Fair Exchange with Guardian Angels

Gildas Avoine and Serge Vaudenay

Swiss Federal Institute of Technology (EPFL)
lasecwww.epfl.ch

Abstract. In this paper we propose a new probabilistic Fair Exchange Protocol which requires no central Trusted Third Party. Instead, it relies on a virtually distributed and decentralized Trusted Third Party which is formalized as a Guardian Angel: a kind of Observer e.g. a tamper proof security device. We thus introduce a network model with Pirates and Guardian Angels which is well suited for Ad Hoc networks.
In this setting we reduce the Fair Exchange Problem to a Synchronization Problem in which honest parties need to eventually decide whether or not a protocol succeeded in a synchronous way through a hostile network which does not guaranty that sent messages will be eventually received. This problem can be of independent interest in order to add reliability of protocol termination in secure channels.

Key words: Fair Exchange, Security Module, Synchronization, Distributed Systems.

1 Introduction

In the Fair Exchange problem, two participants A and B wish to exchange items k_A and k_B in a fair way, i.e. such that either both participants receive the expected item, or nobody can obtain anything valuable. In addition to the fairness property, we generally require that a fair exchange protocol should obey the following properties:

- **Completeness**: the exchange always succeeds when the involved participants behave honestly.
- **Timeliness**: the participants always have the ability to reach, in a finite amount of time, a step in the protocol where they can fairly stop the exchange.

The completeness property avoids the trivial fair protocol where no information at all is exchanged. On the other hand, protocols with infinite communication complexity are avoided by the timeliness property. With regard to the fairness property, literature brings a lot of general or specific definitions [16,22,23,26,29,36,37].

The basic exchange scheme, where the participants send their items one after the other, is obviously unfair: neither A and B would like to be first to send its

item since the first receiver can disrupt the protocol without sending its own item.

Non-perfect fair exchange protocols were proposed in the early eighties in order to decrease the unfairness, involving a complexity cost in term of exchanged messages. These *gradual protocols* are based on the following scheme: each entity alternatively transmits successive bits of the item to exchange until the last bit of each item was sent (the lengths of the items are supposedly equal). (See for example [8,10,11,25,30].) In order to improve this kind of protocol, fractions of bits can be transmitted instead of real bits [31,32]. When the protocol aborts and one party receives the exchanged item, the *a posteriori* computation of the missing bits is made possible to the other party by the protocol. This induces a computation cost. Hence, real fairness of gradual protocols relies on A and B having approximately the same computational power.

The introduction of perfect fairness in these protocols faces to the impossibility result of Even-Yacobi [13], without the help of third parties.

One can achieve perfect fairness with the help of a *Trusted Third Party* (TTP) in the scheme. The first proposed protocols used an *on-line* Trusted Third Party which assures the fairness during the exchanges. The main drawback with this kind of protocol is that it creates a communication bottleneck: TTP must interfere at least once during the protocol. A great improvement of TTP protocols is to use *off-line* Trusted Third Party. Asokan introduced the notion of *optimistic* fairness where the TTP is required only in case of dispute. (See e.g. [1,2,3,4,5,24,28,29,33].) The efficiency of this approach relies on the fact that the environment is mostly honest.

Even if there exists a great body of literature on fair exchange, only one fair exchange protocol, up to our knowledge, tries to take advantage of the presence of security modules [34,35] to enforce fairness, by devising an optimistic fair exchange protocol for electronic commerce. We briefly describe it here. In this protocol, four entities interfere in the exchange: the client, his security module (e.g. a smart card), the vendor (which does not have security module), and the vendor's bank (which is called only in case of conflict). The sketch of the protocol is the following. (1) The vendor sends the item and its description to the client's security module. (2) The client sends the payment and the expected item's description to his security module. (3) After checking item and payment, the security module sends the payment to the vendor. (4) Finally, if the payment is correct, then the vendor must send a payment acknowledgment to the security module, and then later gives the expected item to the client. If the vendor does not send the acknowledgment (he already has the payment), the bank is called in order to restore the fairness. Thus this falls into the optimistic fair exchange protocols category which requires an (off-line) TTP and the assumption that vendors are mostly honest.

All those protocols rely on the assumption that both parties can have access to the TTP though a channel which achieves timeliness: any request to the TTP gets addressed, eventually. This is quite a strong assumption, for instance in mobile systems where an adversary may control all communications

in one network cell. In some environments, like mobile ad hoc networks [19,20], introducing a timely-available unique Trusted Third Party is not desirable nor possible; therefore optimistic protocols are not suitable. We rather concentrate on a Chaum *et Al.*'s observer-based third party. We assume that all participants have one timely-available observer (e.g. a smart card). The originality of this model for fair exchange is that one party may not be able to contact the observer of another party though a timely channel. We can still propose a protocol which ensures probabilistic fairness, without centralized Trusted Third Party, and without assumption on the participants' computational power.

This paper is organized as follow: in Section 2 we first describe the Synchronization Problem, design a probabilistic protocol — the Keep-in-Touch (KiT) Protocol — and analyze it. Section 3 addresses the fair exchange problem in the Pirates and Guardian Angels model. Section 4 illustrates this protocol bringing applications to the Mobile Ad Hoc Networks. We finally conclude.

2 Synchronization Protocol

2.1 Security Model

In this part, we consider two participants A and B who want to achieve a secure transaction in a fair way through a malicious network N. In our model we assume that A and B are able to communicate through two secure channels (one from A to B, the other from B to A) providing confidentiality, integrity, authentication, and sequentiality.

- **Confidentiality** ensures that the message is kept secret from any third party.
- **Integrity** ensures that the message cannot be modified by any third party.
- **Authentication** ensures that no third party can insert a forged message in the channel.
- **Sequentiality** ensures that at any time, the sequence of messages which were received by one party was equal at some time to the sequence of messages which were sent by the other party in the same ordering. In particular, no messages can be replayed, erased, or swapped by any third party.

Note that one important security property is missing in the channel: **timeliness**. Actually, some sent messages can never reach their final destination. Therefore, the only way for a malicious man-in-the-middle N to make the protocol fail is to stop transmitting messages in one direction or another by cutting the channel. Hence our adversarial model for N is a malicious algorithm which decides to cut one channel at some time, or the two channels at the same or at different times. Due to the confidentiality property, the choice on when to cut channels cannot depend on the content of the messages, but only on the number of exchanged messages.

Here is an example of a secure communication channel from A to B. Let m be the message to send and *seq* a sequence number which is incremented each time after a message is sent.

A: increase seq by 1
encrypt m for B
authenticate $(B, seq, \mathrm{ENC}_B(m))$

$A \rightarrow B$: transmission of $\mathrm{AUT}_A(B, seq, \mathrm{ENC}_B(m))$

B: check the identity B
check the authentication from A
check $seq = previous\_seq + 1$
set $previous\_seq \leftarrow seq$
decrypt $\mathrm{ENC}_B(m)$

Here $\mathrm{ENC}_B(m)$ means m encrypted for B and $\mathrm{AUT}_A(m)$ means m authenticated by A. This kind of secure channel can be implemented, for instance by using the SSL/TLS protocol [12].

2.2 Synchronization Problem

We focus here on the synchronization problem[1] which is defined as follows.

Definition 1. *A synchronization protocol between A and B is a protocol which eventually ends with A and B on two possible terminal states: either* `success` *or* `failure`*. We say that the protocol is*

1. **complete** *if A and B always end on the* `success` *state when there is no malicious misbehavior;*
2. **non-trivial** *if either A or B cannot end on the* `success` *state without receiving at least one message;*
3. **fair** *if A and B always end on the same state even in case of misbehavior;*
4. **timely** *if A and B eventually end.*

We say that the protocol is *perfectly fair* when it follows all these properties. When it is not perfectly fair, we define two measures of unfairness.

- *P_a (probability of asynchronous termination) is the maximum of the probability that the protocol ends on an unfair state over all possible misbehavior of N.*
- *P_c (probability that crime pays) is the maximum of the conditional probability that the protocol ends on an unfair state conditioned on N deviating from the protocol over all possible misbehavior of N.*

We recall that we are interested here in honest participants A and B who communicate through an untrusted network but can establish a channel which achieves confidentiality, integrity, authentication, sequentiality, but not timeliness. Perfect fair protocols are impossible in this setting. This motivates our measures for unperfect protocols.

[1] This is equivalent to the well known *Non-Blocking Atomic Commitment* problem in the fault-tolerance literature [17,18].

Note that there is a tricky distinction between P_a and P_c as will be shown in the sequel. The P_a gives confidence to A and B in the fairness of the protocol while P_c gives measures the incentive for a misbehavior.

In order to illustrate the synchronization problem, let's consider the TLS protocol. In TLS 1.0 [12], the client and the server have to close some connections during the execution of a session. To achieve this task, "the client and the server must share knowledge that the connection is ending in order to avoid a truncation attack" [12]. A quite simple procedure is proposed in [12] which consists in sending a `close_notify` message from the initiator to the other party. This prevents from opening new connections in the session until the `close_notify` is properly acknowledged. If the session goes on with new connections it means that the closed connections was fair. This scheme is however not standard and obviously lacks of fair termination, at least for the very last connection. Here fairness of non-terminal connections is guaranteed by the fact that the client and the server keep-in-touch. To keep-in-touch is actually the key idea to solve the synchronization problem.

2.3 Keep-in-Touch Protocol

Our Keep-in-Touch protocol, depicted on Fig. 1, is a quite simple synchronization protocol: the initiator of the protocol picks a random number C which says how many messages will be exchanged. In case of time-out while expecting a message, a participant ends into a failure state. One can notice that, except the first message, the exchanged messages are really empty ones! The participants just keep-in-touch by exchanging sequential authenticated empty messages.

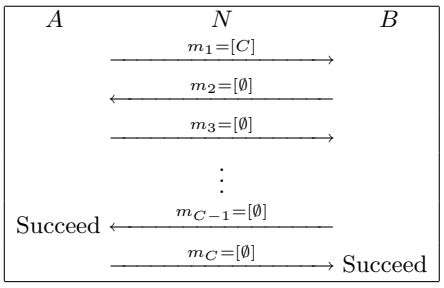

Fig. 1. Keep-in-Touch (KiT) Protocol

Termination side channel protection. In the case that N has access to a side channel saying whether A or B ended in some terminal state, we propose that after the required number of exchanges, the two participants wait for a timeout τ then decide that they succeeded. This extra time prevents a malicious N from

trying to get a side channel from the behavior of the two participants within the time-out interval.

Bit-messages variant. Instead of picking C once and sending it at the beginning of the protocol, we can just ask each participant to toss a biased coin before sending the ith message and sending the output bit in the message. A 0 bit means "let's keep-in-touch" and a 1 bit means "so long". Obviously, if the bit is 1 with probability $\Pr[C = i/C \geq i]$, this variant is fully equivalent to the above protocol.

Since the channel provides confidentiality, the hackers have no clue what C is. The integrity, authentication and sequentiality of messages is also protected, so participants are ensured that (empty) messages were exchanged. The sequence number also prevents from replay attacks. Therefore, the only misbehaving strategy to end up the protocol in an unfair state is to end transmitting messages at some point. If the first dropped message is not the last one, the two participants will end up on a failure state. If it is the last one, the protocol becomes unfair. In other cases, the protocol succeeds before the hackers decide to drop messages. In the analysis of the protocol we will only discuss index number of the first message that hacker drops and the chosen distribution for C.

2.4 Analysis of the KiT Protocol

Here we analyze the Keep-in-Touch Protocol depending on the distribution choice for C.

Complexity. When A, B, and N are honest, the complexity in terms of exchanged messages is exactly equal to C. When someone misbehaves by cutting channels, the complexity is smaller, so we can just focus on C. We let p_i be the probability $\Pr[C = i]$. By definition, the average complexity is

$$E(C) = \sum_{i=1}^{+\infty} i p_i.$$

Note that the communication and time complexities are linear in terms of C due to the simplicity of the message contents and the computations to perform.

Completeness. Obviously, the protocol eventually succeeds with a complexity of C when everyone is honest. Hence the protocol is complete.

Non-triviality. Obviously, the protocol fails if B does not receive C.

Fairness (P_a computation). We assume that N is willing to misbehave the ith message, i.e. to cut the channel which is assumed to transmit this message. If $C < i$ then the misbehavior of N has no influence and the protocol succeeds. If $C > i$, the participant who is expecting the ith message cannot send the next one, so both participants are blocked and the protocol fails in a fair state.

Clearly, the protocol is unfair if $C = i$, thus with probability p_i. Therefore we have

$$P_a = \max_i p_i.$$

Fairness (P_c computation). With the same discussion we can show that the above misbehavior has a conditional probability of success of $\Pr[C = i/C \geq i]$. Hence we have

$$P_c = \max_i \frac{p_i}{\sum_{j \geq i} p_j}.$$

Timeliness. Obviously, the protocol eventually ends due to the timeout management.

Theorem 2. *The KiT Protocol is a complete, non-trivial and timely synchronization protocol. Let $p_1, p_2, \ldots$ denote the probability distribution of C in the protocol. The expected complexity is $E(C) = \sum_{i=1}^{+\infty} ip_i$ and the probability of unfairness is $P_a = \max_i p_i$. The probability that the crime pays is $P_c = \max_i \frac{p_i}{\sum_{j \geq i} p_j}$.*

Example 3. For any n, when $p_1 = \ldots = p_n = \frac{1}{n}$ and $p_i = 0$ for $i > n$ we have an expected complexity of $E(C) = \frac{n+1}{2}$ and a probability of unfairness of $P_a = \frac{1}{n}$. However we have $P_c = 1$ for $i = n$: if the strategy of N is not to forward the nth message, then his risk is void since this is the last message for sure. Hence the protocol is unfair with probability $\frac{1}{n}$ but with no risk at all for Pirates.

Example 4. For any p, when $p_i = (1-p)^{i-1}p$ for $i > 0$ we have an expected complexity of $E(C) = \frac{1}{p}$ and a probability of unfairness of $P_a = p$. In this case we also have $P_c = p$. The equivalent bit-messages variant is where each participant flips a coin of distribution $(1-p, p)$ in every step.

2.5 Optimal Distributions for the KiT Protocol

The distribution choice plays on the complexity and fairness parameters. Obviously there is a trade-off. The optimal case is studied in the following theorem.

Theorem 5. *Let $p_1, p_2, \ldots$ denote the probability distribution of C in the KiT Synchronization Protocol. We have $E(C) \geq \frac{1}{2}\left(\frac{1}{P_a} + 1\right)$ and $E(C) \geq \frac{1}{P_c}$ where P_a and P_c are the highest probability of unfairness and that the crime pays respectively.*

This shows that Example 3 is the optimal case for P_a and that Example 4 is the optimal case for P_c.

Proof. We want to minimize $E(C)$ with $P_a \leq \varepsilon$. It is equivalent to finding $p_1, p_2, \ldots$ such that $0 \leq p_i \leq P_a$ for all i, $\sum p_i = 1$, and $\sum ip_i$ minimal.

Let ε be a probability smaller than P_a. Let $n = \lfloor \frac{1}{\varepsilon} \rfloor$ and $\alpha = \frac{1}{\varepsilon} - n$. We have $\alpha \in [0, 1[$.

Obviously $\sum ip_i$ is minimal when the first p_is are maximal, i.e. when $p_1 = p_2 = \ldots = p_n = \varepsilon$. The sum of all remaining p_i is equal to $1 - n\varepsilon$. Thus we have

$$E(C) \geq \varepsilon + 2\varepsilon + \ldots + n\varepsilon + (n+1)(1 - n\varepsilon).$$

Hence $E(C) \geq \frac{n(n+1)}{2}\varepsilon + (n+1)(1 - n\varepsilon)$. If we substitute $\frac{1}{\varepsilon} - \alpha$ to n we obtain

$$E(C) \geq \frac{1}{2}\left(\frac{1}{\varepsilon} + 1\right) + \alpha + \frac{\alpha\varepsilon}{2}(1 - \alpha).$$

Since $0 \leq \alpha < 1$ we have $E(C) \geq \frac{1}{2}\left(\frac{1}{\varepsilon} + 1\right)$. Since this holds for any $\varepsilon \leq P_a$, it holds for $\varepsilon = P_a$. This proves the first bound.

For the second bound we notice that

$$E(C) = \sum_{i=1}^{+\infty} \sum_{j \geq i} p_j.$$

Since we have $\sum_{j \geq i} p_j \geq \frac{p_i}{P_c}$ by definition of P_c, we obtain that $E(C) \geq \frac{1}{P_c}$. □

3 Fair Exchange with Observers

3.1 Fair Exchange Problem

Several (different) definitions for the fair exchange are available in the literature. Most of them are context-dependent. For completeness we provide an informal one for our purpose.

Definition 6. *An exchange protocol between A and B is a protocol in which A and B own some items k_A and k_B respectively and aim at exchanging them. We say that the protocol is*

1. **complete** *if A gets k_B and B gets k_A at the end of the protocol when there is no malicious misbehavior;*
2. **fair** *if its terminates so that either A gets k_B and B gets k_A (success termination), or A gets no information about k_B and B gets no information about k_A (failure termination) even in case of misbehavior;*
3. **timely** *if A and B eventually end.*

We say that the protocol is *perfectly fair* when it follows all these properties. When the protocol is not perfectly fair, we define two measures of unfairness.

- *P_a (probability of unfair termination) is the maximum of the probability that the protocol ends on an unfair state over all possible misbehaviors.*
- *P_c (probability that crime pays) is the maximum of the conditional probability that the protocol ends on an unfair state conditioned on someone deviating from the protocol over all possible misbehaviors.*

The fair exchange problem looks trivial when A and B are honest: they can just exchange their items one after the other and commit to discard them if the protocol fails. However, if timeliness is not guaranteed for the communication channel, N can just discard the last message and the protocol becomes insecure despite A and B being honest. We solve this here by using the synchronization protocol.

3.2 Security Model: Pirates and Guardian Angels

Our model, based on the notion of "observer" introduced by Chaum and Pedersen [9], considers that both participants own a security module. Contrarily to [9] we assume that the security modules are honest. For this reason, participants and security modules are respectively named "Pirates" and "Guardian Angels". We describe now the properties of these entities.

We assume that the Pirates are powerful in the sense that they are able to communicate with all other devices and their own Guardian Angel. We require no assumption on the computational capabilities of the Pirates, in particular Pirates can have very different capabilities. We have no assumption at all for the inter-Pirates communication channels. In particular they can be totally insecure. On the other hand, the Pirate-Guardian Angel communication channel is assumed to be fully secure: it provides confidentiality, integrity, authentication, sequentiality, and timeliness.

Guardian Angels fulfill the following requirements: they are *tamper-proof*, that is any potential attacker could not have access to the stored data or change the Guardian Angel's behavior. Full access stays possible, but limited to some authorized parties, e.g. for set up. Since the cost for making a tamper-proof device increases steadily with its capabilities, we assume that Guardian Angels are simple and limited devices: their computational and storage ability are low. Moreover they have a single i/o port which is connected to their own Pirate only: they have no other information source about the outside world. In particular we will assume that they have no notion of time but a clock signal which is provided by the pirate. From a practical point of view, Guardian Angels should be some smart cards.

Obviously, Guardian Angels can play the role of a distributed trusted third party. They can establish secure channels between them, providing confidentiality, integrity, authentication, and sequentiality. The originality of our model is that those channels require the cooperation of Pirates, so we cannot assume timeliness. Hence the inter-Guardian angels communication channels can be assumed to correspond to the model of Section 2.

3.3 Fair Exchange with Two Guardian Angels

Let us denote P the Pirates and G the Guardian Angels. In this section we focus on the fair exchange problem between P_A and P_B using G_A and G_B. Note that this can be adapted in a straightforward way for the exchange problem between P_A and G_B or between G_A and G_B.

If P_A and P_B wish to exchange k_A and k_B, they ask their Guardian Angels for assistance. Then, G_A simply sends k_A to G_B through their secure channel, and G_B transmits k_B to G_A. After the exchange itself, they perform a synchronization by using the KiT Protocol. Then, if the protocol succeeded G_A and G_B disclose the received items to the Pirates. This protocol is illustrated on Fig. 2.

To keep our protocol easily readable, some important issues are not depicted on Fig. 2. Firstly, we assume that the Guardian Angels have means to check

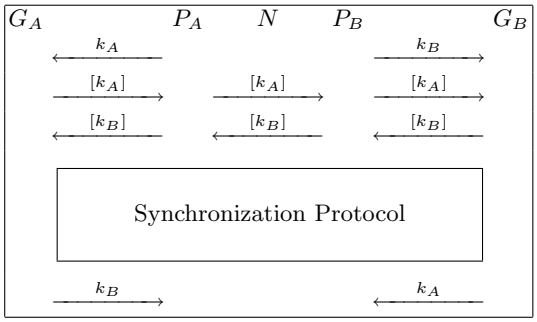

Fig. 2. Fair Exchange Protocol

that the received item is the expected one: Pirates can sent the descriptions of the items to their Guardian Angel. Secondly, since items have an arbitrary size and that Guardian Angels are assumed to have a limited storage facility, we assume that the Guardian Angels forward the received item by encrypting it on-the-fly with a freshly picked key. The key then takes place of the item in the above protocol. Thirdly, the lack of timeliness in the synchronization should be managed by timeouts. But since Guardian Angels are not assumed to have internal clocks, timeouts should be yield by Pirates. Similarly, the end of the synchronization protocol (at least for the Guardian Angel who sends the very last message) should occur only after the Pirate yields a timeout by using the termination side channel protection of the KiT Protocol since the Pirate can detect his state from the behavior.

We give here the initiator's Fair Exchange and Synchronization programs. We assume that P_A is the initiator of the exchange.

G_A's Synchronization Program

- send a random value C to G_B
- while $C > 0$ do
  - standby {wait for a message or a timeout}
  - if a timeout from P_A is received then the synchronization failed
  - {a message from G_B is received} decrement C
  - if $C = 0$ then the synchronization succeeded
  - send an message to G_B and decrement C
- end while
- standby {wait for a timeout}
- the synchronization succeeded

G_A's Fair Exchange Program

- receive k_A from P_A
- establish a secure channel with G_B through P_A
- send k_A to G_B through the channel

– receive k_B from G_B through the channel
– check k_B, if incorrect, abort
– encrypt k_B with a random secret key K
– send k_B encrypted with K to P_A
– execute the Synchronization Protocol
– if the synchronization failed then abort
– send K to P_A

P_A's Fair Exchange Program

– send k_A to G_A
– forward messages between G_A and P_B for the channels between G_A and G_B
– if timeout then
  • send timeout signal to G_A
  • if receive K then decrypt k_B else the protocol failed

3.4 Analysis of our Protocol

Obviously, our fair exchange protocol inherits Theorems 2 and 5 from the KiT Protocol.

Let us assume that we have a fair exchange protocol. We describe the following synchronization protocol between A and B: they decide to exchange some value. If the exchange succeeds, they enter into a `success` state. Otherwise, they enter into a `failure` state. This is obviously a synchronization protocol which inherits his parameters from the fair exchange protocol.

4 Applying to the Mobile Ad Hoc Networks

4.1 Applications

Current Mobile Networks rely on a heavy fixed infrastructure which connects the users through relays. Installing such an infrastructure is often either too expensive or technically impossible and hence some areas are not reachable. Ad Hoc Wireless Networks mitigate this problem by allowing users to route data through intermediate nodes: the network is furthermore self-organized and rely on any established infrastructure anymore.

In such a network, cryptographic protocols cannot use on-line Trusted Third Party for some obvious reasons. One can think that optimistic protocol may run properly. Besides that using off-line Trusted Third Party does not come up to the Mobile Ad Hoc Networks requirements, we cannot assume that most of the nodes will be honest. Indeed, the nodes have to forward the packets of the other ones to keep alive the community, but they will try to cheat as soon as possible in order to save their battery life since forwarding packets have a substantial cost: nodes will become selfish. Participants will have then to require the Trusted Third Party in each transaction. On the other hand we cannot reasonably use Gradual Fair Exchange Protocols since no assumption have been done on the

computational power of the nodes: the latter could be mobile phones, but also PDAs, laptops, etc.

Additionally, extreme case of fully self-organized networks aim at getting rid of any central service. Our model is then fully relevant to this environment since it achieves probabilistic fairness without Trusted Third Party, assuming only that a secure channel between the Guardian Angels is available. Even if economic and practical aspects are not fully designed yet, it makes sense to imagine the following scenario. Some company builds and sells guardian angels who becomes the virtually distributed TTP. This company (who was the network provider in earlier wireless networks) simply makes community rules and install an accounting service. Guardian Angels are responsible for community rules enforcement and keep local accounting in their lifetime. When they expire, their successor keep track of accountings. Users can thus just buy and plug Guardian Angels into their device in order to control fairness, security, accounting, services,... We can later imagine several Guardian Angels manufacturer with specific trade exchange protocols.

As an application we can exchange an inexpensive service against a micropayment with the protocol of Example 4 with $p = \frac{1}{2}$. The risk for both participants is to loose small valuables, but with a probability bounded by $\frac{1}{2}$ instead of a probability which may be equal to 1.

4.2 Improvements

We would like to notice that, besides the exchanged messages during the synchronization step are empty, the complexity of our protocol can be improved by performing several parallelized Fair Exchanges and by factorizing the synchronization steps: if A and B need to perform a new synchronization despite the previous one is not finished, they can just merge the two KiT protocols into a single one. If they need C_1 remaining messages for the previous protocol to end up and C_2 messages for the new one to end up, they perform a protocol with $\max(C_1, C_2)$ messages. The ith protocol will end up when $C_i = \min(C_1, C_2)$ messages will be exchanged. We have already implemented our protocol in a PDAs network and so we have shown that our protocol is practicable [7]. This way the synchronization protocol induces a constant overhead to serial Fair Exchange Protocol.

Note that we can easily design a protocol which involves one Guardian Angel instead of two. This will be detailed in the full paper version.

5 Conclusion

In this paper, we first recalled the definitions and properties related to Fair Exchange Problem and described the main existent ways to achieve fairness. We introduced then the Synchronization Problem devising such a Gradual Synchronization Protocol and shown that the Fair Exchange Problem can be reduced to the Synchronization Problem in our model based on Chaum *et Al.*'s observer.

We designed in this model a Fair Exchange Protocol which provides arbitrarily low unfairness. Our protocol does not require Trusted Third Party and does not come up to computational power assumptions. Even if there exists a great body of literature on Fair Exchange, our protocol is the first non-optimistic one taking advantage of the presence of security modules. Finally, we analyzed the protocol complexity, the probability of unfairness, but also the probability that an attack could achieve.

We also introduced a communication network model with Pirates and Guardian Angels. This model is well suited to Ad Hoc networks and may deserve future research.

Acknowledgment

We are thankful to the students Jérôme Berclaz and Steve Vaquin for their implementation of the protocol.

The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

References

1. N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. Research Report RZ 2858, IBM Research Division, Zurich, Switzerland, September 1996.
2. N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for multi-party fair exchange. Research Report RZ 2892, IBM Research Division, Zurich, Switzerland, December 1996.
3. N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. In *Proceedings of 4th ACM Conference on Computer and Communications Security*, pages 7–17, Zurich, Switzerland, April 1997. ACM Press.
4. N. Asokan, Victor Shoup, and Michael Waidner. Asynchronous protocols for optimistic fair exchange. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 86–99, Oakland, California, USA, May 1998. IEEE Computer Society Press.
5. N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 591–606, Helsinki, Finland, May 1998. Springer-Verlag.
6. Michael Ben-Or, Oded Goldreich, Silvio Micali, and Ronald L. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1):40–46, January 1990.
7. Jérôme Berclaz and Steve Vaquin. Fair exchange between iPAQs. Semester project, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, http://lasecwww.epfl.ch, February 2002.
8. Ernest F. Brickell, David Chaum, Ivan B. Damgård, and Jeroen van de Graaf. Gradual and verifiable release of a secret. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO'87*, volume 293 of *Lecture Notes in Computer Science*,

pages 156–166, Santa Barbara, California, USA, August 1988. IACR, Springer-Verlag.
9. David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *Advances in Cryptology – CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105, Santa Barbara, California, USA, August 1992. IACR, Springer-Verlag.
10. Richard Cleve. Controlled gradual disclosure schemes for random bits and their applications. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 573–588, Santa Barbara, California, USA, August 1990. IACR, Springer-Verlag.
11. Ivan B. Damgård. Practical and probably secure release of a secret and exchange of signatures. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 200–217, Lofthus, Norway, May 1993. IACR, Springer-Verlag.
12. Tim Dierks and Christopher Allen. The TLS protocol – version 1.0, January 1999.
13. Shimon Even and Yacov Yacobi. Relations amoung public key signature systems. Technical Report 175, Computer Science Department, Technicon, Israel, 1980.
14. Matt Franklin and Michael K. Reiter. Fair exchange with a semi-trusted third party. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 1–5, Zurich, Switzerland, April 1997. ACM Press.
15. Matt Franklin and Gene Tsudik. Secure group barter: Multi-party fair exchange with semi-trusted neutral parties. In Rafael Hirschfeld, editor, *Financial Cryptography – FC'98*, volume 1465 of *Lecture Notes in Computer Science*, pages 90–102, Anguilla, British West Indies, February 1998. IFCA, Springer-Verlag.
16. Felix C. Gärtner, Henning Pagnia, and Holger Vogt. Approaching a formal definition of fairness in electronic commerce. In *Proceedings of the International Workshop on Electronic Commerce – WELCOM'99*, pages 354–359, Lausanne, Switzerland, October 1999. IEEE Computer Society Press.
17. Rachid Guerraoui. Revisiting the relationship between non-blocking atomic commitment and consensus. In Jean-Michel Hélary and Michel Raynal, editors, *Proceedings of the 9th International Workshop on Distributed Algorithms – WDAG'95*, volume 972 of *Lecture Notes in Computer Science*, pages 87–100, Le Mont Saint Michel, France, September 1995. Springer-Verlag.
18. Rachid Guerraoui. Non-blocking atomic commit in asynchronous distributed systems with failure detectors. *Distributed Computing*, 15(1):17–25, February 2002.
19. Jean-Pierre Hubaux, Jean-Yves Le Boudec, Silvia Giordano, and Maher Hamdi. The terminode project: Toward mobile ad-hoc wans. In *Proceedings of the Sixth IEEE International Workshop on Mobile, Multimedia Communications – MoMuC'99*, San-Diego, California, USA, 1999.
20. Jean-Pierre Hubaux, Thomas Gross, Jean-Yves Le Boudec, and Martin Vetterli. Towards self-organizing mobile ad-hoc networks: the terminodes project. *IEEE Comm Mag*, 39(1):118–124, January 2001.
21. Markus Jakobsson. Ripping coins for a fair exchange. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology – EUROCRYPT'95*, volume 921 of *Lecture Notes in Computer Science*, pages 220–230, Saint Malo, France, May 1995. IACR, Springer-Verlag.
22. Steve Kremer, Olivier Markowitch, and Jianying Zhou. An intensive survey of non-repudiation protocols. Technical Report ULB–474, Université Libre de Bruxelles, Bruxelles, Belgium, 2001.
23. Olivier Markowitch. *Les protocoles de non-répudiation*. PhD thesis, University of Bruxelles, Bruxelles, Belgium, January 2001.

24. Olivier Markowitch and Shahrokh Saeednia. Optimistic fair exchange with transparent signature recovery. In *Financial Cryptography – FC'01*, Lecture Notes in Computer Science, Cayman Islands, February 2001. IFCA, Springer-Verlag.
25. Tatsuaki Okamoto and Kazuo Ohta. How to simultaneously exchange secrets by general assumptions. In *Proceedings of the 2nd ACM Conference on Computer and Communications Security*, pages 184–192, Fairfax, Virginia, USA, November 1994. ACM Press.
26. Henning Pagnia, Holger Vogt, and Felix C. Gärtner. Fair exchange. *The computer Journal*, 46(1):55–75, January 2003.
27. Michael O. Rabin. Transaction protection by beacons. *Journal of Computer and System Science*, 27(2):256–267, October 1883.
28. Indrakshi Ray and Indrajit Ray. An optimistic fair exchange e-commerce protocol with automated dispute resolution. In Kurt Bauknecht, Sanjay Kumar Madria, and Günther Pernul, editors, *Electronic Commerce and Web Technologies – EC-Web 2000*, volume 1875 of *Lecture Notes in Computer Science*, pages 84–93, London, United Kingdom, September 2000. DEXA Association, Springer-Verlag.
29. Matthias Schunter. *Optimistic Fair Exchange*. PhD thesis, University of Saarlandes, Saarbruken, Germany, October 2000.
30. Paul Syverson. Weakly secret bit commitment: Applications to lotteries and fair exchange. In *Proceedings of the 11th Computer Security Foundations Workshop – PCSFW*, pages 2–13, Rockport, Massachusetts, USA, June 1998. IEEE, IEEE Computer Society Press.
31. Tom Tedrick. How to exchange half a bit. In David Chaum, editor, *Advances in Cryptology – CRYPTO'83*, pages 147–151. Plenum Press, August 1983.
32. Tom Tedrick. Fair exchange of secrets. In George Robert Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 434–438, Santa Barbara, California, USA, August 1985. IACR, Springer-Verlag.
33. Holger Vogt. Asynchronous optimistic fair exchange based on revocable items. In Rebecca N. Wright, editor, *Financial Cryptography – FC'03*, volume 2742 of *Lecture Notes in Computer Science*, Le Gosier, Guadeloupe, French West Indies, January 2003. IFCA, Springer-Verlag.
34. Holger Vogt, Felix C. Gärtner, and Henning Pagnia. Supporting fair exchange in mobile environments. *Journal on Mobile Networks and Applications*, 8(2):127–136, April 2003.
35. Holger Vogt, Henning Pagnia, and Felix C. Gärtner. Using smart cards for fair exchange. In L. Fiege, G. Mühl, and U. Wilhelm, editors, *Proceedings of 2nd International Workshop on Electronic Commerce – WELCOM 2001*, volume 2232 of *Lecture Notes in Computer Science*, pages 101–113, Heidelberg, Germany, November 2001. Springer-Verlag.
36. Jianying Zhou, Robert Deng, and Feng Bao. Evolution of fair non repudiation with TTP. In Josef Pieprzyk, Reihaneh Safavi-Naini, and Jennifer Seberry, editors, *Proceedings of Fourth Australasian Conference on Information Security and Privacy – ACISP 1999*, volume 1587 of *Lecture Notes in Computer Science*, pages 258–269, Wollongong, Australia, April 1999. Springer-Verlag.
37. Jianying Zhou, Robert Deng, and Feng Bao. Some remarks on a fair exchange protocol. In Hideki Imai and Yuliang Zheng, editors, *Proceedings of Third International Workshop on Practice and Theory in Public Key Cryptography – PKC 2000*, volume 1751 of *Lecture Notes in Computer Science*, pages 46–57, Melbourne, Australia, January 2000. Springer-Verlag.