

Exclusions and related trust relationships in multi-party fair exchange protocols

Nicolás González-Deleito *, Olivier Markowitch

Département d'Informatique, Université Libre de Bruxelles, Boulevard du Triomphe – CP212, 1050 Bruxelles, Belgium

Received 21 January 2005; received in revised form 18 November 2005; accepted 29 August 2006

Available online 20 December 2006

Abstract

Some electronic commerce transactions are inherently performed between more than two parties. In this context, it is thus important to determine whether the underlying fair exchange protocols allowing the secure implementation of such transactions enable participants to exclude other entities from a protocol execution. This is an important point that has not been sufficiently addressed when analysing such kind of protocols, and that may be crucial for the successful accomplishment of multi-party electronic transactions. In this paper we define the properties related to exchange protocols and exclusions, study exclusion scenarios on two well-known multi-party fair exchange protocols and point out the implications that exclusions may have on the trust relationships between participants, and, more generally, on electronic commerce. Two new protocols more robust than existing multi-party fair exchange protocols are therefore proposed.

© 2007 Elsevier B.V. All rights reserved.

Keywords: Electronic commerce; Secure multi-party protocols; Fair exchange

1. Introduction

The important growth of open networks such as the Internet has led to the study of related security problems. Achieving the exchange of electronic information (as in contract signing, certified email, etc.) is one of these security challenges. An exchange protocol allows therefore two or more parties to exchange electronic information. Informally, if the exchange is realised in such a way that, at the end of the protocol, either any honest participant has received all the expected items corresponding to the items he has provided or no participant has received anything, then the exchange protocol is said to be *fair*.

A trusted third party (TTP) is often used in order to help the participants to successfully realise the exchange. Depending on its level of involvement in a protocol, a TTP can be said to be *online* or *offline*. Online trusted third parties are involved in each instance of a protocol. On the

contrary, an offline TTP is used when the participants in a protocol are assumed to be honest enough not to need external help in order to achieve fairness. In this case, the TTP will be involved only if a problem emerges. Since it is assumed that problems occur rarely, such protocols are called *optimistic*.

Many fully electronic commerce transactions are based on an underlying fair exchange protocol (e.g. electronic payments, electronic barter, etc.). When more than two parties take part in an exchange protocol, the protocol is said to be *multi-party*.

1.1. Applications to electronic commerce

Protocols achieving exchanges of information have applications to electronic commerce, as they provide a general framework for the implementation of many types of electronic commerce scenarios. Indeed, these protocols allow the realisation of secure exchanges between parties. By *secure exchange* we mean that, at the end of the exchange, no party has to be in an advantageous position compared

* Corresponding author. Tel.: + 32 2 650 55 89; fax: + 32 2 650 56 09.
E-mail address: ngonzale@ulb.ac.be (N. González-Deleito).

to other parties. This kind of secure exchanges often appears in electronic commerce. For example (1) realising an electronic purchase needs the secure exchange of a payment for an electronic good or for a legal promise of delivery; (2) spending electronic money consists in the secure exchange of electronic money for an electronic good, a promise of delivery or just a receipt; (3) signing a digital contract consists in a secure exchange of the digital signature of each signer on the given contract; and (4) realising an electronic barter requires a secure exchange of digital goods of similar value. Secure exchange protocols ensure that all the participants having provided an information (such as a payment, an electronic good, etc.) receive, in return, the information that they expect. Therefore, a protocol respecting this property is called a *fair exchange protocol*.

Fair exchange protocols involving two parties have been extensively studied and several solutions have been proposed in the online [8,15,16] as in the offline [3,4,6,14,17] case. Generalised protocols to more than two parties have also been proposed [2,5,7,9–11]. These multi-party protocols can serve as a framework for implementing everyday life transactions involving several entities (e.g. a merchant, a customer, their respective banks, and even sometimes a notary) in an electronic commerce context.

More precisely, we can consider a common and generic scenario with four parties describing a ring. Let one of these parties be a customer who wants to purchase an electronic item offered by a provider; the payment is realised through the customer's and the provider's banks. This is how each participant views the exchange:

- the provider supplies the expected electronic item to the customer in exchange for having his bank crediting his account;
- the customer sends a payment authorisation to his bank in exchange for the desired electronic information offered by the provider;
- the customer's bank carries out the payment to the provider's bank in exchange for the payment authorisation sent by the customer;
- and, finally, the provider's bank credits the provider's account in exchange for the payment carried out by the customer's bank.

Another scenario is multi-party barter. In such a setting, people interested in getting rid of an electronic item they no longer use would inform a community of other people willing to perform a barter that they are ready to exchange that item for another kind of item. When a match is found between what two or more people offer and expect to receive in return, then an exchange can take place. For example, a person *A* (resp. *B* or *C*) can thus provide in this framework an item to a person *B* (resp. *C* or *A*) in exchange for another item provided by a person *C* (resp. *A* or *B*). One or more rings can therefore be built.

Many proposed multi-party fair exchange protocols [5,7,9,10] allow to perform an exchange over a ring topol-

ogy, as in the examples described above. In other words, each participant P_i offers to participant P_{i+1} an item m_i in exchange for an item m_{i-1} offered by participant P_{i-1} . Of course, all subscripts have to be read modulo n , where n is the number of participants in the exchange (we will omit this hereafter). On the other hand, in [2] a generic optimistic protocol with a general topology is described. However, during this protocol a participant may receive an affidavit from the TTP instead of the expected item. The protocol achieves then what it is called *weak fairness*. This kind of fairness may of course be unacceptable from a practical point of view.

In multi-party fair exchange protocols it is crucial, in order to keep the electronic commerce framework consistent, to prevent participants in a protocol from excluding other participants from the exchange. This is an important point that has not been sufficiently studied when analysing such kind of protocols, and that may be essential for the successful accomplishment of multi-party electronic transactions.

1.2. Contributions

This paper presents an in-depth study of multi-party fair exchange protocols and highlights two trust relationships that exist between participants in existing protocols: a participant having to trust a set of other parties (1) not to be excluded from a protocol execution and (2) not to exclude another group of participants without his consent.

As we will see, the first of these trust relationships may lead to a loss of fairness for the excluded participant, who loses his item without receiving the expected item in return. Its occurrence may thus prevent a protocol from being used in order to implement a practical electronic commerce scenario. The second trust relationship causes excluded participants to consider honest participants not having been excluded as dishonest entities. This may induce these excluded participants to be reluctant to perform other commercial transactions with these honest participants in the future, which will have a negative and improper impact on these honest participants' reputation.

Since these trust relationships may harm the commercial success of an electronic transaction, we propose new multi-party fair exchange protocols that reduce or, if possible, suppress these trust relationships.

Informally, we may say that an entity is excluded from a protocol execution when he is ruled out from it without his consent. In this paper we show exclusion scenarios that may take place during existing multi-party fair exchange protocols, and we highlight their resulting trust relationships between the participants involved in the protocol. We stress that these specific trust relationships do not concern the TTP possibly used in order to successfully complete the protocol, but the main entities.

This paper extends the results of two previous works [9,10]. We define the notions related with exclusion scenarios, like those of an excluded participant, a passive conspirator

excluding some other participants without his consent and a protocol not allowing participants to be excluded. We analyse two well-known protocols with an online and an offline TTP, respectively. Two attacks are described against the first protocol, showing that it is not fair and that excluded participants and passive conspirators may exist. Concerning the second protocol, when discussing about a known attack allowing to exclude participants, we show that the protocol leads to passive conspirators.

In order to avoid the flaws and weaknesses found on the two studied protocols, we present two new multi-party fair exchange protocols, with an online and an offline TTP, respectively. We show that the first protocol does not allow participants to be excluded and that the second one resists against passive conspiracies.

The remaining of this paper is organised as follows. In the next section we define exchange protocols, as well as the fundamental properties that these protocols must respect and the properties related to exclusion scenarios in a multi-party setting. Sections 3 and 4 are devoted, on the first hand, to multi-party fair exchange protocols with an online TTP. In Section 3 we describe the protocol proposed by Franklin and Tsudik [7] and present our attacks on this protocol. A fixed protocol, in which participants cannot be excluded from the exchange, is described in Section 4. On the other hand, Sections 5 and 6 deal with multi-party fair exchange protocols with an offline TTP. In Section 5 we describe the protocol proposed by Bao et al. [5] and discuss about the trust relationships that it creates between the participants in the protocol. We propose a variant protocol in Section 6 in which some of these trust relationships do no longer exist. Finally, Section 7 concludes this work.

2. Definitions and notations

In order to define what we mean by an excluded participant, a passive conspirator or a protocol not allowing participants to be excluded, we will extend the model defined by Markowitch [12,13] to analyse exchange protocols. We briefly recall this model before augmenting it.

Markowitch call a communicating entity an *actor*, and model it by means of a probabilistic interactive Turing machine. An actor communicates with other actors by sending information to one of its write-only communication tapes or by receiving information from one of its read-only communication tapes. These information travel through the communication tapes of another probabilistic interactive Turing machine representing the communication channel connecting two actors. Depending on the quality of service, three different types of communication channels are defined. The first one models a channel where messages inserted into it are delivered within a known, finite and constant amount of time.

Definition 1 [12,13]. A channel is said to be *operational* if and only if (1) whenever a symbol is written on the write-

only communication tape, it is the last non-blank symbol read on the read-only communication tape; (2) the head corresponding to the read-only communication tape does not move towards the left and achieves a movement towards the right when a non-blank symbol is written on the write-only communication tape; and (3) the number of transitions realised right after having read a non-blank symbol on the read-only communication tape up to the writing of the very same symbol on the write-only communication tape is bounded by a known finite constant.

The second type models a communication channel which delivers data after a finite, but unknown, amount of time. It can be straightforwardly defined from the previous definition. Finally, unreliable channels are defined as follows.

Definition 2 [12,13]. If a communication channel is neither operational nor resilient, it is said to be *unreliable*.

An actor is therefore defined [12,13] as a probabilistic interactive Turing machine connected, by means of its communication tapes, to one or several communication channels.

Although not explicitly said in the literature, a protocol allowing an exchange of items is composed of a preliminary *setup phase*, followed by an *exchange phase*. Except for non-repudiation and certified email protocols, during the setup phase, actors willing to participate in a given exchange protocol execution agree on the set of actors who will take part in this protocol execution, on the items to be exchanged and on how these items will be exchanged during the exchange phase. Upon completion of that setup phase, the exchange phase is performed by means of a fair exchange protocol. Markowitch call each of those items to be exchanged a *target information* [12,13].

As usual, an actor is said to be *honest* if he only performs the steps of the protocol.

Definition 3 [12,13]. An *exchange protocol* in which n actors are involved is a protocol during which m ($\leq n$) of the involved actors, if they are honest, exchange target information. The exchange is such that, during the protocol, each of those m actors receives one or several of these target information and sends one or several related target information to one or several of the $m - 1$ other actors (no actor sends information to himself, nor receives information from himself). The $n - m$ actors not concerned by the target information take part in the protocol as middlemen. During the exchange protocol, non-target information can also be transmitted, and target information may be encoded segmented and/or grouped in the messages effectively transmitted.

Definition 4 [12,13]. The m actors of an exchange protocol exchanging target information are called *target actors*. The $n - m$ other actors are called *participating actors*.

Definition 5 [12,13]. The exchanges of target information having to be performed during an exchange protocol are described by an *exchange graph* π , in which vertices represent the target actors and directed edges represent the sendings of target information to achieve. These arcs are labelled by the non-empty sets of target information sent by the origin to the destination of the arc, each target information being represented by a dedicated variable. An exchange graph π will always be strongly connected.

After a successful execution of an exchange protocol between honest actors, all the exchanges described by π are achieved once and only once. An actor may occur in many exchanges simultaneously; hence, we shall also consider exchange systems described by many individual exchange graphs. However, we shall assume that the target information are uniquely identified, so that each one may be attached to one and only one of the composing exchanges.

Definition 6 [12,13]. The exchanges of target information having to be performed in an exchange system are described by a *set Π of exchange graphs* π_i , $1 \leq i \leq |\Pi|$. These exchange graphs are such that no variable occurs twice around a target actor and no effective target information corresponds to more than one variable around him. The union of the different exchange graphs π_i of Π is strongly connected.

Exchange graphs do not specify the manner nor the order in which the exchanges must be realised during the protocol, but correspond to a formalisation of what must be exchanged at the end of the protocol. Several different protocols (systems) can correspond to a same (set of) exchange graph(s).

The fairness property is defined on the basis of *completion points*.

Definition 7 [12,13]. In an exchange system, a *strong completion point for an honest target actor A* corresponds to a step of the protocol after which A is in one of the following *actor's states*:

1. For at least one exchange graph of Π , A has received all her expected target information. Moreover, for each of these exchange graphs in which A has received all her expected target information,
 - all the target actors that must receive some target information sent by A received these target information, or are able to receive them regardless of A 's actions;
 - or all the honest target actors having provided A with her expected target information also received their expected target information (belonging to this same exchange graph), or are able to receive them regardless of A 's actions.

For all the remaining exchange graphs of Π in which A is involved, no target information having to be sent by A has been received by the corresponding target

actors, nor will be received in the future if A does not proceed.

2. A did not receive any target information for which she is the intended recipient and no target information having to be sent by A has been received by the corresponding target actors, nor will be received in the future if A does not proceed.

We have now all the necessary elements in order to define the different possible flavors of fairness.

Definition 8 [12,13]. An exchange system is *strictly fair* when all the honest target actors are always able to reach one of their strong completion points, and these completion points are such that all the honest target actors are in the same actor's state. The behaviour (honest or not) of the participating actors and the quality (operational, resilient or unreliable) of the communication channels must be specified.

This definition models the usual definition of fairness, also known as *all-or-nothing fairness*. The following definition is slightly weaker, but it is useful in some optimistic multi-party protocols [5,9].

Definition 9 [12,13]. An exchange system is *strongly fair* when all the honest target actors are always able to reach one of their strong completion points. The behaviour (honest or not) of the participating actors and the quality (operational, resilient or unreliable) of the communication channels must be specified.

For an exchange protocol to be secure, in addition to one of the previously defined flavors of fairness, the *viability* (or *effectiveness*) and the *timeliness* properties also have to be respected. The former means that if all the target actors are willing to perform the protocol and if no problem occurs, then the exchange is always achieved. The latter guarantees that the protocol will finish, for all the honest target actors, in a finite amount of time, the exchange having taken place or not.

We are now able to extend this model, in the framework of multi-party exchange protocols, in order to deal with exclusions of target actors.

Definition 10. An honest target actor A of an exchange system Π is said to be *excluded* from a protocol execution either if she is not able to reach one of her strong completion points, or if she has reached one of her strong completion points in the second actor's state and at least one honest target actor involved in Π has reached one of his strong completion points in the first actor's state.

As we will see, in optimistic protocols it seems impossible to avoid exclusions. Therefore, we think that it is important in this context to give possible excluded target actors the opportunity of proving that they have been excluded. From a practical point of view, the following definition may thus be useful to enable target actors to be compensated for having been excluded.

Definition 11. An honest target actor is said to be *weakly excluded* from a protocol execution when he can prove to a particular actor (a judge), by means of another protocol, that he has been excluded from this protocol execution.

We will call an honest target actor who does not do anything in order to prevent possible excluded target actors from seeing him as a dishonest target actor, a passive conspirator.

Definition 12. A protocol execution is said *with passive conspirators* associated with a coalition of dishonest target actors excluding some other target actors from the protocol execution, if at least one honest target actor not being excluded (we call her the *passive conspirator A*) does not send to at least one of the possible excluded target actors (we call him *B*) a specific non-target information. This non-target information is such that it allows to notify *B* that if the protocol execution respects the exchange graphs of Π , then *B* should receive the target information of these exchange graphs for which he is the intended recipient.

In order not to be a possible passive conspirator, this definition only requires that *A* sends a non-target information to all the remaining target actors notifying them that if the exchange protocol execution respects the exchange graphs of Π , then the latter should receive their expected target information. It is therefore possible that this non-target information is not received by some target actors due to a poor communication channel quality. But thanks to the non-target information, possible excluded actors will realise that they are excluded if they do not receive anything else, and, depending on the protocol, they will be able to become weakly excluded.

Definition 13. A multi-party exchange protocol is *strongly exclusion-free* if no honest target actor can ever be excluded.

Note that this definition means that all the target actors are always able to reach one of their strong completion points, and that these completion points are such that all the honest target actors are in the same actor's state. This corresponds, word for word, to the definition of a strictly fair protocol. We can therefore conclude that a multi-party exchange protocol is strongly exclusion-free if and only if it is strictly fair.

We think that by looking for potential exclusion scenarios it is possible to analyse more deeply whether a multi-party protocol really respects the fairness property or not. This new definition can thus be of interest in this context.

Definition 14. A multi-party exchange protocol is *weakly exclusion-free* if either at least one honest target actor is weakly excluded and the remaining honest target actors have all reached one of their strong completion points in the first actor's state, or all the honest target actors reach a strong completion point such that all those target actors are in the same actor's state.

We point out that in a protocol not providing strong exclusion-freeness any honest target actor has to trust the remaining target actors not to exclude him (even weakly) from the protocol execution.

Through the remaining of this paper we will use the following notations:

- $X \rightarrow Y: m$ denotes actor *X* sending a message *m* to actor *Y*;
- $X \Rightarrow \mathcal{Y}: m$ denotes actor *X* multicasting a message *m* to the set of actors \mathcal{Y} (a one-to-many communication);
- $X \Rightarrow: m$ denotes actor *X* broadcasting a message *m* (a one-to-any communication);
- $X \leftarrow Y: m$ denotes actor *X* retrieving a message *m* from a read-only public directory managed by actor *Y*;
- $E_X(p)$ is the result of applying an asymmetric encryption algorithm *E* to the plaintext *p* with the actor *X*'s public key;
- $D_X(c)$ is the result of applying an asymmetric decryption algorithm *D* to the ciphertext *c* with *X*'s secret key;
- $S_X(i)$ denotes the digital signature of actor *X* on the information *i*;
- $(i, \dots, j, S_X(\star))$ is a shortcut for $(i, \dots, j, S_X(i, \dots, j))$;
- f_x is a publicly known flag indicating the purpose of a message in a given protocol, where *x* identifies the corresponding message in that protocol;
- *label* is an information identifying a protocol run (it will be precisely defined for each protocol).

3. Franklin and Tsudik's fair exchange protocol

We describe here the multi-party fair exchange protocol with an online trusted third party proposed by Franklin and Tsudik [7]. The protocol assumes that the exchange is cyclic, i.e. a target actor $P_i, i \in [1, n]$, sends his target information m_i to P_{i+1} in exchange for P_{i-1} 's target information m_{i-1} ; P_n sends his target information to P_1 .

In their protocol, the authors assume the presence of a third party which is *semi-trusted*. Such a third party is trusted to ensure the fairness during a protocol run, but as long as all the actors involved in the protocol remain honest the semi-trusted third party will not succeed in learning the exchanged target information. The authors also consider that all the exchanged messages are private and authentic.

The protocol is based on a homomorphic one-way function *f* and a *n*-variable function F_n such that

$$F_n(x_1, f(x_2), \dots, f(x_n)) = f(x_1 \cdot x_2 \cdots x_n).$$

The authors proposed to use $f(y) = y^2 \bmod N$ and $F_n(y_1, y_2, \dots, y_n) = y_1^2 \cdot y_2 \cdots y_n \bmod N$, where *N* is the product of two large distinct primes.

At the end of the setup phase:

- each target actor knows (at least) the identity of the previous and the next target actor in the ring;

- the target actors have agreed on the identity of the TTP that will be contacted during the protocol execution and on the instance of the functions f and F that will be used;
- and the description of the target information to be exchanged, $f(m_i) \forall i \in [1, n]$, are made public.

3.1. The protocol

Each target actor P_i begins the protocol by randomly choosing a value R_i and sending it to P_{i+1} .

Upon receiving R_{i-1} , each target actor P_i computes $C_i = m_i \cdot R_{i-1}^{-1}$ and

$$A_i = F_n(m_i, f(m_1), \dots, f(m_{i-1}), f(m_{i+1}), \dots, f(m_n)),$$

and sends them to the TTP along with $f(R_i)$.

The TTP compares the received A_i and tests if they are all equal. We point out that Franklin and Tsudik do not state from which moment in time the TTP should perform this test. For example, a deadline t after which the TTP can no longer be contacted could have been defined. It would also have been possible to assume that the TTP waits to receive n messages, or even that the TTP knows the identity of all the target actors for each protocol execution. However, as we will show below, none of these classical solutions, even when used together, ensure the fairness property.

So, if the above test succeeds, Franklin and Tsudik ask the TTP to compute $C = C_1 \cdots C_n$ and $F_{n+1}(C, f(R_1), \dots, f(R_n))$. The latter value should be equal to $f(m_1 \cdots m_n)$; the TTP verifies thus whether one A_i is equal to this value. If these two tests succeed, the TTP broadcasts $\mathcal{C} = \{C_j \mid 1 \leq j \leq n\}$.

After having received \mathcal{C} from the TTP, each P_i can verify for which $C_j, j \in [1, n], f(C_j \cdot R_{i-1})$ is equal to $f(m_{i-1})$, and obtain then m_{i-1} .

Here is a summary of that protocol:

Protocol 1 Franklin and Tsudik’s fair exchange protocol

1. $\forall i \in [1, n]: P_i \rightarrow P_{i+1} : R_i$
2. $\forall i \in [1, n]: P_i \rightarrow TTP : A_i, C_i, f(R_i)$
3. $TTP \Rightarrow \mathcal{C}$

Franklin and Tsudik claimed in a sketch of proof [7] that their protocol was correct. We show below that this is not true by describing two attacks on this protocol.

3.2. A first attack

Unfortunately, the above protocol is not fair. When communicating with the TTP, a target actor, say P_i , can randomly choose a value \tilde{R} distinct from the R_i transmitted to P_{i+1} at the previous step of the protocol and can therefore send to the TTP the “normal” $A_i, C_i = m_i \cdot \tilde{R}^{-1}$ and $f(\tilde{R})$.

The TTP will not be able to realise that the value \tilde{R} provided by P_i is different from the value R_i received by P_{i+1} . Indeed, the TTP compares all the received $A_j, j \in [1, n]$, which are not related to R_i nor \tilde{R} , and computes

$$F_{n+1}(C_1 \cdots C_n, f(R_1), \dots, f(R_{i-1}), f(\tilde{R}), f(R_{i+1}), \dots, f(R_n)),$$

which should be equal to any A_j .

Therefore, P_{i+1} will obtain from the TTP the set of all C_j , where $j \in [1, n]$. P_{i+1} has to compute $\widehat{m}_j = C_j \cdot R_i$ for all $j \in [1, n]$ until $f(\widehat{m}_j) = f(m_i)$. Unfortunately, for P_{i+1} this last equality will never be verified. He will not be able to retrieve his expected target information and, regardless of the solution used by the TTP to determine the moment at which the exchange can be resolved, the fairness property will thus be broken.

As a result, we have that P_i excludes P_{i+1} from the exchange (P_{i+1} is not able to reach one of his strong completion points) with the passive assent of the remaining target actors, who become passive conspirators of P_i .

3.3. A second attack

We present now a second exclusion attack with passive conspirators. Suppose that P_i decides to exclude P_{i+1} from the exchange. If the TTP resolves the exchange without knowing the identities of all the target actors and if the one-way function f is, as in [7], such that $f(a) \cdot f(b) = f(ab)$, which with discrete arithmetic one-way functions is a reasonable assumption, then P_i can act as described below (Fig. 1; the information sent by P_i is detailed in the textual description).

P_i follows normally the protocol, except that he does not send his random value R_i to P_{i+1} . The protocol does not specify that the target actors have to wait to receive the random value from the previous target actor in the ring before sending their own random value to the next target actor. Therefore, as for the remaining target actors, P_{i+1} sends his random value to P_{i+2} . The only target actor having not received his expected random value is then P_{i+1} . In order to remain in a fair state, P_{i+1} does not con-

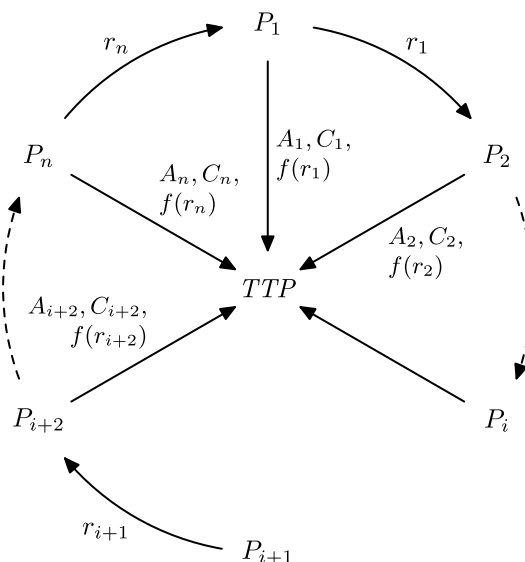


Fig. 1. An exclusion scenario.

tinue the protocol, as he will not be able to retrieve m_i without R_i .

All the target actors except P_i and P_{i+1} contact the TTP as described in the second step of the protocol. P_{i+1} does not send anything. If the TTP resolves the exchange at the time specified by a deadline t , P_i contacts the TTP by sending to the latter

$$A_i = F_n(m_i, f(m_1), \dots, f(m_{i-1}), f(m_{i+1}), \dots, f(m_n)),$$

$C_i = m_i \cdot R_i^{-1}$, and $f(R_i) \cdot f(m_{i+1})$ instead of $f(R_i)$. For all $j \in [1, n]$, with $j \neq i + 1$, $A_j = f(m_1 \cdots m_n)$. The TTP computes then

$$F_n(C_1 \cdots C_i \cdot C_{i+2} \cdots C_n, f(R_1), \dots, f(R_{i-1}), f(R_i) \cdot f(m_{i+1}), f(R_{i+2}), \dots, f(R_n)),$$

which is equal to all the A_j , and cannot therefore detect that P_{i+1} has not contacted it.

On the other hand, if the TTP resolves the exchange once it has received n messages (possibly before a deadline t), P_i sends the message that he should normally send according to the protocol, as well as a second message containing

$$\widetilde{A}_{i+1} = F_n(m_i, f(m_1), \dots, f(m_{i-1}), f(m_{i+1}), \dots, f(m_n)),$$

$\widetilde{C}_{i+1} = \tilde{R}^{-1}$, and $f(\tilde{R}) \cdot f(m_{i+1})$, where \tilde{R} is a randomly chosen value. Again, all the tests performed by the TTP succeed, and this participating actor cannot detect that P_{i+1} has not contacted it.

Note that P_{i+1} is excluded but remains in a fair state (he does not receive m_i and does not send m_{i+1}). However, P_{i+2} sends his target information m_{i+2} without receiving P_{i+1} 's target information m_{i+1} , and he is thus not able to reach one of his strong completion points. Fairness is then broken.

This attack allows P_i to exclude P_{i+1} and P_{i+2} with the passive assent of the remaining target actors. Therefore, under our assumptions, this protocol is not strongly nor weakly exclusion-free, it implies passive conspirators and does not respect the fairness property.

We can thus see that, in this protocol, the exclusion of target actors from the exchange leads to a loss of fairness and prevents the protocol from being implementable in an electronic commerce application. Moreover, the existence of passive conspirators when an exclusion occurs may induce excluded target actors to refuse any future transaction with the remaining target actors, which may have a negative effect on the excluded target actor's confidence towards electronic commerce and will damage the passive conspirator's reputation.

4. A strongly exclusion-free fair exchange protocol

The protocol that we describe in this section is a fixed version of the multi-party fair exchange protocol with an online trusted third party described above. The exchange topology of this protocol describes also a ring and the communication channels between target actors may be unreliable, while those used between each target actor and the

TTP are assumed to be resilient. Functions f and F are defined as for the original protocol.

At the end of the setup phase:

- all the target actors have agreed on the identity of the TTP that will be contacted during the protocol execution and on the instance of the functions f and F that will be used;
- each target actor knows the set $\mathcal{P} = \{P_0, P_1, \dots, P_{n-1}\}$ of identities of all the target actors in the exchange and (at least) the identity of the previous and the next target actor in the ring;
- the description of the target information to be exchanged, $f(m_i) \forall i \in [0, n - 1]$, is made public;
- a deadline t before which the TTP should have received all the necessary information in order to successfully complete the protocol has been agreed by all the target actors;
- and a label, identifying a protocol run and resulting from applying the concatenation of \mathcal{P} , the description of the target information expected by each target actor (in the order found in \mathcal{P}), and of the deadline t (in this order) to a one-way collision-resistant hash function, is known by all the target actors.

During the protocol, when receiving a signed message, each actor checks the validity of the signature. If the verification fails, then the message is ignored.

4.1. The protocol

As in the original protocol, each target actor $P_i \forall i \in [0, n - 1]$, randomly chooses a value R_i and sends it, in a private way, to P_{i+1} .

Upon receiving R_{i-1} , each P_i computes $C_i = m_i \cdot R_i^{-1}$ and sends $E_{TTP}(C_i)$ to the TTP along with $f(R_i)$ and $f(R_{i-1})$, the label, the set \mathcal{P} of target actors, and the deadline t .

In order to avoid attacks from actors not belonging to \mathcal{P} , the TTP verifies for each received message if its sender is included in the set \mathcal{P} found in the message. The TTP also verifies if this set, the public information corresponding to all the target actors in \mathcal{P} , and t are consistent with the label that has been sent. If so, the TTP stores the received message. Otherwise, it discards the message and informs its sender of this.

At time t , the TTP verifies if the set of identities of the target actors having contacted it with the same label is equal to the corresponding set \mathcal{P} . As shown in the first attack for the previous protocol, this test is not sufficient to prevent exclusions. It is also necessary for the TTP to be sure that the random values of the first step have been properly exchanged, i.e. the TTP checks if for each $i \in [0, n - 1]$ the $f(R_i)$ sent by P_i is equal to the $f(R_i)$ sent by P_{i+1} and if the $f(R_{i-1})$ sent by P_i is equal to the $f(R_{i-1})$ sent by P_{i-1} .

If these two first checks succeed, the TTP has still to verify whether each target actor P_i will be able to recover his expected target information from C_{i-1} , as this last value

could be intentionally malformed. More precisely, the TTP checks if, for each $i \in [0, n - 1]$, $F_2(C_i, f(R_i)) = f(m_i)$.

Note that [11] the technique used in the original protocol (testing whether $\forall i \in [1, n]$, $A_i = F_{n+1}(C_1 \cdots C_n, f(R_1), \dots, f(R_n))$) is not sufficient to verify if each target actor will be able to recover his expected target information, since two dishonest target actors, say P_j and P_k , could send to the TTP respectively $\widetilde{C}_j = m_j \cdot R_k^{-1}$ and $\widetilde{C}_k = m_k \cdot R_j^{-1}$ and therefore exclude P_{j+1} and P_{k+1} , who will not be able to recover their expected target information. The well-formedness of the received C_i has thus to be verified individually for each C_i .

Finally, if all these tests succeed, the TTP multicasts $\mathcal{C} = \{C_j | 0 \leq j \leq n - 1\}$ to all the target actors. Any P_i can therefore verify for which C_j , with $j \in [0, n - 1]$, $f(C_j \cdot R_{i-1}) = f(m_{i-1})$ holds, and can retrieve m_{i-1} . Otherwise, if any of the above tests fails, the TTP multicasts to all the target actors a failure message.

Here are the three steps of that protocol:

Protocol 2 A strongly exclusion-free fair exchange protocol

1. $\forall i \in [0, n - 1]$:
 $P_i \rightarrow P_{i+1} : f_1, P_{i+1}, label, E_{P_{i+1}}(R_i), S_{P_i}(\star)$
 2. $\forall i \in [0, n - 1]$:
 $P_i \rightarrow TTP : f_2, TTP, label, \mathcal{P}, t, E_{TTP}(C_i), f(R_i), f(R_{i-1}), S_{P_i}(\star)$
 3. if the tests succeed
 $TTP \Rightarrow \mathcal{P} : f_{success}, \mathcal{P}, label, \mathcal{C}, S_{TTP}(\star)$
 else
 $TTP \Rightarrow \mathcal{P} : f_{failure}, \mathcal{P}, label, S_{TTP}(\star)$
-

4.2. Analysis

Property 1. The first and last steps of the protocol are strong completion points for all the target actors.

Property 2. The protocol is strictly fair.

Proof. If some target actors stop the protocol after performing the first step (or before), all the target actors will be able to reach one of their strong completion points and these completion points will be such that all the honest target actors will be in the second state of the definition of a strong completion point.

Now, after performing the second step of the protocol, no target actor has interest in stopping the protocol because he might not receive his expected target information. If at time t the TTP has not received all the necessary messages in order to complete the protocol or if one of these messages induces a test failure, then the TTP will multicast a failure message to all the target actors and no target information will be exchanged. Otherwise, the TTP will multicast \mathcal{C} to all the target actors, who will therefore receive their expected target information.

In both cases, thanks to the assumed resiliency of the communication channels with the TTP, all the honest target actors will be able to reach their strong completion point at the third step of the protocol. In the first scenario, these strong completion points will be such that all the honest target actors will be in the second state of the definition of a strong completion point. In the second scenario, these strong completion points will be such that all the honest target actors will be in the first state of this definition. \square

Corollary 3. As the protocol is strictly fair, it respects the strong exclusion-freeness property.

Property 4. The protocol is viable and respects the timeliness property.

4.3. Conclusion

Regardless robust protocol design [1], fixing the exclusion problems of the original protocol is not as simple as adding $f(R_{i-1})$ to the second message of the protocol. The TTP must assure strict fairness by verifying whether all the target actors have contacted it, whether all the random values of the first step of the protocol have been properly exchanged and whether all the target information can be extracted from \mathcal{C} with the help of all the $f(R_{i-1})$. While the first and last steps of this protocol remain quite similar to those of the original protocol, the tests realised by the TTP are totally different.

5. Bao et al.'s fair exchange protocol

We now focus on the study of exclusions in the framework of the optimistic multi-party fair exchange protocol proposed by Bao et al. [5]. The exchange topology is a ring. At the end of the setup phase:

- all the target actors in the exchange have agreed on the identity of the TTP that will be contacted if a problem occurs and know their own position in the ring;
- the identity of P_0 , the target actor that initiates the exchange, is known by the TTP;
- and the description of the target information to be exchanged, $f(m_i)$, with $i \in [0, n - 1]$, are made public (f is defined hereunder).

Before describing the protocol, let us explain the technique of verifiable encryption schemes used in this protocol.

5.1. Verifiable encryption schemes

To illustrate the aim of this technique, imagine a scenario with a trusted third party and two actors, Alice and Bob, who know the public encryption key of this TTP. And let f be a one-way function.

Alice knows a secret information i , with $f(i)$ being public, that Bob aims to obtain. Alice can use a verifiable

encryption scheme to deliver this information to Bob in two steps as follows. She encrypts i with the TTP's public key in order to produce $c = E_{TTP}(i)$, generates a certificate $cert_A = certify(i, c, TTP)$ using a public algorithm *certify*, and sends c and $cert_A$ to Bob.

Bob checks that $cert_A$ is a correct certificate by using a public algorithm *verify* such that $verify(c, cert_A, f(i), TTP) = yes$ if and only if $f(D_{TTP}(c)) = f(i)$. Bob is then convinced that c is indeed the encryption of i with the TTP's public key. Later, Bob will be able to obtain i either directly from Alice or by asking the TTP to decrypt c .

A verifiable encryption scheme must thus satisfy [5] the two following properties:

- it is computationally unfeasible for Alice to generate a certificate $cert_A$ such that $verify(c, cert_A, f(i), TTP) = yes$, while $f(D_{TTP}(c)) \neq f(i)$;
- and it is computationally unfeasible for Bob to get i from c without knowing the TTP's secret decryption key.

Asokan et al. [4] gave some examples of verifiable encryption schemes implementations. Bao et al. [5] proposed an implementation of a non-interactive scheme, corresponding to the description above.

5.2. Main protocol

P_0 begins the main protocol by sending to P_1 the ciphertext c_0 of the target information m_0 encrypted with the TTP's public key, along with a certificate $cert_0$ proving, as described above, that c_0 is indeed the ciphertext of m_0 .

After receiving $(c_0, cert_0)$, P_1 checks $cert_0$. If this certificate is valid, he encrypts m_1 and sends $(c_1, cert_1)$ to P_2 . For $i = 2, 3, \dots, n-1$, target actor P_i behaves similarly.

When P_0 receives $(c_{n-1}, cert_{n-1})$ he checks the certificate $cert_{n-1}$ and, if it is valid, he sends the target information m_0 to P_1 . For $i = 1, 2, \dots, n-1$, after receiving m_{i-1} from target actor P_{i-1} , P_i sends the target information m_i to P_{i+1} .

These are the two rounds of the main protocol:

Protocol 3 Bao et al.'s fair exchange protocol: main protocol

1. For $i = 0, \dots, n-1$: $P_i \rightarrow P_{i+1} : c_i, cert_i$
 2. For $i = 0, \dots, n-1$: $P_i \rightarrow P_{i+1} : m_i$
-

5.3. Recovery protocol

If all the target actors (and the communication channels between them) behave correctly, after the main protocol execution each target actor should obtain his expected target information. However, if some P_i does not receive m_{i-1} (within a reasonable amount of time), he has to run the following recovery protocol.

P_i begins this protocol by sending $(c_{i-1}, cert_{i-1})$ to the TTP. The latter checks the certificate $cert_{i-1}$, and, if it is

valid and if $P_i \neq P_0$, waits for the time¹ it would take to P_0 to obtain m_{n-1} if no problem occurs, before asking P_0 if he has received a valid $(c_{n-1}, cert_{n-1})$ during the first step of the main protocol. Only if P_0 answers *yes*, the TTP will accept to decrypt the corresponding c_{i-1} . On the other hand, if the certificate is valid and if the target actor running the recovery protocol is P_0 , the TTP decrypts c_{n-1} and considers that he has answered *yes* to the above question.

The TTP does not contact P_0 each time that some other target actor runs the recovery protocol: if P_0 answered *yes* in the first recovery execution then the TTP will accept further recovery requests; otherwise the TTP will reply with an *abort* message. The above call to P_0 prevents a target actor P_i from getting c_{i-1} decrypted without having sent $(c_i, cert_i)$.

Here are the four steps of the recovery protocol, initiated by some P_i having not received m_{i-1} . Steps 2 and 3 are only executed the first time that this protocol is invoked.

Protocol 4 Bao et al.'s fair exchange protocol: recovery protocol

1. $P_i \rightarrow TTP : c_{i-1}, cert_{i-1}$
 2. $TTP \rightarrow P_0 : call$
 3. $P_0 \rightarrow TTP : yes$ or *abort*
 4. $TTP \rightarrow P_i : m_{i-1}$ or *abort*
-

5.4. Analysis

This protocol respects the strong fairness property (note that the protocol of Section 4 was strictly fair). We refer the reader to the original paper [5] for the details.

5.4.1. Exclusions and passive conspiracies

As pointed out by Bao et al. [5], in that protocol, two or more target actors can collude in order to exclude some other target actors from the exchange. Consider for example that P_0 colludes with P_i . The attack would take place as follows (Fig. 2).

The first step of the main protocol is correctly executed until P_i receives the message $(c_{i-1}, cert_{i-1})$ from P_{i-1} . Instead of sending $(c_i, cert_i)$ to P_{i+1} , P_i runs the recovery protocol in order to get c_{i-1} decrypted by the TTP. The latter asks then P_0 if he has received $(c_{n-1}, cert_{n-1})$, P_0 answers with a false *yes*, and the TTP decrypts c_{i-1} .

P_i will obtain m_{i-1} without having sent $(c_i, cert_i)$. This causes no harm as long as the TTP correctly follows the protocol: P_1 to P_{i-1} will be able to run the recovery protocol, obtain respectively m_0 to m_{i-2} and reach their strong completion point in the first actor's state at the third step of the recovery protocol; and P_{i+1} to P_{n-1} will not receive nor send anything: they will be simply excluded from the

¹ This time is estimated by the TTP.

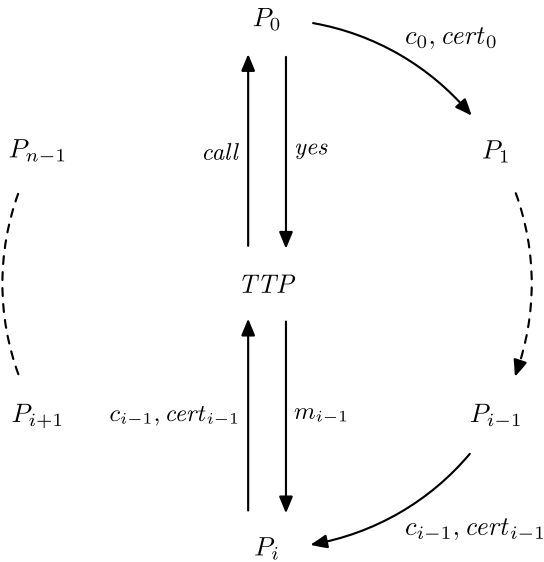


Fig. 2. An exclusion scenario.

exchange, and can thus be seen to be in the second state of the definition of a strong completion point.

Even if the strong fairness property is respected (all the honest target actors reach one of their strong completion points, but in different actor’s states), in the coalition described above, P_1 to P_{i-1} become passive conspirators. They must trust P_0 not to answer with a false *yes* to the TTP during the recovery protocol. In fact, as P_i could for example collude with P_{i+2} in order to exclude P_{i+1} from the exchange (merely by bypassing P_{i+1} and directly contacting P_{i+2}), target actors in that protocol have to trust each other not to become passive conspirators of a coalition excluding some other target actors from the exchange.

On the other hand, we note that even if P_0 decides to send a false *abort* to the TTP during the recovery protocol, strong fairness is still preserved. Once the first step of the main protocol has been successfully completed, P_0 sends m_0 to P_1 and target actors P_1 to P_{i-1} behave similarly. If P_i decides not to send the correct m_i , or if m_i is lost, after a certain amount of time P_{i+1} will realise that he must run the recovery protocol in order to obtain m_i . However, if P_0 sends a false *abort* to the TTP, the protocol will be terminated without P_{i+1} receiving m_i . If the TTP behaves correctly, P_0 will be the only target actor in a non-fair state at the end of the protocol, and P_{i+1} to P_{n-1} will be excluded.

In this protocol, fairness is achieved even if exclusions occur. However, as for the Franklin and Tsudik’s protocol, the existence of passive conspirators is an adverse consequence for any honest target actor. We believe that this remains unacceptable for a practical electronic commerce application.

6. A weakly exclusion-free fair exchange protocol

We propose now a variant of the multi-party fair exchange protocol with an offline trusted third party

described above. The aim of this new protocol is to no longer require target actors to trust each other not to become passive conspirators of a coalition excluding some other target actors, and allow possible excluded target actors to become weakly excluded. The exchange topology is a ring. The communication channels between target actors may be unreliable and those between each target actor and the TTP are assumed to be resilient.

At the end of the setup phase:

- all the target actors have agreed on the identity of the TTP that will be contacted if a problem occurs and know their own position in the ring;
- the identity of P_0 , the target actor that initiates the exchange, is known by all the target actors as well as by the TTP, and the address of a public read-only directory managed by P_0 is assumed to be associated with his identity;
- each target actor knows the set $\mathcal{P} = \{P_0, P_1, \dots, P_{n-1}\}$ of identities of all the target actors in the exchange;
- the description of the target information to be exchanged, $f(m_i) \forall i \in [0, n - 1]$, is made public;
- and a label, identifying a protocol run and resulting from applying the concatenation of \mathcal{P} and of the description of all the target information to be exchanged (in this order) to a one-way collision-resistant hash function, is known by all the target actors.

During the protocol, when receiving a signed message, each actor checks the validity of the signature. If the verification fails, then the message is ignored.

6.1. Main protocol

As in the original protocol, P_0 begins the main protocol by sending to P_1 the ciphertext c_0 of the target information m_0 , the certificate $cert_0$ proving that c_0 is indeed the ciphertext of m_0 encrypted with the TTP’s public key, and his signature on these information.

After receiving $(c_0, cert_0)$ from P_0 , if $cert_0$ is valid, P_1 encrypts m_1 and sends $(c_1, cert_1)$, digitally signed, to P_2 . For $i = 2, 3, \dots, n - 1$, each P_i behaves similarly.

Upon P_0 receiving $(c_{n-1}, cert_{n-1})$, if the certificate sent by P_{n-1} is valid, P_0 stores $(label, E_{TTP}(S_{P_0}(label)))$ in his public directory and verifies, by consulting the logs of this public directory, if the TTP has tried to perform a recovery before he stored this encrypted signature. As it will be described below, the TTP consults this public directory when it is invoked for the first time. Thus, if a recovery has not been invoked yet for this protocol execution, P_0 multicasts $S_{P_0}(label)$ to the set $\mathcal{P} \setminus P_0$ of target actors.

This signature is a non-target information representing the successful completion (in P_0 ’s opinion) of the first step of the protocol, that will allow non-excluded honest target actors not to become potential passive conspirators. By sending it to possible excluded actors, these honest target

actors will be able to inform the possible excluded actors that, if the exchange protocol execution respects the exchange graph involving all the target actors, then the possible excluded actors should receive their expected target information.

In addition to this signature, P_0 sends the target information m_0 to P_1 , digitally signed. When P_i , with $i \in [1, n - 1]$, receives a valid $S_{P_0}(label)$ for the first time, he multicasts this signature to $\mathcal{P} \setminus P_i$. Upon receiving such a signature and m_{i-1} from P_{i-1} , P_i sends m_i to P_{i+1} , digitally signed.

Here is a summary of the main protocol:

Protocol 5 A weakly exclusion-free fair exchange protocol: main protocol

1. For $i = 0, \dots, n - 1$: $P_i \rightarrow P_{i+1} : f_{m1}, P_{i+1}, label, c_i, cert_i, S_{P_i}(\star)$
 2. $P_0 \Rightarrow \mathcal{P} \setminus P_0 : f_{S_{P_0}(label)}, \mathcal{P} \setminus P_0, label, S_{P_0}(label), S_{P_0}(\star)$
 $P_0 \rightarrow P_1 : f_{m2}, P_1, label, m_0, S_{P_0}(\star)$
 3. For $i = 1, \dots, n - 1$:
 $P_i \Rightarrow \mathcal{P} \setminus P_i : f_{S_{P_0}(label)}, \mathcal{P} \setminus P_i, label, S_{P_0}(label), S_{P_i}(\star)$
 $P_i \rightarrow P_{i+1} : f_{m2}, P_{i+1}, label, m_i, S_{P_i}(\star)$
-

6.2. Recovery protocol

If some P_i does not receive m_{i-1} during the main protocol, then he has to run the following recovery protocol in order to ask the TTP to decrypt c_{i-1} . P_i should also invoke this protocol if he does not receive $S_{P_0}(label)$, in order to receive this signature and be able to send it to possible excluded target actors afterwards.

P_i begins this protocol by sending c_{i-1} and $cert_{i-1}$ to the TTP, along with the label, the set \mathcal{P} of target actors, and his digital signature on all this information.

In order to avoid attacks from actors not belonging to \mathcal{P} , the TTP verifies for each received message if its sender is included in the set \mathcal{P} found in the message. The TTP also verifies if this set and the public information corresponding to all the actors in \mathcal{P} are consistent with the label that has been sent. If not, the TTP discards the message.

The first time that the TTP is invoked for a protocol execution identified by the label contained in a received message, it tries to fetch $(label, E_{TTP}(S_{P_0}(label)))$ from the public directory managed by P_0 . If this signature can be retrieved and properly verified, that is to say if the first step of the main protocol is considered as successfully completed, then the TTP will accept the current as well as future recovery requests concerning this label. Otherwise, it will always reply with a failure message.

If the TTP accepts recovery requests, then it will verify if the certificate $cert_{i-1}$ is correct. If it is the case, the TTP decrypts c_{i-1} and sends $S_{P_0}(label)$ and m_{i-1} to P_i . If not, it sends an error message to P_i (asking him to resend a correct message).

The three steps of the recovery protocol for some P_i having not received m_{i-1} or $S_{P_0}(label)$ are shown below. Step 2 is only performed the first time that this protocol is invoked.

Protocol 6 A weakly exclusion-free fair exchange protocol: recovery protocol

1. $P_i \rightarrow TTP : f_{r1}, TTP, label, \mathcal{P}, c_{i-1}, cert_{i-1}, S_{P_i}(\star)$
 2. $TTP \leftrightarrow P_0 : f_{r2}, TTP, label, E_{TTP}(S_{P_0}(label)), S_{P_0}(\star)$
 3. a. if the TTP rejects recovery requests
 $TTP \rightarrow P_i : f_{failure}, P_i, label, S_{TTP}(\star)$
 b. else if the tests succeed
 $TTP \rightarrow P_i : f_{success}, P_i, label, S_{P_0}(label), m_{i-1}, S_{TTP}(\star)$
 c. else
 $TTP \rightarrow P_i : f_{error}, P_i, label, S_{TTP}(\star)$
-

6.3. Analysis

Property 5. The first step of the main protocol is a strong completion point in the second actor's state for the target actor P_0 .

Property 6. The third step of the main protocol and the step 3b of the recovery protocol are strong completion points in the first actor's state for the target actor P_0 .

Property 7. The third step of the main protocol is a strong completion point for a target actor P_i , with $i \in [1, n - 1]$.

Proof. We consider here that the TTP always accepts recoveries, the case where the TTP rejects recoveries will be discussed in [Property 8](#). Several scenarios are possible for an honest target actor P_i :

- if P_i receives m_{i-1} from P_{i-1} and has received $S_{P_0}(label)$, then he performs the third step of the main protocol. At that moment, P_i has received all the target information that he expected and has sent his target information to P_{i+1} . As an honest P_{i-1} would not have sent m_{i-1} if he would have not received m_{i-2} (in this setting P_0 can always obtain m_{n-1}), P_i ends the protocol in the first state of the definition of a strong completion point.
- if P_i does not receive m_{i-1} from P_{i-1} but has received $S_{P_0}(label)$, then he performs the recovery protocol, obtains his expected target information and performs the third step of the main protocol. As an honest P_{i-1} would not have sent $(c_{i-1}, cert_{i-1})$ if he would have not received $(c_{i-2}, cert_{i-2})$, then P_{i-1} is also able to obtain his expected target information and P_i ends the protocol in the first state of the definition of a strong completion point.
- suppose now that P_i has received m_{i-1} from P_{i-1} but has not received $S_{P_0}(label)$. As an honest P_{i-1} would not have sent m_{i-1} if he would have not received m_{i-2} , P_i has already reached at that moment the first state of the definition of a strong completion point. However,

an honest P_i will try to perform the recovery protocol, obtain $S_{P_0}(label)$ and perform the third step of the main protocol.

- if P_i does neither receive m_{i-1} from P_{i-1} nor $S_{P_0}(label)$, P_i needs to contact the TTP in order to obtain his expected target information. By doing so, P_i will receive $S_{P_0}(label)$ as well as m_{i-1} , and will thus be able to perform the third step of the main protocol. As an honest P_{i-1} would not have sent $(c_{i-1}, cert_{i-1})$ if he would have not received $(c_{i-2}, cert_{i-2})$, P_{i-1} will also be able to obtain his expected target information. P_i will therefore end the protocol in the first state of the definition of a strong completion point. \square

Property 8. The step 3a of the recovery protocol is a strong completion point for a target actor P_i , with $i \in [1, n - 1]$.

Proof. We consider here that the TTP always rejects recovery requests. Two scenarios are possible for an honest target actor P_i :

- if P_i does not receive m_{i-1} from P_{i-1} but has received $S_{P_0}(label)$, then he performs the recovery protocol and obtains a failure message from the TTP. P_i ends therefore the protocol in the second state of the definition of a strong completion point since P_i will not send m_i and P_{i+1} will not be able to obtain his expected target information from the TTP.
- suppose now that P_i has neither received m_{i-1} from P_{i-1} nor $S_{P_0}(label)$. P_i will therefore perform the recovery protocol. As the TTP rejects recovery requests, P_i will receive a failure message and, as in the previous scenario, will end the protocol in the second state of the definition of a strong completion point. \square

Property 9. The protocol is strongly fair.

Proof. After the first step of the main protocol only P_0 can stop the protocol. Any other target actor P_i , with $i \in [1, n - 1]$, has no interest in stopping the protocol because P_0 could perform the second step of the main protocol and P_{i+1} could therefore successfully ask the TTP to recover m_i . However, if P_0 stops the protocol at that point, no remaining target actor could successfully invoke the recovery protocol and no target information would be exchanged. All these target actors would reach their strong completion point at the step 3a of the recovery protocol as shown in [Property 8](#). As all the target actors would be in the second state of the definition of a strong completion point, strict fairness would be achieved.

After performing the second step of the main protocol, P_0 has no interest in stopping the protocol by the same argument as above. P_0 has therefore to wait for m_{n-1} or invoke the recovery protocol in order to reach, respectively, his strong completion point at the third step of the main protocol or at the step 3b of the recovery protocol. As shown in [Property 7](#), all the remaining honest target actors having received their expected ciphertext and

certificate at the first step of the protocol are always able to reach their respective strong completion point at the third step of the main protocol. Honest target actors having not received their expected ciphertext and certificate (for example, because their respective neighbours have excluded them) will be waiting for the protocol to begin; they can thus be seen to be in the second state of the definition of a strong completion point. Strong fairness is therefore achieved.

Finally, if P_0 performs the second step of the main protocol but does not store $(label, E_{TTP}(S_{P_0}(label)))$ in his public directory, the third step of the main protocol will be properly performed until a target actor invokes the recovery protocol, as the TTP will reject recoveries. Honest target actors having received their expected target information and $S_{P_0}(label)$ will reach their strong completion point at the third step of the main protocol. Remaining honest target actors will either reach their strong completion point at step 3a of the recovery protocol if they have been involved in the first step of the main protocol, or remain in the second actor's state otherwise. The protocol is thus strongly fair. \square

Note that due to dishonest target actors, all honest target actors do not necessarily end the protocol in the same state of the definition of a strong completion point, i.e. target actors reaching the second state of this definition will be considered as being excluded from the exchange. We will further discuss this in the following subsection.

Property 10. The protocol is viable and respects the timeliness property.

Property 11. The protocol allow honest target actors not to become passive conspirators excluding other target actors from the exchange.

Proof. The existence of the non-target information $S_{P_0}(label)$ indicates that P_0 considers that the first step of the main protocol has successfully ended and that the two remaining steps of the main protocol can be performed. Multicasting $S_{P_0}(label)$ allows therefore honest target actors being involved in the last step of the main protocol to inform possible target actors having not received the first message of the main protocol, i.e. being excluded, that the exchange is taking place. It is thus up to each target actor obtaining this non-target information to decide whether multicast $S_{P_0}(label)$ to all the remaining target actors or not, i.e. whether be honest or contribute to a possible existing coalition excluding target actors from the protocol execution. \square

6.4. Complaint protocol

If an honest target actor receives the ciphertext of the expected target information and the corresponding certificate during the first step of the main protocol, then it will be possible for him to run the recovery protocol. Only if $S_{P_0}(label)$ is not present in P_0 's public directory, the TTP

will reply with a failure message. This message will allow the recipient of this message to prove that he has been excluded (i.e. he is weakly excluded) in case other target actors have reached the first state of the definition of a strong completion point.

Otherwise, if an honest target actor does not receive this ciphertext and the corresponding certificate but receives $S_{P_0}(label)$, he will be able to prove to an external party, by executing the following *complaint protocol* with the TTP, that he has been excluded, i.e. he is weakly excluded.

A target actor P_i , with $i \in [0, n - 1]$, begins this complaint protocol by sending $(label, \mathcal{P}, S_{P_0}(label))$ to the TTP. The latter checks if P_i belongs to the set \mathcal{P} , if \mathcal{P} is consistent with the label and if $S_{P_0}(label)$ is a valid signature. If so, the TTP multicasts the signature of P_0 on the label to the target actors belonging to $\mathcal{P} \setminus P_i$ and asks each of them to provide the signature received during the first step of the main protocol.

Due to the resiliency of the communication channels between each target actor and the TTP, all the target actors receive $S_{P_0}(label)$ at that moment. Other excluded target actors having not received previously this signature can therefore also invoke the complaint protocol.

If after a deadline chosen by the TTP according to the communication channels quality, none of the remaining target actors is able to present the signature that P_i should have issued during the first step of the main protocol, the TTP will issue an affidavit attesting that something wrong happened during the protocol. Two (indistinguishable) cases are possible: either an honest target actor was excluded from the exchange, or a dishonest target actor was able to run the complaint protocol because the following target actor in the ring actively involved in the exchange (intentionally) did not provide to the TTP the signature received from the former actor at the first step of the main protocol.

Otherwise, if P_i 's signature is provided, the TTP rejects his complaint, as an honest target actor only performs the first step of the main protocol (and sends the corresponding signature) if he receives $(c_{i-1}, cert_{i-1})$.

Here is a summary of that protocol. Steps 2 and 3 are only executed the first time that this protocol is invoked for a given exchange. Step 4 only occurs if the TTP does not receive the signature that P_i should have issued during the first step of the main protocol.

Protocol 7 A weakly exclusion-free fair exchange protocol: complaint protocol

1. $P_i \rightarrow TTP : f_{c1}, TTP, label, \mathcal{P}, S_{P_0}(label), S_{P_i}(\star)$
 2. $TTP \Rightarrow \mathcal{P} \setminus P_i : f_{c2}, \mathcal{P}, label, S_{P_0}(label), S_{TTP}(\star)$
 3. For $j = 0, \dots, i - 1, i + 1, \dots, n - 1$:
 $P_j \rightarrow TTP : f_{c3}, TTP, f_{m1}, label, c_{j-1}, cert_{j-1},$
 $S_{P_{j-1}}(f_{m1}, P_j, label, c_{j-1}, cert_{j-1}), S_{P_j}(\star)$
 4. $TTP \rightarrow P_i : f_{c4}, P_i, label, \mathcal{P}, S_{TTP}(\star)$
-

Two comments arise while looking at this protocol. The first one is that it is possible that, if there are excluded target actors, none of them receives the signature of P_0 on the

label during the main protocol. In that case, either there is a problem with the communication channels (and the signatures sent by honest target actors are not received by the excluded actors) or there are no honest target actors having realised the exchange. This coalition of a group of target actors in order to exclude all the remaining actors does not seem very interesting to us because, as the strong fairness property is assured for excluded entities, dishonest target actors will not be able to obtain any target information from these excluded actors, and so they do not need other actors in order to carry out an exchange.

However, if there is a problem with the communication channels and $S_{P_0}(label)$ is not received by any excluded actor, then the complaint protocol cannot be invoked by the excluded target actors. Honest non-excluded target actors can therefore run the complaint protocol, and send as well to the TTP the signature that they have received at the first step of the main protocol, if they do not receive $S_{P_0}(label)$ from all the other target actors during the main protocol.

The second comment is the following. Suppose that P_i has invoked the complaint protocol. \mathcal{P} has been defined as a set of target actors. If all the target actors in $\mathcal{P} \setminus P_i$ reply as expected to the second message of that protocol, the TTP will know how the signatures from the first step of the main protocol have been exchanged and will know the disposition of the (non-excluded) target actors in the ring. However, the TTP will not be able to determine where the excluded actor P_i should be, as he could be between any couple of target actors.

Otherwise, if \mathcal{P} is defined as an ordered set according to the agreed topology, it will be possible for the TTP to determine the identity of at least one target actor having actively contributed to the coalition. Consider for example the complaint protocol execution shown in Fig. 3. Target actors P_1, P_4 and P_5 have succeeded in excluding P_2 and

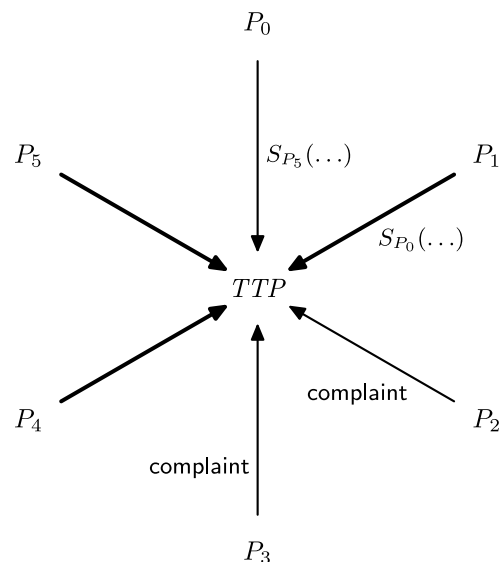


Fig. 3. A complaint scenario.

P_3 from the protocol, without the consent of P_0 . The complaint protocol is invoked (by P_0 , P_2 or P_3) and the TTP asks for the signatures exchanged during the first step of the main protocol. Target actors P_0 and P_1 reply honestly with the signatures of P_5 and P_0 respectively, and P_2 and P_3 join the complaint protocol execution by claiming to have been excluded from the exchange.

P_4 and P_5 can reply in several different ways. P_5 cannot pretend to be excluded because the TTP receives his signature from P_0 . He can neither ignore the TTP's demand because, as the communication channel between him and the TTP is resilient, this will show that he has something to hide. If he does not want to be declared as a conspirator, P_5 must betray the coalition and reply with P_4 's signature.

Now, if P_4 replies with P_1 's signature, then the TTP will be able to determine that at least P_1 and P_4 have conspired in order to exclude P_2 and P_3 from the exchange. If he decides not to answer, then, independently of P_5 's reply, the TTP will be able to consider that P_4 has contributed to the coalition. Finally, if P_4 invokes the complaint protocol, only if P_5 sends P_4 's signature to the TTP, the latter will be able to conclude that P_4 is a member of the coalition.

Property 12. This complaint protocol allows the optimistic multi-party fair exchange protocol described above to be weakly exclusion-free.

Dishonest target actors will either wrongly invoke the complaint protocol or stop the protocol. Stopping the protocol is not a problem because they will not receive any kind of affidavit from the TTP. If they invoke the complaint protocol they may receive an affidavit, however this does not prevent true excluded target actors from obtaining an evidence attesting having been excluded, and become weakly excluded.

6.5. Conclusion

In this optimistic protocol there is no way to prevent P_i and P_{i+2} , for example, from colluding in order to exclude P_{i+1} from the exchange. Fortunately, the protocol remains strongly fair, there will not be passive conspirators and, moreover, P_{i+1} will be able to prove to an external actor that he has been excluded. But he will not be able to obtain its expected target information. This local drawback prevents this protocol from respecting the strong exclusion-freeness property.

The proposed protocol is very similar to the protocol of Bao et al. from a structural point of view. Moreover, it also requires that target actors trust each other not to be excluded from a protocol execution. However, contrary to the original protocol, it allows target actors not to become passive conspirators excluding some other target actors from the protocol execution. Therefore, target actors do no longer need to trust each other not to become passive conspirators. Trust relationships are reduced by increasing communication needs. It is not easy to compare

this reduction of trust relationships with the resulting increase of communication needs. However, the latter is measurable, unlike trust aspects.

7. Final remarks

In this paper, we have studied exclusion scenarios in the framework of multi-party fair exchange protocols, as well as the related trust relationships that may exist between target actors. We have seen that exclusions may lead to a loss of fairness and that the existence of passive conspirators may improperly disable future commercial transactions with them. None of these events must occur in order to electronic commerce transactions involving more than two entities to be successful.

From a theoretical point of view, we think that, due to its close connection with the fairness properties, exclusion-freeness can be an interesting property to consider when designing new multi-party exchange protocols, in order to analyse more deeply if the fairness property is really respected.

Finally, as discussed above, the protocol with an online TTP respects the strong exclusion-freeness property. However, this is not the case for our optimistic protocol. Whether strictly fair optimistic multi-party fair exchange protocols exist remains an open question to us.

References

- [1] M. Abadi, R. Needham, Prudent engineering practice for cryptographic protocols, *IEEE Transactions on Software Engineering* 22 (1) (1996) 6–15.
- [2] N. Asokan, M. Schunter, M. Waidner, Optimistic protocols for multi-party fair exchange, Research Report RZ 2892 (# 90840), IBM Research, December, 1996.
- [3] N. Asokan, V. Shoup, M. Waidner, Asynchronous protocols for optimistic fair exchange, in: *Proceedings of 1998 IEEE Symposium on Security and Privacy*, IEEE Computer Society Press, 1998, pp. 86–99, Research Report RZ 2973 (#93019), IBM Research.
- [4] N. Asokan, V. Shoup, M. Waidner, Optimistic fair exchange of digital signatures, *IEEE Journal on Selected Areas in Communications* 18 (4) (2000) 593–610.
- [5] F. Bao, R. Deng, K.Q. Nguyen, V. Vardharajan, Multi-party fair exchange with an off-line trusted neutral party, in: *Proceedings of the 10th International Workshop on Databases and Expert Systems Applications (DEXA)*, IEEE Press, 1999, pp. 858–863.
- [6] F. Bao, R.H. Deng, W. Mao, Efficient and practical fair exchange protocols with off-line TTP, in: *Proceedings of 1998 IEEE Symposium on Security and Privacy*, IEEE Computer Society Press, 1998, pp. 77–85.
- [7] M. Franklin, G. Tsudik, Secure group barter: multi-party fair exchange with semi-trusted neutral parties, in: *Proceedings of the 2nd International Conference on Financial Cryptography*, Lecture Notes in Computer Science, vol. 1465, Springer, Berlin, 1998, pp. 90–102.
- [8] M.K. Franklin, M.K. Reiter, Fair exchange with a semi-trusted third party, in: *Proceedings of the 4th ACM Conference on Computer and Communications Security*, ACM Press, 1997, pp. 1–5.
- [9] N. González-Deleito, O. Markowitch, An optimistic multi-party fair exchange protocol with reduced trust requirements, in: *Proceedings of the 4th International Conference on Information Security and Cryptology*, Lecture Notes in Computer Science, vol. 2288, Springer, Berlin, 2001, pp. 258–267.

- [10] N. González-Deleito, O. Markowitch, Exclusion-freeness in multi-party exchange protocols, in: Proceedings of the 5th Information Security Conference, Lecture Notes in Computer Science, vol. 2433, Springer, Berlin, 2002, pp. 200–209.
- [11] S. Kremer, A. Mukhamedov, E. Ritter, Analysis of a multi-party fair exchange protocol and formal proof of correctness in the strand space model, in: Proceedings of the 9th International Conference on Financial Cryptography and Data Security, Lecture Notes in Computer Science, vol. 3570, Springer, Berlin, 2005, pp. 255–269.
- [12] O. Markowitch, Les protocoles de non-répudiation, Ph.D. Thesis, Université Libre de Bruxelles, January, 2001.
- [13] O. Markowitch, R. Devillers, N. González-Deleito, On the formalisation of exchange protocols, Unpublished manuscript.
- [14] O. Markowitch, S. Saeednia, Optimistic fair-exchange with transparent signature recovery, in: Proceedings of the 5th International Conference on Financial Cryptography, Lecture Notes in Computer Science, vol. 2339, Springer, Berlin, 2001, pp. 339–350.
- [15] N. Zhang, Q. Shi, Achieving non-repudiation of receipt, *The Computer Journal* 39 (10) (1996) 844–853.
- [16] J. Zhou, D. Gollmann, A fair non-repudiation protocol, in: Proceedings of 1996 IEEE Symposium on Security and Privacy, IEEE Computer Security Press, 1996, pp. 55–61.
- [17] J. Zhou, D. Gollmann, An efficient non-repudiation protocol, in: Proceedings of the 10th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, 1997, pp. 126–132.