Fair Electronic Exchange with Invisible Trusted Parties

by

SILVIO MICALI

ABSTRACT

Assume each of two parties has something the other wants. Then, a *Fair* exchange is an electronic protocol guaranteeing that either both parties get what they want, or none of them does. Protocols relying on traditional trusted parties easily guarantee such exchanges, but are inefficient (because a trusted party must be part of every execution) and expensive (because trusted parties want to be paid for each execution).

We put forward a new class of protocols, relying on *invisible* trusted party (a new type of trusted party), and show that they provide Fair exchange more efficiently than before, and at a fraction of the cost. In particular, we provide very attractive protocols for Fair certified e-mail and contract signing.

1 Fair Electronic Exchange

EXCHANGE vs. FAIR EXCHANGE
Assume that Alice has a string "a" and Bob a string "b." Neither party knows the other's string, but will *recognize* it if he sees it. In an *exchange* protocol, it is *desired* (but not guaranteed) that the parties electronically exchange their strings; that is, Alice gets "b" and Bob gets "a." A protocol for doing so is called an Fair *exchange* if (without forcing an exchange against the parties' will) it *guarantees* that Bob gets (i.e., learns) "a" if and only if Alice gets "b." That is, the protocol should have only two possible (good) outcomes:

1. Alice gets "b" and Bob gets "a" (whenever both parties want so)

or

2. Alice gets nothing and Bob gets nothing (if one party wants so)

EXAMPLE 1: Fair Certified E-Mail (Fair CEM)

In a Certified E-Mail (CEM) application, Alice is the sender, Bob the recipient, "a" Alice's message for Bob, and "b" Bob's receipt for Alice, and it is desired that Bob gets the message and Alice gets the receipt.

- A Fair CEM protocol *guarantese* that Bob gets the message if and only if Alice gets the receipt. That is, message and receipt are exchanged "simultaneously" or not at all, *even if one of the parties is malicious and deviates arbitrarily from his prescribed instructions.*

EXAMPLE 2: Fair Electronic Contract Signing

In an Electronic Contract Signing (ECS) application, Alice and Bob are parties who wish to execute a would-be contract C they have already negotiated, "a" is Alice's digital signature of C, "b" is Bob's digital signature of C, and it is desired that each party ends up with the signature of the other.

- A *Fair ECS* protocol *guarantees* that Alice gets Bob's signature if and only if Bob gets Alice's signature. That is, C is executed "simultaneously" or not at all, *even if one of the parties is malicious and deviates arbitrarily from his prescribed instructions.*

Fairness is a crucial aspect of every form of exchange, and Fair electronic exchange is crucial to e-commerce. No real e-commerce may exist without being able to prove" who said what" (i.e., without Fair CEM) or without "simultaneous transactions" (i.e., without Fair ECS). So: *Can we achieve Fair electronic exchange?*

1.1 Fair Electronic Exchange without Trusted Parties

IF BOTH PARTIES WERE HONEST, NO THIRD TRUSTED PARTY WOULD BE NEEDED

Were both Alice and Bob *guaranteed* to be always honest, Fair electronic exchange would be trivially achievable without invoking any external, trusted party. For instance, here are two 2-message protocols for CEM and ECS.

Protocol *Honest CEM*:
- Alice sends her message M to Bob.
- Upon receiving M, Bob returns his own signature of it, $SIG_B(M)$, to Alice as a receipt.

Protocol *Honest ECS*:
- Alice sends her own signature of C, $SIG_A(C)$, to Bob.
- Upon receiving $SIG_A(C)$, Bob returns $SIG_B(C)$ to Alice.

IF ONE PARTY CAN BE DISHONEST, TRUSTED THIRD PARTIES ARE NEEDED

Unfortunately, if Bob were dishonest, Honest CEM would not be an Fair exchange at all: upon learning M, Bob may never return $SIG_B(M)$, thus forever depriving Alice of her receipt. Similarly, Honest ECS is not a Fair ECS protocol at all if Bob is dishonest: upon receiving Alice's digital signature, Bob may not reciprocate, thus seriously damaging Alice. Consider, for example, a malicious execution of Honest ECS in which Alice is a seller, Bob a malicious buyer, and C Alice's offer to sell her house to Bob. Alice sends $SIG_A(C)$ to Bob and Bob never responds. Can she sell her house to someone else? What if Bob returns $SIG_B(C)$ and claims the house as his? Without the participation of a trusted third party, these problems would not disappear by just inserting in C a naive clause such as "void unless counter-signed by Bob in 24 hours." In fact, Bob may claim that he did indeed delivered a signed copy of C to Alice, and that she is pretending to have never received it. (Of course, if an Fair CEM sub-protocol were demanded for this delivery, Bob's claim would not hold. But no such demands are made by Honest ECS!) In sum, Honest CEM and Honest ECS are not Fair if their participants may be malicious, and some naive "fixes" fail to work. Can these 2-party protocols be fixed in some other way? Unfortunately, the answer is no.

No purely-two-party protocol is a Fair exchange if one of the two participants is malicious.

Leaving formalization aside, the intuition supporting the above statement is clear: we wish to go from the (initial) fair state in which neither Alice nor Bob has what he/she wants, to the (desired) fair state in which both have what they want. But information is exchanged non-simultaneously, thus we must pass through an intermediate *unfair* state: either

3. Alice gets "b" first, while Bob has not gotten "a" yet

or

4. Bob gets "a" first, while Alice has not yet gotten "b".

Now, recall that the assumed nature of the secrets is such that each of Alice and Bob can recognize the other's secret upon receiving it. Thus, if malicious, the party who gets first what he/she wants may halt, thereby forcing the exchange to be incomplete. That is, electronic signals may travel at the speed of light, but strategic behavior may dictate delaying or withholding the very sending of these signals, thus enabling a malicious user of a purely two-party protocol to disrupt the Fairness of the exchange. This conclusion can be rephrased as follows.

To withstand malicious behavior, Fair electronic exchange must rely on a third party.

But, if third parties must be used, the *quality* of a Fair exchange crucially depends on the *type* of third party used.

1.2 Fair Electronic Exchange with Visible Trusted Parties

Fair exchange, electronic or not, is possible with the help of a trusted party (TP). For instance, we are all very familiar with paper-based certified mail, which is acceptably Fair, but requires a great amount of help from a special TP: the Post Office. Indeed, a mailman must reach the recipient in person, and obtain a signed receipt when the message is delivered. This process can be mimicked electronically (by using e-mail, digital signatures, and an electronic Post Office) as exemplified by the following 4-message protocol.

Protocol CEM0:

- Alice sends her message M to the Electronic Post Office.
- The electronic Post Office sends M to Bob (making sure that)
- Bob gives his receipt $SIG_B(M)$ to the electronic Post Office
- The electronic Post Office forwards $SIG_B(M)$ to Alice.

A similar TP also enables Fair ECS protocols. Here is a 4-message example.

Protocol ECS0:

- Alice sends her own digital signature of C, $SIG_A(C)$, to the TP.
- Bob sends his own digital signature of C, $SIG_B(M)$, to the TP.
- Upon receiving both $SIG_A(C)$ and $SIG_B(M)$, TP sends $SIG_A(C)$ to Bob and $SIG_B(M)$ to Alice.

Notice that such traditional TPs are *involved in every transaction*, and thus we shall call them *visible* (in order to contrast them with our new type of trusted parties).

DRAWBACKS OF VISIBLE TPs

CEM0, ECS0, and all protocols with visible TPs suffer from 3 main drawbacks:

- ***Cost:*** A transaction with a visible TP is more expensive than a transaction without it.
  Visible TPs are quite expensive to run: they must offer services on a 24-hour basis every day of the year, with bandwith and computing power capable of handling, *in real time*, a potentially enormous traffic. Thus a visible TP has *very high infrastructural costs*.

- ***Congestion:*** A visibleTP creates significant congestions.
  Possibly millions of certified e-messages per day would have to be routed through a visible Post Office, thereby creating a major bottleneck in the network. At the same time, relying on many visible Post Offices is not advisable. In fact, trusting a single network site is very different from trusting many sites!

- ***Liability:*** A visible TP incurs major liabilities.
  For instance, a computer crash or error may cause the visible Post Office of ECM0 to loose messages, receipts, etc., with possibly disastrous consequences for the users, who might try to be compensated in court for their losses. Moreover, it must archive proof that he discharged properly its obligations for every transaction and for a long time. Thus, it must use highly secure and geographically dispersed archiving facilities for safekeeping a large amount of information against hackers and earthquakes alike. These are items with high operating costs. In addition, because in large volumes something goes always wrong, a visible Post Office should carry substantial liability insurance, a noteworthy cost that, ultimately, would have to be borne by the users. A similar argument applies for the visible TP of ECS0.

1.3 Fair Electronic Exchange with Invisible Trusted Parties

Because Fair electronic exchange cannot work without third TPs, and because traditional (i.e., visible) TPs have serious drawbacks, we wish to put forward a new type of TPs. We call our new TPs *invisible*, because they keep in the background and intervene very rarely, if at all, in the execution of an exchange protocol. Yet, invisible TPs guarantee Fair exchange in a way that is both more efficient and way more economical!

HIGH-LEVEL PROPERTIES OF INVISIBLE TPs

In an Fair exchange protocol with an invisible TP, the following properties hold:

- The TP will not participate in any execution in which both Alice and Bob behave honestly (indeed, the TP is not even aware that Alice and Bob are starting an execution);
- The TP needs not store any secret (or message) of either party before or during an execution; AND YET,
- If Alice gets what she wants while Bob does not, Bob accesses the TP providing whatever information he happens to have in his possession, and the TP, without any cooperation from Alice, completes the exchange *exactly as Alice should have done, had she be honest* (and vice versa if Bob gets what he wants while Alice does not).

MAIN ADVANTAGES OF INVISIBLE TPs

- *An invisible TP intervenes very rarely.*
  An invisible TP intervenes (by performing a simple operation) only when cheating occurs, in which case the cheated party gets from the TP what he was entitled to get from the other party. Thus, because nothing is to be gained by cheating, we can assume that cheating will be rare.

- *An invisible TP generates no congestion or bottlenecks.*
  Because an invisible TP intervenes only very rarely, the users will complete most exchanges by directly communicating with each other, bypassing the TP altogether.

- *An invisible TP generates minimal expenses and minimal liabilities.*
  Because the invisible TP intervenes very rarely, it needs to handle and be liable for very few messages, even if the entire country sends every e-mail message certified! .Thus its infrastructure and liability costs are very modest.

THE SUBSCRIPTION BUSINESS MODEL

Because it intervenes so rarely, an invisible TP is better off charging the users on a *subscription basis*. By paying a moderate yearly fee, a user has the right to receive the TP's help (at cost) only when he really needs it. On the other hand, if a user has not paid such a subscription fee, then the TP may refuse to help him when he is in trouble (or demand a much higher fee). In essence, therefore, such an invisible TP acts as an insurance company, but with a major difference: while an insurance company must, every now and then, pay considerable sums, an invisible TP must, every now and then, help a user in trouble by executing a very simple algorithm, and being paid too! Thus, such a subscription is a win-win proposition. In fact, (1) by paying a *small* yearly fee, the users can perform *arbitrarily many* Fair exchanges, while (2) the invisible TP pockets a yearly fee from all users for doing essentially no work (and is paid extra for those rare transactions in which its help is invoked).

In essence, an invisible TP is selling its *potential* intervention, and in the rare case of an *actual* intervention, it performs a simple operation and is separately paid its strict costs. In sum, therefore,

- In an Fair exchange with a *visible* TP, the users essentially agree to pay, *a priori*, a transaction fee for each transaction (even when all parties are honest, out of the fear that cheating may occur);
- In an Fair exchange with an *invisible* TP, the users agree to pay a modest yearly fee a priori, plus a transaction fee, *a posteriori*, only for those rare transactions in which cheating, though useless, truly occurs (in which case they are more than happy to pay that transaction fee).

Humanity has relied on TPs from time immemorial for facilitating commerce and other orderly transactions. And from time immemorial TPs have been charging high fees because of their deep involvement in every single transaction. Invisible TPs are a new tool that breaks ranks with a long tradition, and brings about potentially enormous opportunities. In an "invisible setting", it is in fact possible to price the TP's services so that both the users and the TP are better off than in a "visible setting."

3. Fair Certified E-Mail with Invisible Trusted Parties

PRELIMINARIES

Each user in the system has a unique identifier. For variation of discourse, we may call Alice the *sender*, Bob the *recipient*, and the invisible trusted party the *Post Office*. We denote Alice's identifier by A, Bob's by B, and the Post Office's by PO. Alice, Bob and the Post Office can all sign messages: again, X's signature of a message M is denoted by $SIG_X(M)$, and we assume, for convenience, that M is always retrievable from $SIG_X(M)$. Alice, Bob and the Post Office can also encrypt messages by means of a public-key encryption algorithm. (Each thus has a public encryption key and a corresponding secret decryption key.) By $E_A(M)$, $E_B(M)$, and $E_{PO}(M)$, we denote, respectively, the encryption of a message M with the public key of Alice, Bob, and the Post Office.[1] That is, everyone can encrypt a message M with the public key of party X, but only X understand $E_X(M)$.

Let us present our Fair CEM protocol assuming that Alice is not concerned about the privacy of her message M to Bob, provided that Fairness is guaranteed. That is, Bob should learn M if and only if Alice gets a receipt for it, but it does not matter whether the Post Office else learns M too. (Total privacy could be guaranteed by assuming that M has been encrypted in Bob's public key, so that only he can understand it.)

The protocol below envisages 5 possible steps of communication: A1 and A2 for Alice, Bl and B2 for Bob, and POl for the Post Office. However, at most 3 steps Al, Bl, and A2 will ever need to be executed if Alice and Bob are both honest. Steps B2 and POl will be executed only if Alice fails to execute properly Step A2.

Protocol Fair CEM

- Al: Given her message M, Alice computes $Z = E_{PO}(A,B,M)$, the encryption in the Post Office's public key of (1) Alice's identifier (signifying that she is the sender), (2) Bob's identifier (signifying that he is the recipient), and (3) the message M. She then sends Z to Bob.

- Bl: Upon receiving Z from Alice, Bob digitally signs it and sends $SIG_B(Z)$ to Alice as a receipt.

- A2: If Alice receives the properly signed receipt from Bob, she sends M in the clear to Bob.

- B2: *If* Bob receives a string M such that encrypting A,B,M with the Post Office's public key yields the value *Z* received in Step B1, then he halts: the CEM protocol has been successfully completed. *Else,* he sends the originally received value Z and $SIG_B(Z)$ to the Post Office, indicating that Alice is the sender and he is the recipient.

- POl: If Bob's signature of Z is correct, the Post Office decrypts Z with its secret key. If the result is a triplet consisting of A, B, and a string M, then the Post Office (i) sends Alice the value Z signed by Bob as the receipt, and (ii) sends M to Bob.

REMARKS

Given Bob's signature $SIG_B(Z)$, Alice can, by releasing M, precisely prove to anyone the content of the message for which $SIG_B(Z)$ is a receipt. Indeed, anyone can use the Post Office's public key so as to compute $E_{PO}(A, B,M)$, and then verify that indeed $E_{PO}(A, B,M) = Z$.

By definition, the message M for which $SIG_B(Z)$ is a receipt is *the decryption of Z relative to the Post Office's key*, and may be nonsensical. Indeed, nothing prevents Alice from sending Bob a garbled message. However, she can only get a receipt for this same garbled message.

[1]For simplicity, we assume that messages are encrypted directly with a public-key algorithm. But, according to standard practice, we could first encrypt a message M conventionally with some key K, and then encrypt K with a public-key algorithm.

WHY FAIR CEM WORKS

Informally, if the encryption algorithm is properly secure, when receiving the value $Z = E_{PO}(A,B,M)$ from Alice, Bob will have difficulties in computing M from Z without the Post Office 's secret key. Thus, if he halted now, Alice would not get her receipt, but Bob would not get the message either.

Assume now that Bob signs Z and sends it to Alice. Because this gives Alice a valid receipt from Bob for her message M, for the exchange of message and receipt to be Fair, we must now show that Bob easily obtains M. This is certainly true if Alice sends him M in Step A2. Assume therefore that Alice does not send him M. Then, Bob presents Z signed by him to the Post Office, essentially asking it to retrieve M from Z on his behalf. The Post Office complies with his request. In doing so however, the Post Office also sends Alice, as a receipt, Z signed by Bob. (It does so to prevent one last possibility: that Bob, upon receiving Z from Alice in Step Al, rather than sending her a receipt in Step Bl, Bob may go *directly* to the Post Office in order to have *M* "extracted" from Z.)

Summarizing, if Alice sends an encrypted message to Bob, and Bob does not send Alice his signature of it, or access the Post Office, then there is nothing to worry about. In fact, Alice never gets the receipt and Bob never learns M. Else, Alice is guaranteed to get her receipt for M either from Bob directly or via the PO. On the other hand, if Bob sends Alice the receipt, then he is guaranteed that he will obtain the message, either via Alice or via the Post Office.

(Note: a formal proof of CEM1 is derivable from SIG_B being a signature scheme non-existentially forgeable by an adaptive chosen message attack and E_{PO} being a public key encryption scheme non-malleable/secure against adaptive chosen cipher-text attack.)

Our Fair CEM protocol can easily incorporate a cut-off time, so as to guarantee that either both parties get what they want by the chosen cut-off time, or neither party will ever get what he wants (unless they try again).

4. Beyond Certified E-Mail

- *Electronic Checks*. Very often we need receipts for something other than traditional messages. For instance, it would be nice that the recipient gets our electronic check to him if and only if we get a corresponding receipt that he got it!
- *Information Requirements*. Often some agents have an obligation to inform others (e.g., about coming deadlines to exercise certain options), and they would like to do so in a manner that prove their having complied with their obligations. Our Fair CEM protocol is ideally suited for such applications, and much cheaper and faster than traditional approaches.
- *Electronic Sales*. Goods that can be put in digital form (such as computer programs, music, etc.) continue to be distributed via physical stores, who easily retain 50% of the proceeds. If they could be directly transferred from the producer to the consumer, the producer could retain a much higher percentage of the proceeds, and the consumer could receive a big discount too. But things are not so simple. Assume that, over the Internet, a consumer C asks a producer P to purchase a good G costing D dollars. P insists that U provide first his credit-card number; U complies; and then P ships him G (preferably encrypted in a key of U). However, a month later, when seeinga D-dollar entry about G on his statement, a malicious U may call the credit card company and claim that he never received G (or even that he never asked for G). At the point the credit card contacts P, who may offer to ship G again to C. But at this point C may say that he had a need for G a month ago (e.g., he needed music for his birthday party) while getting G now is of no use to him! If instead P provided C with G via our Fair CEM protocol (i.e., if our protocol were executed with G as the message), then C could not claim that he never got G: P would have a receipt, and one of the highest quality and precision, that C got G!

- Fair Exchange protocols exist for essentially all possible transactions. In particular, we have designed very efficient Fair ECS protocols.

5. Intellectual Property

My technology is protected by several domestic and foreign patents. Valuable forms of Fair Electronic Exchange with visible TPs are also patented protected. The issued patents include:

1. U.S. Patent No. 5,666,420: *Simultaneous Electronic Transactions*
2. U.S. Patent No. 6,134,326: *Simultaneous Electronic Transactions*
3. U.S. Patent No. 5,553,145: *Simultaneous Electronic Transactions with Visible Trusted Parties*
4. U.S. Patent No. 5,629,982: *Simultaneous Electronic Transactions with Visible Trusted Parties*
5. U.S. Patent No. 6,137,884: *Simultaneous Electronic Transactions with Visible Trusted Parties*
6. U.S. Patent No. 6,141,750: *Simultaneous Electronic Transactions with Subscriber Verification*
7. U.S. Patent No. 5,615,269: *Ideal Electronic Negotiations*
8. Canadian Patent No. 2,215,908: *Simultaneous Electronic Transactions*
9. EPO Patent No. 96910516.2: *Simultaneous Electronic Transaction;* nationalized in the United Kingdom
10. EPO Patent No. 96910516.2: *Simultaneous Electronic Transaction;* nationalized in Italy