC94]. In other words, while the transaction information may be lost due to the malfunctioning of his PC, his money is still available. On the other hand, the shop server is assumed to be powerful enough to address either hardware or software malfunctioning. This is plausible due to the fact that many fault tolerant mechanisms for servers are available.

[9] It should be noted that, no anonymous channel is needed in the context of the original [JM98]. However, here we consider the whole process in an electronic commerce application rather than solely the payment protocol as in [JM98].

4 Atomicity Extension to [FTY96, T97, CFT98] E-Cash Scheme

4.1 The Extended Payment-Delivery Protocol

We first present a small extension of the original payment protocol to incorporate the delivery mechanism. It is easy to see that this extension compromises neither security nor privacy. The original protocols are copied in **Appendix 1**. Though the original deposit protocol also needs to be updated to include checking for the validity of the payment (not yet expired), due to space limitation, we omit this extension here.

1. **U** and **S** negotiate through some interaction, and agree on the price, the goods description *desc*, and invalid date/time $\tau = \tau_0 + \Delta$ where τ_0 is the time that the interested transaction begins and Δ is the negotiated lifetime for it. Alternatively, τ is the deadline for **S** to deposit the received e-coins.
2. **U** sends **S** his temporary public key with intention to generate a standard DH key k [DH76] for the encryption (and decryption) of the e-goods[10].
3. **S** sends the encrypted goods $E_k(goods)$ to **U**, where E_k is a public, secure against chosen message attack, symmetric cipher algorithm.
4. **U** sends to **S**: $A_1 = g_1^{u_1 s}, A_2 = g_2^s, A, B, (z, a, b, r)$.
5. **S** checks $A = A_1 A_2$, $A \neq 1$, $sig(A, B) = (z, a, b, r)$, and then responses to **U**: $d = H_1(A_1, B_1, A_2, B_2, I_S, date/time, desc, \tau)$ (where I_S is the shop's ID).
6. **U** computes $r_1 = d(u_1 s) + x_1$, $r_2 = ds + x_2$, and then sends r_1, r_2 to **S**.
7. **S** accepts iff $g_1^{r_1} = A_1^d B_1$ and $g_2^{r_2} = A_2^d B_2$, and then sends her share of the DH key k (i.e., her temporary public key) to **U**.

4.2 The Solution

The intuitive solution, as discussed earlier, is to activate some recovery protocol in which the participants re-think the transcript. While this is natural, indeed, for withdrawal and deposit protocols, it is subtle for the payment-delivery protocol (see below). On the other hand, double-withdrawal mentioned above can be blocked using the original *coin tracing* technique in [FTY96, T97] (which is another reason to build upon FOLC). The rationality behind this choice of recovery at which phases comes partially from the fact that in withdrawal **U** has to authenticate himself to establish the ownership of his account, and in deposit the shop's account is embedded in the transcript, whereas the state of a non-atomic purchase transaction (in general) may be very complicated.

As argued in subsection 1.1, **U** and **S** will face a dilemma if the network crashes (alternatively, the customer's PC malfunctions, or there is a denial-of-service attacker) while neither **U** is acknowledged by **S** for his e-coins nor **S** is

[10] k is chosen by only one party, either **U** or **S**, the system is subject to denial-of-service attack if the corresponding recovery protocol is not carefully designed (i.e., without requesting **U** to prove the ownership of the e-coins concerned). However, in this paper the notations "share of the decryption key" and "decryption key" are interchangeable.

acknowledged by **U** for her share of the decryption key. Regardless of whether **S** has received the e-coins or not, if **U** does not lose the state information for the transaction, he can activate simply a recovery transaction. Otherwise, he needs to activate a resolve protocol with **TTP** and **B** to see if his coins has ever been deposited within the negotiated τ. If so, **S** is asked to send **U** the goods (otherwise she will face a legal action); if not, **U** has to refund the e-coins for the sake of anonymity.

However, there are still two problems yet to be addressed. One is that when a claimed customer recovers a purchase transaction, if he wants to change the original share of the negotiated DH key k used for the encryption (therefore decryption) of the e-goods (say, he may claim that he has lost his original share of k), a protocol for "ownership proof" of the e-coins has to be activated. Otherwise, a denial-of-service attacker can snatch the e-goods. Another, is the security of the refund protocol in which the "ownership proof" for the e-coins is also necessary to frustrate the determined denial-of-service adversary. To block the two possible attacks, for the sake of efficiency, a proof for knowledge of Schnorr scheme [S91] can be initiated. For example, a Schnorr proof for knowledge $\log_{g_1} B_1 = x_1$ is competent. In the rest of this paper, this "ownership proof" is used as a primitive without further detailed.

Recovery protocol for *withdrawal atomicity*:

1. **U** proves to **B** that he is the owner of certain account.
2. **U** sends to **B** the transcript $a^{'}, b^{'}, c^{'}$ of the broken transaction (alternatively, if **U** wants to generate new coins as he claimed that the state information has been lost and then he is unable to present it, or **U** claims that he has received no e-coins, or **B**'s signature key has just been replaced, the original coins should be blacklisted using the original *coin tracing* technique of [FTY96] to realize double-withdrawal robustness).
3. Otherwise, after checking that there really exists this session, **B** responses with $r^{'} = c^{'}x + w$ to **U**. If it has not yet debited, it does so now.

A replaced signature key of the bank should be still valid for a period of time in which coins signed by it are acceptable. Within this period the recovery protocol above can also be used to "refresh coins" of users, while merchants are assumed to rush to deposit such coins before they become invalid.

If a payment-delivery protocol is broken, **U** has two choices: either to recover or to resolve it through the following Recovery and Resolve protocols respectively.

Recovery protocol for *payment-delivery atomicity:*

1. **U** sends the same transcript in the broken payment-delivery protocol to **S** (alternatively, if he wants to change the original share of k, the protocol for the proof of ownership of the coins is initiated).

2. After checking the validity, **S** sends her share of k to **U**.

Recovery protocol for *deposit atomicity*:

1. **S** sends **B** the transcript of the not yet acknowledged deposit.
2. **B** checks the existence and the state (i.e., whether or not has been cancelled due to expiration) of the claimed session, and does whatever as this is a normal run of deposit protocol. All coins not yet credited are credited now.

If **U** does not get the share of k from **S** until the invalid date/time τ (no matter he has initiated the Recovery protocol or not), he can now activate the resolve protocol.

Resolve protocol: (performed between **U**, **S**, **B** and **TTP** over an anonymous channel, the **TTP** can be any one chosen from the set of trustees responsible for revocation of anonymity)

1. **U** sends **TTP** the transcript of the broken payment-delivery transaction (if he has lost the state information of the transaction, he can simply present the coins).
2. After checking that the payment has expired (otherwise, **S** may still honestly manage to send her share of k to **U** before τ), **TTP** first asks **U** to prove his ownership of the e-coins using the above mentioned "ownership proof" primitive (as **S** may need to refund). **TTP** asks **B** to see if these coins have ever been deposited. If yes, **TTP** asks **S** to send her share of k to **U** otherwise she will face a legal action; if not, this payment is cancelled and the coins are refunded. (Alternatively, **U** and **B** can activate another instance of withdrawal protocol to exchange these coins for fresh coins blindly.)

Remarks: 1. In withdrawal and deposit protocols we use implicit time while we use explicit one in payment-delivery protocol. The rationality is that sometimes delivery may not be completed immediately. However, it is necessary for **S** to deposit all received coins before their invalid date/time τ.

2. It is interesting to notice that the original payment protocol in [B93, FTY96] used the information *date/time* (it is the τ_0 for our purpose) however, it is not intended to address the problem of atomicity. The synchronous clock used in [B93, FTY96] can also be replaced with the global standard time.

3. The I_S used in the payment is important as there is no measure for **S** to prove that she is the coin's owner.

4. The shop's share should be consistent with the k of $E_k(goods)$, otherwise the customer will not be able to correctly decrypt it. (If she intends to deny the contents/semantics of the goods, *desc* will frustrate her as it may include a cryptographic hash of the content!). Though **S** changing her share will not result in any security or privacy vulnerability, this may need to be avoided unless it is compromised (say, due to a Trojan Horse) because the goods may be large and resuming the transmission from the broken point may become necessary (just like the function supported by most of the current FTP tools). Notice also

that **S**'s temporary public key used in different transactions are different (so the public key may be used to encrypt an actual content key, in case we do not want to change the content's encryption).

4.3 Claims

We show that, while our solution inherits all the properties proved in [FTY96] (therefore, the extensions in [T97, CFT98]), it also is atomic (therefore, fair exchange and conservation of money are assured).

Theorem 1 *The extended fair off-line e-cash scheme satisfies unreusability, unforgeability, unexpandability and (conditional) untraceability.*

The proof is almost the same as [T97]. The rationality is that in random oracle model, the distribution

$$H_1(A_1, B_1, A_2, B_2, I_S, date/time, desc, \tau)$$

used in the payment is identical to the original

$$H_1(A_1, B_1, A_2, B_2, I_S, date/time).$$

Theorem 2 *The above extended scheme based transactions are of withdrawal atomicity, payment-delivery atomicity, and deposit atomicity. The assurance for the conservation of the money is implemented by them (and the recovery and resolve protocols if necessary).*

We only give the proof for *payment-delivery atomicity*, since the rest are quite trivial.

proof: (sketch) When we consider a transaction in which there are two receivers (one for money and the other for e-goods), there are only four cases:

1. Everyone succeeds in receiving the intended items. It means that there is a successful transaction and, of course, *payment-delivery atomicity*.
2. **S** received (i.e., having already been credited to her account[11]) the money and **U** did not receive the share of k from **S**. In this case, **U** can activate the resolve protocol. As long as **S** has deposited the coins, **S** has to send her share of k to **U**, otherwise **S** will face a legal action. (Obviously, the goods description *desc* is useful when **U** has lost all the state information.)
3. **U** received the goods and **S** did not receive the coins. This is impossible since we assume that **S** sends her share of k after seeing the coins and she has a powerful fault tolerant system for reliability.
4. Both **U** and **S** receive nothing. In this case the resolve protocol will assure the customer that his money is conserved.[12]

[11] It is plausible to assume that **S** is bound to deposit the coins she has received.

[12] **S** may not deposit the e-coins as she has not yet sent her share of the decryption key to **U**. In this case, she has e-coins only in a mathematical sense within the deadline τ.

□

It is easy to see that privacy for the customer is preserved throughout the processes.

It is interesting to note that the well known problem of fair exchange with off-line **TTP** (refer to [ASW98]) is not as difficult as may have been imagined in the context of "fair exchange e-coins for e-goods" of the current paper. The rationality behind this is that in this context the bank **B** plays an important role.

5 Atomicity Extension to [JY96, J97] E-Cash Scheme

5.1 The Extended Payment-Delivery Protocol

Due to the limitation of space, we only focus here on the original payment protocol. The problems arising at withdrawal and deposit can be addressed using the ideas similar to the ones in the last section. The difference is that the protocol used to prove the ownership of the coins may be different. A practical proof can be constructed from the concrete implementation of the original cryptographic primitive (see **Appendix 2**) in which (x, y) is the corresponding pair of secret and public keys. A possible solution is a challenge in which value 0 and a random number are included. It should also be noted that this proof is necessary only when the customer wants to change his share of the original key k or refund a coin due to reasons resulted from non-atomicity (though refunding is unnecessary in this system, the customer may really want to do it).

1. **U** and **S** negotiate and agree on the price, the goods description $desc$, and invaliddate/time $\tau = \tau_0 + \Delta$ where τ_0 is the time that the interested transaction begins and Δ is the negotiated lifetime for it. Alternatively, τ is the deadline for **S** to deposit the received e-coins.
2. **S** sends the encrypted goods $E_k(goods)$ to **U**, where k is a DH key as mentioned above, E_k is a public, secure against chosen message attack, symmetric cipher algorithm.
3. **U** sends to **S**: (y, s).
4. **S** checks $s = S_B(S_O(y))$ and sends back challenge c (in which $desc$ and τ are included) [13] to **U**.
5. **U** sends the answer $a = S_x(c)$ to **S**.
6. **S** checks $V_y(a, c) = 1$ and keeps the transcript (y, s, c, a) and then her share of k is sent to **U**.

[13] the challenge sent by **S** is composed of some part of a predetermined form (challenge semantics) and a random string.

5.2 The Solution

Here we only give the recovery protocol for payment-delivery and the resolve protocol since they are very different from the ones in the last section.

Recovery protocol for *payment-delivery atomicity*:

1. **U** sends the transcript in the broken payment-delivery protocol to **S** (alternatively, if he wants to change his original share of k, the protocol for "ownership proof" of the coins is initiated).
2. After checking the validity of the transcript as in the last section, her share of k is sent back to **U**.

Similarly, **U** can activate the following resolve protocol if necessary.

Resolve protocol:

1. **U** sends **TTP** the transcript in payment-delivery transaction (if he has lost the state information of the transaction, he may simply present the coins).
2. After checking that the payment has expired, **TTP** first asks **U** to prove his ownership for the e-coins using the abovementioned primitive "ownership proof" in case he wants to refund. Then **TTP** asks **B** to see if those coins have ever been deposited. If yes, **TTP** asks **S** to send her share of k to **U** otherwise she will face a legal action; if not, this payment is naturally cancelled and **U** may not need to refund.

Remarks: 1. In [JY96], the authors presented a "user complaints" protocol. However, it is used for the user to complain that the coins received are not correctly constructed.

2. We recommend that it is better to incorporate the identity of **S** into the challenge as she can not prove the ownership of the coins while deposit. In this case, even the communication channel in deposit protocol is corrupted, the bad guy is still unable to get the coins via denial-of-service attack.

5.3 Claims

While our solution inherits all the properties proved in [JY96, J97], it is also atomic as we show below (implying, fair exchange and conservation of money). From [JY96, J97] and the fact that the basic protocols do not change, we get:

Theorem 3 *The extended electronic cash scheme satisfies unforgeability, impersonation safety, overspending detection, overspending robustness, traceability, revocability, anonymity, framing-freeness, and refundability.*

Theorem 4 *The above extended scheme based transactions are of withdrawal atomicity, payment-delivery atomicity, and deposit atomicity. The assurance for the conservation of the money is implemented by them (and the recovery and resolve protocols if necessary).*

Again, we only give the proof for *payment-delivery atomicity*, since the rest are easy.

proof: (sketch) When we consider a transaction in which there are two receivers (one for money and the other for e-goods), there are only four cases:

1. Everyone succeeds in receiving the intended items. It means that there is a successful transaction and, of course, *payment-delivery atomicity*.
2. **S** received (i.e., having already been credited to her account) the money and **U** did not receive her share of k from **S**. In this case, **U** can activate the resolve protocol. As long as **S** has deposited the coins, **S** has to send her share of k to **U**, otherwise **S** will face a legal action. (Obviously, the goods description *desc* is useful when **U** has lost all the state information.)
3. **U** received the goods and **S** did not receive the coins. This is impossible as we assume that **S** sends her share of k after seeing the coins and she has a powerful fault tolerant system for reliability.
4. Both **U** and **S** receive nothing. In this case the resolve protocol will assure the customer that his money is conserved.

□

6 Conclusion

We have presented a comprehensive, secure, anonymous, yet efficient solution to conserve the amount of money via atomicity in fair off-line e-cash based e-commerce transactions. The adopted *recovery* approach employs simple protocols and at the same time it assures the involved participants that they can re-think the transactions when things go wrong, thus the atomicity of the transactions. We also take an optimistic approach to *fair exchange* which can be done in only 2-round of communication (which is trivially the lower bound). All of this is done with no additional **TTP** involvement since there have already been ones for the possibly requested revocation of anonymity in the system. Consequently, our solution assures conservation of money and fair exchange through *withdrawal atomicity*, *payment-delivery atomicity*, and *deposit atomicity* whereas the security and anonymity of the original schemes are naturally inherited.

This paper shows the strength of the setting of fair off-line e-cash schemes (FOLC). It not just enables better control of e-cash when things go wrong due to "criminal suspicion" and other "regulatory/legal" issues, but the same parties that take care of this can also assure atomicity and control of conservation of money in case of failure during the transactions. This kind of piggybacking atomicity control over "anonymity revocation" fits nicely since both actions are done by off-line invocation of the same **TTP**s. The obvious conclusion is that fair off-line e-cash schemes ease atomicity in e-cash systems.

Acknowledgments: Many thanks to Markus Jakobsson for very valuable discussions. We also thank the anonymous referees for useful comment. The first author also thanks Feng Bao, Jan Camenisch, David Chaum, Ronald Cramer, and Wenbo Mao for their valuable feedback.

References

[ASW98] N. Asokan, V. Shoup, and M. Waidner, Optimistic Fair Exchange of Digital Signature, Eurocrypt'98

[B93] S. Brands, Untraceable Off-Line Cash in Wallets with Observers, Crypto'93

[BBC94] J. Boly, A. Bosselaers, R. Cramer et al., The ESPRIT Project CAFE: High Security Digital Payment Systems, ESORICS'94

[BDM98] F. Bao, R. Deng, and W. Mao, Efficient and Practical Fair Exchange Protocols with Off-Line TTP, IEEE Security and Privacy, 1998

[BGJY98] M. Bellare, J. Garay, C. Jutla, and M. Yung, *VarietyCash*: A Multi-Purpose Electronic Payment System (Extended Abstract), Usenix Workshop on Electronic Commerce'98

[BGK95] E. F. Brickell, P. Gemmell, and D. Kravitz, Trustee-Based Tracing Extensions to Anonymous Cash and the Making of Anonymous Change, SODA'95

[BN97] P. A. Bernstein and E. Newcomer, Principles of Transaction Processing, Morgan Kaufmann Publishers, Inc., 1997

[BP89] H. Burk and A. Pfitzmann, Digital Payment Systems Enabling Security and Unobserability, Computer & Security, 8/5, 1989, 399-416

[C82] D. Chaum, Blind Signatures for Untraceable Payments, Crypto'82

[CFN88] D. Chaum, A. Fiat, and M. Naor, Untraceable Electronic Cash, Crypto'88

[CFT98] A. Chan, Y. Frankel, and Y. Tsiounis, Easy Come-Easy Go Divisible Cash, Eurocrypt'98

[CHTY96] J. Camp, M. Harkavy, J. D. Tygar, and B. Yee, Anonymous Atomic Transactions, 2nd Usenix on Electronic Commerce, 1996

[CPS96] J. Camenisch, J. Piveteau, and M. Stadler, An Efficient Fair Payment System, ACM CCS'96

[DFTY97] G. Davida, Y. Frankel, Y. Tsiounis, and M. Yung, Anonymity Control in e-cash. In the 1-st Financial Cryptography, LNCS 1318 Springer.

[dST98] A. de Solages and J. Traore, An Efficient Fair off-line electronic cash with extensions to checks and wallets with observers, In the 2-d Financial Cryptography.

[DH76] W. Diffie and M. E. Hellman, New Directions in Cryptography, IEEE Transactions on Information Theory, 1976, 644-654

[F93] N. Ferguson, Extensions of Single-Term Coins, Crypto'93

[FTY96] Y. Frankel, Y. Tsiounis, and M. Yung, Indirect Discourse Proofs: Achieving Fair Off-Line E-Cash, Asiacrypt'96.

[FTY98] Y. Frankel, Y. Tsiounis, and M. Yung, Fair Off-line e-cash made Easy, Asiacrypt'98.

[FY93] M.K. Franklin and M. Yung, Secure and Efficient Off-line Digital Money, ICALP'93 LNCS 700, Springer Verlag. 1993.

[J95] M. Jakobsson, Ripping Coins for a Fair Exchange, Eurocrypt'95

[J97] M. Jakobsson, Privacy vs. Authenticity, PhD thesis, 1997

[JM98] M. Jakobsson and D. M'Raihi, Mix-based Electronic Payments, Workshop on Selected Areas in Cryptography, 1998
[JY96] M. Jakobsson and M. Yung, Revokable and Versatile E-Money, 3rd ACM Computer and Communication Security, 1996
[JY97] M. Jakobsson and M. Yung, Magic Ink Signature, Eurocrypt'97.
[O95] T. Okamoto, An Efficient Divisible Electronic Cash Scheme, Crypto'95
[OO89] T. Okamoto and K. Ohta, Disposable Zero-Knowledge Authentication and Their Applications to Untraceable Electronic Cash, Crypto'89
[OO91] T. Okamoto and K. Ohta, Universal Electronic Cash, Crypto'91
[S91] C. P. Schnorr, Efficient Signature Generation by Smart Cards, J. Cryptology, 4(3), 1991, 161-174
[SPC95] M. Stadler, J. M. Piveteau, and J. Canmenisch, Fair Blind Signature, Eurocrypt'95
[ST95] M. Sirbu and J. D. Tygar, NetBill: An Internet Commerce System, IEEE Compcon'95
[T96] J. D. Tygar, Atomicity in Electronic Commerce, ACM Symposium on Principles of Distributed Computing, 1996
[T97] Y. S. Tsiounis, Efficient Electronic Cash: New Notions and Techniques, PhD thesis, 1997
[vSN92] B. von Solms and D. Naccache, On Blind Signatures and Perfect Crimes, Computers & Security, 11(6), 1992, 581-583
[W98] M. Waidner, Open Issues in Secure Electronic Commerce, 1998

Appendix 1 The Protocols of [FTY96]

All the notations used here are the same as in [FTY96]. Briefly, G_q is a prime order q subgroup of Z_p where p is also a prime. g, g_1 and g_2 are generators of G_q. H, H_0, H_1, ..., are hash functions of collision intractable. **B** chooses her secret $x \in_R Z_q$. **B** publishes p, q, g, g_1, g_2, (H, H_0, H_1, ...,) and her public keys $h = g^x$, $h_1 = g_1^x$, $h_2 = g_2^x$.

When **U** setups an account with **B**, **B** associates **U** with $I = g_1^{u_1}$ where the secret $u_1 \in_R Z_q$ is chosen by **U** such that $g_1^{u_1} g_2 \neq 1$. It is needed for **U** to prove to **B** that he knows the represent of I with respect to g_1. **U** computes $z' = h_1^{u_1} h_2 = (Ig_2)^x$.

Withdrawal: **U** gets a coin $(A, B, (z, a, b, r))$ using blind signature.

1. **U** proves to **B** that he is the owner of some account.
2. **B** chooses randomly and uniformly $w \in_R Z_q$, compute $a' = g^w$, $b' = (Ig_2)^w$, and then sends to **U** a' and b'.
3. **U** selects $s, x_1, x_2, u, v \in_R Z_q$; computes $A = (Ig_2)^s$, $z = (z')^s$, $B_1 = g_1^{x_1}$, $B_2 = g_2^{x_2}$, $B = [B_1, B_2]$, $a = (a')^u g^v$, $b = (b')^{su} A^v$, $c = H(A, B, z, a, b)$, $c' = c/u$; and then sends c' to **B**.
4. **B** responses $r' = c'x + w$ to **U**.
5. **U** calculates $r = r'u + v$, checks $g^{r'} = h^{c'} a'$ and $(Ig_2)^{r'} = (z')^{c'} b'$. **U** obtains a coin $(A, B, Sig_{Bank}(A, B))$ where $Sig_{Bank}(A, B) = (z, a, b, r)$, such that $g^r = h^{H(A,B,z,a,b)}$ and $A^r = z^{H(A,B,z,a,b)} b$.

Payment:

1. **U** sends to **S**: $A_1 = g_1^{u_1 s}$, $A_2 = g_2^s$, $A, B, (z, a, b, r)$.
2. **S** checks $A = A_1 A_2$, $A \neq 1$, $sig(A, B) = (z, a, b, r)$. If all these succeed, **S** sends **U**: $d = H_1(A_1, B_1, A_2, B_2, I_S, date/time)$.
3. **U** computes $r_1 = d(u_1 s) + x_1$, $r_2 = ds + x_2$, and then sends r_1, r_2 to **S**.
4. S accepts iff $g_1^{r_1} = A_1^d B_1$ and $g_2^{r_2} = A_2^d B_2$.

Deposit:

1. **S** sends **B** the transcript of payment A_1^i, B_1^i, A_2^i, B_2^i, (z^i, a^i, b^i, r^i), I_S^i, $date/time^i$, d^i, r_1^i, r_2^i for $i = 1, \cdots, n$.
2. **B** checks I_S^i is the identity of the shop, and does whatever **S** has done in payment protocol for $i = 1, \cdots, n$. Finally **B**'s account is credited.

Appendix 2 The Protocols of [JY96, J97]

All the notations used here are as in [JY96, J97]. Briefly, (S, V) is some existentially unforgeable signature scheme. A coin is a pair (y, s) where $s = S_B(S_O(y))$ and y is the public key corresponding to the secret key x.

Withdrawal:

1. **U** runs the key generation algorithm to get (x, y). He proves his identity to **B** potentially in a manner that does not allow an eavesdropper to get any information about his identity.
2. Using the magic ink signature generation scheme, **U**, **B**, and the Ombudsman servers compute an output so that the withdrawer gets a **B/O** signature $s = S_B(S_O(y))$, and **B** (and possibly the **O**) server gets a tag tag linked to the signed message, i.e., such that $Corresponds(tag, coin)$ is satisfied.

Payment*:*

1. **U** sends **S**: (y, s).
2. **S** checks $s = S_B(S_O(y))$ and sends back challenge c to **U**.
3. **U** sends **S** the answer $a = S_x(c)$.
4. **S** checks $V_y(a, c) = 1$ and keeps the transcript (y, s, c, a).

Deposit:

1. **S** forwards to **B** the transcript (y^i, s^i, c^i, a^i) where $i = 1, \cdots, n$.
2. **B** checks that $s^i = S_B(S_O(y^i))$ and $V_{y^i}(a^i, c^i) = 1$, and further verifies that the same transcript has not been deposited before, and then credits the depositor's account.