On the Efficiency of Multi-party Contract Signing Protocols

Gerard Draper-Gil[1(✉)], Josep-Lluís Ferrer-Gomila[2], M. Francisca Hinarejos[2], and Jianying Zhou[3]

[1] University of New Brunswick, Fredericton, NB E3B 5A3, Canada
gerard.draper@unb.ca
[2] Universitat de les Illes Balears, 07122 Palma, Spain
{jlferrer,xisca.hinarejos}@uib.es
[3] Institute for Infocomm Research, Singapore 138632, Singapore
jyzhou@i2r.a-star.edu.sg

Abstract. This paper presents an efficiency study of fair exchange protocols for Multi-Party Contract Signing (MPCS) from their architecture point of view, an approach that has not been previously explored. A set of common topologies is presented and defined: ring, star sequential and mesh. Some common terms and notions, such as round and message, are defined according to the topology where they are applied. The suitability of such common terms to measure the efficiency of the protocols is discussed. Finally, we present the design of optimal asynchronous optimistic MPCS protocols for different topologies and evaluate them under the unified definition/criterion of the efficiency parameters. These results are important to support secure and efficient online business which is part of our efforts for building secure and smart cyber society.

Keywords: Multi-party contract signing · Contract signing efficiency · Abuse-freeness

1 Introduction

The objective of a Multi-Party Contract Signing (MPCS) protocol is to allow a set of participants P_i $(2 < i < N)$ to exchange a valid signature on a contract C, without any of them gaining advantage over the others. We can describe the protocol as an application of fair exchange: N parties want to sign a contract C, but none of the participants is willing to give his signature away unless he has an assurance that he will receive all the other participants' signatures.

Most of the solutions we can find in the literature for MPCS protocols are based on the existence and possible involvement of a Trusted Third Party (TTP). The TTP is an external entity that assures the protocol is executed correctly, providing the participants who contact it with evidence proving the state of the execution. In fact, even for two-party contract signing protocols there is a consensus that solutions without a TTP are not practical. One step further is

J. Lopez and C.J. Mitchell (Eds.): ISC 2015, LNCS 9290, pp. 227–243, 2015.
DOI: 10.1007/978-3-319-23318-5_13

to decide if this TTP will intervene in every protocol run (inline or online TTP) or only in case of exception (offline TTP, also called optimistic solutions). The majority of scientific proposals tend to use offline TTPs, where the TTP is only involved if a dispute arises, which is expected to be an exceptional case.

We can find different proposals for MPCS in the scientific literature [3,4,8, 12–14]. Some of them claim to propose optimal solutions or define lower-bounds to design MPCS protocols [8,12,13], but the different criteria applied to define requirements like fairness, or terms like round, step, etc., makes it difficult to assert the validity of such optimal solutions. Moreover, even though we can use different topologies to design MPCS protocols, none of these solutions contemplates the influence of the topology over the efficiency of the final result.

The objective of this paper is to design asynchronous protocols in which N participants sign the same contract C. We choose to design asynchronous protocols instead of synchronous ones, to avoid the problems related to the participant's clock synchronization.

Our Contribution: The contributions of this paper are manifold. First, we discuss the parameters that are generally used to measure the efficiency of MPCS protocols, making clear definitions of each one and defining new ones when the commonly used parameters are not good for measuring efficiency. Second, we describe four of the most common architectures (ring, star, sequential and mesh) and we define them according to the efficiency parameters. Finally, we describe a method to design asynchronous optimistic MPCS protocols, and we propose one as example. We also informally prove that our proposals are optimal, improving the existent proposals of lower-bounds for asynchronous optimistic MPCS protocols.

2 MPCS Requirements

MPCS is a particular case of fair exchange protocols in which we have more than 2 participants and the items to be exchanged are signatures on a contract. Requirements for optimistic fair exchange protocols were defined by Asokan *et al.* [1]: *effectiveness, fairness (strong and weak), timeliness and non-repudiation*, and later, re-formulated by Zhou *et al.* [19]. In this section we will adapt these requirements to the asynchronous optimistic MPCS scenario.

Effectiveness. If all participants in a MPCS protocol behave correctly (and there are no network or system errors), the protocol will finish without the intervention of the TTP.

Strong Fairness. Upon finalization of a MPCS protocol, either all honest participants have the signature from the other participants, or all of them have proof that the signature has been canceled. None of the participants can receive evidence that contradicts the final state of the protocol execution.

Weak Fairness. Upon finalization of a MPCS protocol, either strong fairness is met or all honest participants can prove they have behaved correctly.

Non-repudiation. Upon finalization of a MPCS protocol, none of the participants can deny having participated. In particular, the participants cannot deny having originated (non-repudiation of origin) the signatures exchanged.

Timeliness. Any participant in a MPCS protocol can be sure that the duration of the protocol execution is finite. And once the protocol is finished, any honest participant will maintain the level of fairness obtained.

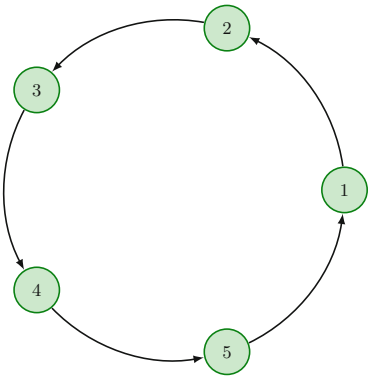
In addition to the requirements stated by Asokan *et al.* [1] and Zhou *et al.* [19], Garay *et al.* [7] introduced abuse-freeness. Its objective is to avoid dishonest participants to misuse the information acquired during the protocol execution (e.g., the commitment to sign a contract from other participants).

Abuse-Freeness. After receiving P_i a partial signature from another participant P_j, the recipient P_i cannot convince others but himself that the partial signature is from the sender P_j.

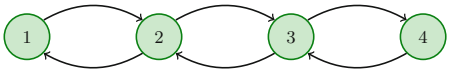
3 Efficiency

It is usually accepted that a protocol is efficient when it makes a *reasonable* use of resources to fulfil its purpose. But we do not have a reference measure to distinguish reasonable from unreasonable, therefore authors usually talk about the efficiency of their solutions compared to others. Most of the papers use the computational power as *the resource* to measure, giving their value of efficiency in terms of number of mathematical operations, but it is not always easy to grasp the real value of these measures.

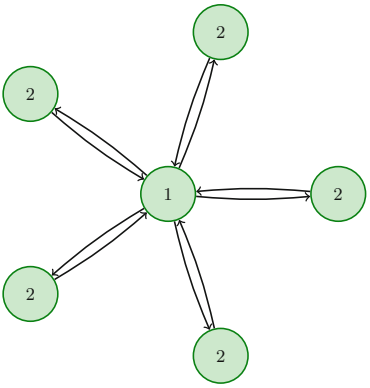
Throughout the solutions found in the literature, authors use the terms 'round' and 'step' without clearly defining them, which often brings on confusion with respect to the metric to be used for its efficiency evaluation, or what they are exactly measuring. Another value typically provided is the number of messages required to complete a protocol execution, but again they fail to give a clear description of it. In our opinion, the term round should not be used for measuring the efficiency of a protocol, but to help in its description. As we will see in Sect. 4, the problem is that rounds in different topologies are not equal, e.g., in a ring topology a round requires the participants to make N transmissions of information (1 per participant). Moreover, in a ring topology the protocol execution must follow a certain order, and this information can be used by the TTP to detect malicious users (see TTP rules for ring topology, in Sect. 7.1), meanwhile in a mesh topology there is no execution order among participants. The use of message as a parameter to measure efficiency has also the same problem: a transmission of information may contain more than one message. In this paper we will take a different approach, we will focus on the participants to measure the protocol efficiency. We will measure the protocol complexity in terms of how many transmissions are required, and how many messages each user has to generate. These terms can later be translated in a time estimation, giving each participant an idea of time, how long will a protocol execution take. Following we make a definition of message and transmission, to clearly state their meaning.



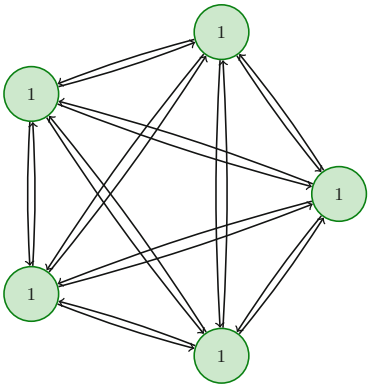
(a) Ring Topology with $N = 5$



(b) Sequential Topology with $N = 4$



(c) Star Topology with $N = 6$



(d) Full Mesh Topology with $N = 5$

Fig. 1. Multi-party contract signature topologies

Definition 1 ***(Transmission)*** *The action of transmitting one or more messages from an originator A to a recipient B.*

Definition 2 ***(Message)*** *A logical set of information sent from an originator A to B, where B can be a set of recipients $\{B_1, ..., B_N\}$.*

4 Topologies

In this section we will define four of the most used topologies when designing MPCS protocols. For each topology we will define the meaning of round, and we will calculate the number of transmissions required to complete a round, assuming N participants $\{P_1, ..., P_N\}$.

4.1 Ring

In a ring topology the transmissions occur between two adjacent nodes P_i and $P_{(i+1)}$, until the execution flow reaches P_N, whose transmission recipient is P_1, the initiator node. The ring architecture executes the transmissions on a serial basis. In Fig. 1a we have depicted a complete round of a ring topology.

Definition 3 ***(Ring-Round.)*** *A round begins when P_1 executes a transmission to P_2, then P_2 transmits to P_3, ..., and ends when P_1 receives the transmission from P_N, closing the ring.*

A complete *ring-round* requires N transmissions, and generates information on the execution order that can be used by the TTP to detect attempts of misbehaviour.

4.2 Sequential

In a sequential topology the transmissions are executed on a serial basis. The protocol execution flows from P_1 to P_N, and back to P_1, going through all the participants in between. In Fig. 1b we have a complete round execution of a sequential topology depicted.

Definition 4 ***(Sequential-Round.)*** *A round is started by the participant P_1, transmitting one or more messages to P_2. The transmissions continue through all the participants in a certain order (e.g., incrementing the subindex i: $P_i, P_{(i+1),..}$), until it reaches P_N, who reverses the order transmitting to $P_{(N-1)}$, who executes a transmission to $P_{(N-2)}$, etc. The round ends when P_1 receives a transmission from P_2.*

A complete *sequential-round* requires $2(N-1)$ transmissions. It also generates information on the execution order of the transmissions.

4.3 Star

In a star topology the transmissions between participants are routed through a central node/participant. The central node P_1 receives all the transmissions from the participants P_j ($j \in [2..N]$), and then P_1 returns to each P_j the corresponding messages. Figure 1c depicts a complete round execution of star topology depicted.

Definition 5 ***(Star-Round.)*** *A round begins when the initiator P_1 transmits some message or messages to all P_j ($j \in [2..N]$), and ends when P_1 has received the corresponding transmission from each P_j. Alternatively, the round can be started by all P_j ($j \in [2..N]$) transmitting to P_1, and finish when each P_j has received P_1's transmission.*

The star topology only generates information about who initiated the *star-round*. It requires $2(N-1)$ transmissions.

4.4 Mesh

In a mesh topology the transmissions are executed on a parallel basis. Each P_i, with $1 \leq i \leq N$ will execute a transmission to each P_j, with $j \in [1..N]$, $j \neq i$. In Fig. 1d we have a complete round execution of mesh topology.

Definition 6 ***(Mesh-Round.)*** *A round begins when P_i, with $1 \leq i \leq N$ executes a transmission to each P_j, with $j \in [1..N]$, $j \neq i$. The round will end when every participant has received a transmission from the other $N-1$ participants.*

A complete *mesh-round* requires $N(N-1)$ transmissions, and it does not generate additional information: the participants are not ordered.

5 Related Work

Baum-Waidner *et al.* propose in [3] a MPCS protocol that requires $N+1$ rounds without abuse-freeness, and $N+3$ with it (optimistic case). Their protocol uses a mesh topology, but they also describe how to transform it into a protocol with star topology. The number of rounds and messages required for each protocol are presented as a function of the number of dishonest parties t. But it is not clear the usefulness of it, because we cannot know the number of dishonest participants beforehand. In [2] Baum-Waidner presents an optimization of the previous proposal [3] where they assume a number of dishonest participants $t < (N-1)$, but it is difficult to see the utility of this enhancement because, as we just said, we cannot predict the number of dishonest parties beforehand.

Khill *et al.* [11] propose a protocol for multi-party fair exchange using a ring topology. They affirm that the ring model is more efficient than the full mesh topology. Their protocol consists on 3 rounds and $3N$ messages in the optimistic case, and $7N$ messages in the worst case. A serious drawback of the protocol is that it is supposed that the TTP broadcasts its decision to all parties. This assumption is dangerous, because the channels can be resilient, but some party can be unreachable for other reasons.

Chadha *et al.* [4] analyze formally two previous works: Garay *et al.* [8] and Baum-Waidner *et al.* [3]. They focus on three properties: fairness, timeliness and abuse-freeness. They conclude that the proposal of Baum-Waidner *et al.* [3] has no security problems. On the other hand, they prove that the proposal of Garay *et al.* [8] presents a security flaw when $N = 4$: it is not fair. Mukhamedov *et al.* [15] prove that Chadha *et al.*'s [4] proposal (a fix to [8]) is also flawed. An interesting issue discussed in that paper [15] is the abort chaining problem (or resolve impossibility): non-honest parties can group together to propagate a TTP's abort decision. In fact, this is a way to prove the necessity of more than $(N-1)$ rounds for N users (in order to avoid the abort chaining attack). The abort chaining attack is a sequence of requests made by dishonest participants trying to force the TTP to deliver cancel evidence, even though some other honest participant may have already signed the contract.

Ferrer *et al.* present in [5] an optimal solution for asynchronous optimistic MPCS with a ring topology, requiring quasi N rounds (more than $N-1$ but

less than N) for N parties. Their solution meets the following requirements: effectiveness, weak fairness, timeliness, non-repudiation and verifiability of TTP. The proposal takes into account the abort chaining problem.

In [14] Mukhamedov and Ryan criticize the work of Baum-Waidner *et al.* [3] alleging that they use a non-standard notion of signed contract and they need $(N+1)N(N-1)$ messages, more than in the solution provided in [14], $N(N-1)(\lceil N/2 \rceil + 1)$. In [14] fairness, abuse-freeness and timeliness are considered. They use a *hybrid* topology, a mixture of sequential and mesh, where the participants are ordered. The protocol needs $(\lceil N/2 \rceil + 1)$ rounds, and authors observe that it is not coherent with Garay's Theorem of [7], but they argue that the concept of round, used in different papers, is not clear.

Mauw *et al.* [13] use the concept of abort chaining of Mukhamedov *et al.* [15] to derive a lower bound on the number of messages in MPCS protocols. The authors model contract signing protocols as sequences of numbers. They consider three security requirements: fairness, timeliness and abuse-freeness (but they affirm that the latter "will not play a role in our observations on message minimality").

Zhang *et al.* propose in [18] a game-based verification of MPCS protocols. They assume that MPCS protocols have to satisfy three properties: fairness, timeliness and abuse-freeness. They analyze the protocols provided in [13,16], proving the latter to be flawed for 3 signers and proposing a fix. Authors assume that "once having contacted TTP by initiating a sub-protocol, the signers would never be allowed to proceed the main protocol any more", but we cannot forbid a dishonest party to contact the TTP and proceed with the main protocol.

Following a similar reasoning than [13], Kordy *et al.* [12] propose protocols derived from sequences of numbers. They consider the following requirements: fairness, timeliness and abuse-freeness. An example with $N = 3$ results in a protocol (sequential topology) with 18 messages, that can be converted to 12 messages. They cannot provide closed expressions for all values of N, and only provide upper bounds.

6 MPCS Protocols Overview

This section presents a simple method to design asynchronous optimistic MPCS protocols. An asynchronous optimistic MPCS protocol will be composed of two sub-protocols: the exchange sub-protocol and the resolution sub-protocol. If all participant behave correctly and there are no network errors, only the exchange sub-protocol will be executed and the TTP will not intervene. As defined in Sect. 2, the MPCS protocol will meet the security requirements for asynchronous optimistic MPCS: effectiveness, weak fairness, non-repudiation and timeliness. And since it is a contract signing protocol, it should also consider the abuse-freeness requirement.

The MPCS protocol execution will follow a simple principle: in turns, the participants will exchange series of commitments to sign the contract C, until they have enough evidence to consider the contract signed. The commitments

are signatures on the contract C and an index k. What is a "turn" or what is "enough evidence" will be determined by the topology of the protocol (ring, star, etc.).

In Sect. 7 we have an example of an asynchronous optimistic MPCS protocol using a ring topology. Along the rest of the paper, we will use the following notation:

- $\mathbf{N}$ Number of participants.
- $\mathbf{P_i}$ Participant i, $1 \leq i \leq N$.
- $\mathbf{Tx_{(r,i)}}$ Transmission generated by the participant i, during round r.
- $\mathbf{C}$ Contract to be signed.
- **CID** unique Contract IDentifier. A random number used to uniquely identify a protocol execution.
- $\mathbf{h(M_i)}$ Hash Function of message M_i.
- $\mathbf{S_j[M_i]} = \mathbf{SK_j[h(M_i)]}$ j's Digital Signature on M_i (where SK_j is j's private key).

We assume that the contract C includes the necessary information, as the identity of the participants, the TTP, the number N of participants, etc.

As regards the communications channels we make some usual assumptions ([1,6]):

- channels among participants P_i are unreliable, the messages can be delayed or lost.
- channels among participants P_i and the TTP are resilient, the messages can be delayed but not lost.

To meet the abuse-freeness requirement we can use signature schemes like Designated Verifier Signatures (DVS) presented by Jakobsson *et al.* [10], Multi DVS (MDVS) [17], Private Contract Signatures (PCS) introduced by Garay *et al.* [7] or the Ambiguous Optimistic Fair Exchange (AOFE) scheme from Huang *et al.* [9]. In essence, these signature schemes allow the participants to generate a "weak" signature as commitment (partial signature), that can only be verified by the intended recipient (or recipients in the case of MDVS). Once all commitments are exchanged, they can generate a signature that can be verified by third parties (full signature).

6.1 The TTP

The TTP is a third-party that assures the fairness of the protocol providing the participants with proof of the protocol execution state. When a participant does not receive a signature expected, either because an error occurred or because a misbehaving participant, he can send a resolution request to the TTP. The TTP will answer with a canceled or a signed token.

To solve the resolution requests (Table 1) the TTP follows a set of rules. These rules are based on a group of variables the TTP updates on every request received, indicating the state of a protocol execution. Following we have this group of variables, their definition, and some notation used along the rules definition.

- $\overline{X}_N = \{P_1, ..., P_N\}$ set of participants in a MPCS.
- $\overline{XR}$ set of participants who already requested resolution.
- $\overline{XC}$ set of participants who have received a canceled token from the TTP.
- $\overline{XS}$ set of participants who have received a signed token from the TTP.
- $\overline{TxR}$ set of transmissions $Tx_{(r',i')}$ received by the TTP.
- $\overline{PC}$ set of participants that are allowed to cancel the contract signature.
- *canceled* boolean value stating that the contracting protocol execution has been canceled if its value is true.
- *signed* boolean value stating that the contracting protocol execution has been finished (signed) if its value is true.

The rules are the same for all the protocols, but there are some particularizations depending on the topology, that we will explain in the example of MPCS protocol (see Sects. 7.1 and 7.2). Following we have the common set of rules that the TTP will follow to solve the resolution requests (the term *x-round* refers to the particular round of each topology):

RULE 0 (R0). The TTP will only accept one resolution request per participant P_i: if $P_i \in \overline{XR}$, the TTP will dismiss the request.

RULE 1 (R1). If the TTP receives a request from $P_i \in \overline{PC}$ during *x-round* $r = 1$, and the execution has not been previously finished (*signed=true*) by other party, the TTP will cancel it and send a canceled token to P_i.

RULE 2 (R2). If the TTP receives a request from P_i during *x-round* $r > 1$, and the execution has not been previously canceled by other party, the TTP will finish it (*signed=true*) and send a signed token to P_i.

RULE 3 (R3). If the TTP receives a request from P_i during *x-round* $r \geq 1$, and the execution has been previously finished (*signed=true*) by other party, the TTP will send a signed token to P_i.

RULE 4 (R4). If the TTP receives a request from P_i during *x-round* $r > 1$, and the execution has been previously canceled (*canceled=true*) by other party, the TTP will check the previously received requests. If the TTP can prove that all previous requestors cheated, it will change the protocol status from canceled to finished, and deliver the signed token to P_i. Otherwise the TTP will send a cancel token to P_i.

7 Asynchronous Optimistic MPCS Protocols

In this section we present a set of asynchronous optimistic MPCS protocols, one for each topology (ring, sequential, star and mesh) described in Sect. 4. All

Table 1. Resolution sub-protocol for all topologies

MPCS resolution sub-protocol
$P_i \rightarrow$ TTP: $CID, C, r, Tx_{(r,i)}, S_{P_i}[CID, C, r, Tx_{(r,i)}]$
if the TTP decides canceled
$P_i \leftarrow$ TTP: Cancel Token
else
$P_i \leftarrow$ TTP: Signed Token
Cancel Token: $S_{TTP}[CID, C, r, CANCELED]$; where CANCELED is a string
Signed Token: $S_{TTP}[CID, C, \overline{m_{(k,i)}}_N]$

protocols meet the requirements defined in Sect. 2: effectiveness, weak fairness, timeliness, non-repudiation, and abuse-freeness. The examples presented in this section assume the use of the Private Contract Signature (PCS) scheme ([7]) to meet the abuse-freeness requirement. We could replace the PCS signature scheme for any other mentioned in the precious section, and the protocol would still be valid (TTP rules, number of transmissions, security requirements, etc.), according we make the necessary modifications on the content of each transmission $Tx_{(r,i)}$.

7.1 An Asynchronous Optimistic MPCS Protocol Using Ring Topology

This protocol is based on the optimistic MPCS protocol from Ferrer-Gomila *et al.* [5], where the authors present a solution with *quasi* N rounds (more than $N-1$ but less than N) for N parties, meeting the security requirements (Sect. 2). Table 2 shows the exchange sub-protocol execution using the PCS signature scheme, following the nomenclature from Garay *et al.* [7] for the PCS.

In every turn each participant generates a commitment, a private signature (PCS), for each of the other participants in the protocol execution. The index of the commitments, k, is decremented by one every time a participant receives all k-commitments from the other participants (see Table 3 for values of k in a ring protocol). From then, the participants generate $(k-1)$-commitments, until again a participant receives all $(k-1)$-commitments and he decrements its value again. This process is repeated until k reaches the value -1. When the index k reaches the value 0, the commitments will be generated on the contract C, without index. The next iteration, when k is -1, the participants will start to transmit the full signature on the contract C, final evidence that the protocol has finished successfully. In Table 4 we can see an example of a complete execution for $N = 3$.

All participants except P_N can cancel the protocol, therefore we have that in TTP's rule R1 $\overline{PC} = \{P_1, ..., P_{(N-1)}\}$. When P_N receives the first transmission $Tx_{(1,(N-1))}$ he already has evidence that proves that all other participants are willing to sign the contract. If he does not want to sign the contract, he only

Table 2. Asynchronous optimistic MPCS protocol with ring topology and PCS signature scheme

MPCS protocol with ring topology

for $r = 1$

for $i = 1$ **to** N: $P_i \rightarrow P_{(i+1)}$ $Tx_{(1,i)}$

$PCS_i((C,k),P_j,TTP) \quad \forall j \in [1..N] \setminus [i]$

$PCS_{(i-1)}((C,k),P_j,TTP) \; \forall j \in [1..N] \setminus [i,(i-1)]$

$\vdots$

$PCS_1((C,k),P_j,TTP) \quad \forall j \in [1..N] \setminus [i,(i-1),...1]$

for $r = 2$ **to** $(\mathbf{N-1})$

for $i = 1$ **to** N: $P_i \rightarrow P_{(i+1)}$ $Tx_{(r,i)}$

$PCS_i((C,k),P_j,TTP) \quad \forall j \in [1..N] \setminus [i]$

$PCS_{(i-1)}((C,k),P_j,TTP) \; \forall j \in [1..N] \setminus [i,(i-1)]$

$\vdots$

$PCS_{(i-K)}((C,k),P_j,TTP) \forall j \in [1..N] \setminus [i,(i-1),...(i-K)]$

for $r = N$

for $i = 1$ **to** $(N-1)$: $P_i \rightarrow P_{(i+1)}$ $Tx_{(N,i)}$

$S - Sign_j(C) \quad \forall j \in [1..N] \setminus [2,..(i+1)]$

$K = N - 2$

$PCS_i(C,P_j,TTP)$, Private Contract Signature of P_i over C for P_j with Trusted Third Party TTP

$S - Sign_i(C)$ Universally verifiable signature of P_i over C

Operations are mod, e.g., $P_i \rightarrow P_{(i+1)}$ when $i = N$ is $P_N \rightarrow P_1$

needs to discontinue the protocol execution. In a protocol with ring topology, TTP's rule R4 states:

- if $\exists\, Tx_{(r',i')} \in \overline{TxR}$ / $(r' = r)$ or $(r' = r-1$ and $i' > i)$, the TTP will send a cancel token to P_i to maintain fairness for the previous honest requestors.
- if $\forall\, Tx_{(r',i')} \in \overline{TxR}$ / $r' < r-1$, then $P_{i'}$ cheated.
- if $\forall\, Tx_{(r',i')} \in \overline{TxR}$ / $r' = r-1$ and $i' < i$, then $P_{i'}$ cheated.

Notice that the TTP's rule R4 for a ring topology uses the information generated by the protocol flow (when comparing the index i with i'), the execution order to detect cheating participants.

The *cancel Token* ($S_{TTP}[CID,C,r,CANCELED]$) is evidence provided by the TTP proving the contract signature has been canceled. It is the TTP's universally verifiable signature on the unique Contract Identifier CID, the contract C itself, the round number r in which the request was sent to the TTP (it can be used later to prove the validity of the assertion), and a value indicating the final state of the protocol execution.

The *signed Token* is evidence provided by the TTP proving the contract is signed. It will depend on the signature scheme used. In the case of the PCS

Table 3. Value of k according to the round r and the participant i for an asynchronous optimistic MPCS protocol with ring topology

r	i	k
1	$1 \leq i \leq (N-1)$	$k = (N-2)$
1	$i = N$	$k = (N-3)$
2	$1 \leq i \leq (N-2)$	$k = (N-3)$
2	$(N-1) \leq i \leq N$	$k = (N-4)$
3	$1 \leq i \leq (N-3)$	$k = (N-4)$
3	$(N-2) \leq i \leq N$	$k = (N-5)$
⋮		
(N-2)	$1 \leq i \leq 2$	$k = 1$
(N-2)	$3 \leq i \leq N$	$k = 0$
(N-1)	$i = 1$	$k = 0$
(N-1)	$2 \leq i \leq N$	$k = -1$
N	$1 \leq i \leq (N-1)$	$k = -1$

and ambiguous signatures, both schemes have a method to transform the partial signatures in full signatures. Therefore the signed token will be the TTP's full signature on the CID, C, the round r, and the partial signatures converted into full signatures.

Lemma 1. *All asynchronous optimistic MPCS protocols with ring topology, meeting timeliness, require at least* $(N+1)(N-1)$ *transmissions to be fair.*

Proof. We will prove it by contradiction. Assume that $(N+1)(N-1)-1$ transmissions are enough (we eliminate the last transmission: $Tx_{(N,(N-1))}$). It means that P_N has all the evidence when he receives $Tx_{(N-1),(N-1)}$. Now we will construct the abort-chaining attack.

Let us suppose $P_{(N-1)}$ sends a resolution request claiming he has sent $Tx_{(1,(N-1))}$ but he has not received $Tx_{(2,(N-2))}$. He is the first to contact the TTP, therefore the TTP will apply rule R1, canceling the protocol and delivering a canceled token to $P_{(N-1)}$.

Next, $P_{(N-2)}$ sends a resolution request claiming he has sent $Tx_{(2,(N-2))}$ but he has not received $Tx_{(3,(N-3))}$. The TTP will apply rule R4 ($r^{'} = r-1$ and $i^{'} > i$) and it will send a canceled token to $P_{(N-2)}$.

Following, $P_{(N-3)}$ sends a resolution request claiming he has sent $Tx_{(3,(N-3))}$ but he has not received $Tx_{(4,(N-4))}$. The TTP will apply rule R4. This time, the TTP will detect that $P_{(N-1)}$ $((r^{'} = 1) < (r-1=2))$ cheated, but to maintain fairness for $P_{(N-2)}$ $(1 = 2-1$ and $(N-1) > (N-2))$, it will send a canceled token to $P_{(N-3)}$.

We can continue this abort-chaining attack, until P_2 sends a resolution request claiming he has sent $Tx_{((N-2),2)}$ but he has not received $Tx_{((N-1),1)}$. Applying

R4, the TTP detects that P_4 cheated, but to maintain fairness for P_3 it sends a canceled token to P_2.

Finally, P_1 sends a resolution request claiming he has sent $Tx_{((N-1),1)}$ but he has not received $Tx_{((N-1),N)}$. Again, the TTP will apply rule R4, and deliver a canceled token to P_1 to maintain fairness for P_2 (the TTP can prove that $\{P_3, P_4, ..., P_{(N-2)}, P_{(N-1)}\}$ have cheated). In this scenario, an honest P_N may have received all evidence, but an honest P_1 has a canceled token from the TTP ($Tx_{((N-1),N)}$ may be lost due to a network error), therefore fairness is broken.

But if we add another transmission, $Tx_{(N,(N-1))}$, we can avoid the abort chaining attack. Continuing the previous execution, with the additional transmission, we have two possibilities:

- If P_1 is honest, he will not continue with the protocol execution, therefore P_N will send a resolution request to the TTP claiming the missing evidence. The TTP will be able to prove that P_2 cheated, but again, to maintain fairness for the honest participants it will send a canceled token to P_N. Both P_1 and P_N, honest, will have a canceled token.
- If P_1 is dishonest, and all other dishonest participants continue with the protocol execution, P_N will receive $Tx_{(N,(N-1))}$, therefore he will have evidence the contract has been signed.

In both cases weak fairness is met. Therefore we can affirm that the minimum number of transmissions that an asynchronous optimistic MPCS protocol with ring topology needs to be fair is $(N+1)(N-1)$.

7.2 An Asynchronous Optimistic MPCS Protocol with Sequential, Star and Mesh Topology

Following the same method we used to design the MPCS protocol with ring topology, we can design a protocol using a sequential, star and mesh topology. Due to the lack of space, we cannot include their full description, but we briefly present their measures of efficiency and the TTP rules.

Sequential: $(N+1)(N-1)$ transmissions are necessary.
In a protocol with sequential topology we have: $\overline{PC} = \{P_1, ..., P_{(N-1)}\}$, i.e. all participants except P_N can cancel the protocol. To apply R4, the TTP follows these statements:

- if $\exists\, Tx_{(r',i')} \in \overline{MR}$ / $(r' = r)$ or $(r' = r-1$ and $i' < i)$, the TTP will send a canceled token to P_i to maintain fairness for the previous honest requesters.
- if $\forall\, Tx_{(r',i')} \in \overline{MR}$ / $r' < r-1$, then $P_{i'}$ cheated.
- if $\forall\, Tx_{(r',i')} \in \overline{MR}$ / $r' = r-1$ and $i' > i$, then $P_{i'}$ cheated.

Star: $(2N-1)(N-1)$ transmissions are necessary.
In an asynchronous optimistic MPCS protocol with star topology, all participants except P_1 can cancel the protocol: $\overline{PC} = \{P_2, ..., P_N\}$, TTP's R1. To apply R4 the TTP follows the next assertions:

Table 4. Example of asynchronous optimistic MPCS protocol, ring topology, PCS signature scheme and $N = 3$

$r = 1$
$P_1 \rightarrow P_2$ $PCS_1((C,1),P_2,TTP)$, $PCS_1((C,1),P_3,TTP)$
$P_2 \rightarrow P_3$ $PCS_2((C,1),P_3,TTP)$, $PCS_2((C,1),P_1,TTP)$
$PCS_1((C,1),P_3,TTP)$
$P_3 \rightarrow P_1$ $PCS_3(C,P_1,TTP)$, $PCS_3(C,P_2,TTP)$
$PCS_2((C,1),P_1,TTP)$
$r = 2$
$P_1 \rightarrow P_2$ $PCS_1(C,P_2,TTP)$, $PCS_1(C,P_3,TTP)$
$PCS_3(C,P_2,TTP)$
$P_2 \rightarrow P_3$ $S-Sig_2$
$PCS_1(C,P_3,TTP)$
$P_3 \rightarrow P_1$ $S-Sig_3$
$S-Sig_2$
$r = 3$
$P_1 \rightarrow P_2$ $S-Sig_1$
$S-Sig_3$
$P_2 \rightarrow P_3$ $S-Sig_1$

- if $\exists\ Tx_{(r',i')} \in \overline{MR}\ /\ r' \geq r-1$, the TTP will send a canceled token to P_i to maintain fairness for the previous honest requesters.
- if $\forall\ Tx_{(r',i')} \in \overline{MR}\ /\ r' < r-1$, then $P_{i'}$ cheated.

Mesh: $N^2(N-1)$ transmissions are necessary.
In the mesh topology, all participants can cancel the protocol: $\overline{PC} = \{P_1, ..., P_N\}$, in TTP's R1. Regarding the detection of cheating users, R4 for a mesh topology states:

- if $\exists\ Tx_{(r',i')} \in \overline{MR}\ /\ r' \geq r-1$, the TTP will send a canceled token to P_i to maintain fairness for the previous honest requesters.
- if $\forall\ Tx_{(r',i')} \in \overline{MR}\ /\ r' < r-1$, then $P_{i'}$ cheated.

8 Protocol Comparison

In Table 5 we can compare the efficiency of our MPCS protocol proposals and some of the most relevant presented in the related work (we eluded proposals that have been proved flawed). In Sect. 3 we have given a definition of message and transmission to avoid confusions (a transmission can include several messages).

Within our solutions, the proposals with ring and sequential architecture are the most efficient, requiring only $(N+1)(N-1)$ transmissions. But comparing the

Table 5. Efficiency of asynchronous optimistic MPCS protocols

Protocol	**Topology**	**Transmissions**$^{(1)}$	**Messages/User**$^{(2)}$
★	Ring	$(N+1)(N-1)$	$(N-1)^2+1$ when $P_i = P_1$
			$(N-2)(N-1)+1$ when $P_i \neq P_1$
★	Sequential	$(N+1)(N-1)$	$\lceil (N-1)/2 \rceil (N-1)+1$ when N is odd
			$\lceil (N-1)/2 \rceil (N-1)-i+2$ when N is even
[14]	Sequential/ Mesh	$(\lceil N/2 \rceil + 1)N(N-1)$	$(\lceil N/2 \rceil + 1)(N-1)+1$
★	Star	$(2N-1)(N-1)$	$(N-2)(N-1)+1$
[3]	Star	$4(N+3)(N-1)$	$(N+2)$
★	Mesh	$N^2(N-1)$	$(N-1)^2+1$
[3]	Mesh	$N(N-1)(N+3)$	$(N+2)$

★ Our proposal.
N, number of participants.
(1) Optimistic case, the TTP does not intervene, and $N-1$ malicious participants assumed.
(2) Number of messages (signatures) generated by each user. $i \in [1..N]$ i^{th} participant

number of messages each user has to generate, we can see that the sequential protocol has advantage, requiring approximately half the messages, which is translated in less computational power needed.

Baum-Waidner *et al.* [3] propose a MPCS protocol that achieves abuse-freeness using *standard* public key cryptography. In their paper, the authors propose a mesh solution and they also explain how to convert it into a star solution. Comparing the efficiency of their solution, we see that our proposals require less transmissions in both cases: ring and mesh architecture. But the number of messages generated by each user is lower in their proposals, due to the use of *standard* public key cryptography.

The proposal from Mukhamedov *et al.* [14] uses an architecture that is a mix of sequential and mesh. Compared with Mukhamedov *et al.* [14], our proposals with ring and sequential architecture require less transmissions. Therefore both should be more efficient than Mukhamedov *et al.*'s. But comparing the number of messages generated by each user, we have that Mukhamedov *et al.*'s proposal requires each user to generates less messages than our solution with ring topology, but more than our sequential proposal. Therefore, requiring less transmissions and less messages, we can affirm that our solution with sequential topology is the most efficient asynchronous optimistic MPCS protocol, setting a lower-bound for the minimum number of transmissions required in MPCS protocols.

With these results we can see that with our new definition of efficiency parameters we are able to compare protocols using different topologies, but only

if they use the same cryptographic techniques (signature schemes, encryption schemes, etc.). A step further into measuring the efficiency of MPCS protocols is to find other parameters that would allow us to compare protocols, even when they use different cryptographic techniques, and to have a better notion of the influence of the architecture, e.g., sending messages in a sequence (ring, sequential) is slower than sending them in parallel (star, mesh), but sending messages in parallel (independent threads) requires more resources than sending messages in a sequence. This topic is part of the future work.

9 Conclusions

Using a common approach for the design of asynchronous optimistic MPCS protocols, we have proposed an asynchronous optimistic MPCS protocol for a ring, sequential, star and mesh topology. Each proposal meets the asynchronous optimistic MPCS protocol requirements, including abuse-freeness. Moreover, the number of transmissions required by each protocol is the minimum needed to maintain fairness, therefore we have also defined a new set of lower-bounds, minimum number of transmissions, for each topology.

As future work we plan to extend this work including hybrid topologies (mesh/sequential, etc.) and a study of different efficiency parameters. Our final goal is to use these results to enhance the efficiency of existing MPCS protocols, and establish a common framework for the evaluation of MPCS topologies.

Acknowledgements. This work is the result of a visiting Ph.D. Fellowship done by Gerard Draper-Gil at the Institute for Infocomm Research (I2R). Jianying Zhou's work is partially supported by A*STAR funded project SPARK-1224104047.

References

1. Asokan, N., Shoup, V., Waidner, M.: Asynchronous protocols for optimistic fair exchange. In: IEEE Symposium on Security and Privacy, S&P 1998, pp. 86–99. IEEE Computer Society, Los Alamitos (1998)
2. Baum-Waidner, B.: Optimistic asynchronous multi-party contract signing with reduced number of rounds. In: Orejas, F., Spirakis, P.G., van Leeuwen, J. (eds.) ICALP 2001. LNCS, vol. 2076, pp. 898–911. Springer, Heidelberg (2001)
3. Baum-Waidner, B., Waidner, M.: Round-optimal and abuse-free optimistic multi-party contract signing. In: Welzl, E., Montanari, U., Rolim, J.D.P. (eds.) ICALP 2000. LNCS, vol. 1853, pp. 524–535. Springer, Heidelberg (2000)
4. Chadha, R., Kramer, S., Scedrov, A.: Formal analysis of multi-party contract signing. In: Proceedings of the 17th IEEE workshop on Computer Security Foundations, 28–30 June 2004, pp. 266–279. IEEE Computer Society, Washington, DC, Pacific Grove (2004)
5. Ferrer-Gomila, J.L., Payeras-Capellà, M.M., Huguet-Rotger, L.: Optimality in asynchronous contract signing protocols. In: Katsikas, S.K., López, J., Pernul, G. (eds.) TrustBus 2004. LNCS, vol. 3184, pp. 200–208. Springer, Heidelberg (2004)

6. Ferrer-Gomilla, J.L., Onieva, J.A., Payeras, M., Lopez, J.: Certified electronic mail: properties revisited. Elsevier Comput. Secur. **29**, 167–179 (2010)
7. Garay, J.A., Jakobsson, M., MacKenzie, P.D.: Abuse-free optimistic contract signing. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 449–466. Springer, Heidelberg (1999)
8. Garay, J.A., MacKenzie, P.D.: Abuse-free multi-party contract signing. In: Jayanti, P. (ed.) DISC 1999. LNCS, vol. 1693, pp. 151–166. Springer, Heidelberg (1999)
9. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Ambiguous optimistic fair exchange. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 74–89. Springer, Heidelberg (2008)
10. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated verifier proofs and their applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
11. Khill, I., Kim, J., Han, I., Ryou, J.: Multi-party fair exchange protocol using ring architecture model. Elsevier Comput. Secur. **20**, 422–439 (2001)
12. Kordy, B., Radomirovic, S.: Constructing optimistic multi-party contract signing protocols. In: Proceedings of the 25th Computer Security Foundations Symposium, CSF 2012, pp. 215–229. IEEE Computer Society, Los Alamitos (2012)
13. Mauw, S., Radomirovic, S., Dashti, M.T.: Minimal message complexity of asynchronous multi-party contract signing. In: Proceedings of the 2009 22nd IEEE Computer Security Foundations Symposium, CSF 2009, pp. 13–25. IEEE Computer Society, Washington, DC (2009)
14. Mukhamedov, A., Ryan, M.D.: Improved multi-party contract signing. In: Dietrich, S., Dhamija, R. (eds.) FC 2007 and USEC 2007. LNCS, vol. 4886, pp. 179–191. Springer, Heidelberg (2007)
15. Mukhamedov, A., Ryan, M.D.: Resolve-impossibility for a contract-signing protocol. In: Proceedings of the 19th IEEE Workshop on Computer Security Foundations, CSFW 2006, pp. 167–176. IEEE Computer Society, Washington, DC (2006)
16. Mukhamedov, A., Ryan, M.D.: Fair multi-party contract signing using private contract signatures. Elsevier Inf. Comput. **206**, 272–290 (2008)
17. Tian, H.: A new strong multiple designated verifiers signature. Int. J. Grid Util. Comput. **3**(1), 1–11 (2012)
18. Zhang, Y., Zhang, C., Pang, J., Mauw, S.: Game-based verification of multi-party contract signing protocols. In: Degano, P., Guttman, J.D. (eds.) FAST 2009. LNCS, vol. 5983, pp. 186–200. Springer, Heidelberg (2010)
19. Zhou, J., Deng, R., Bao, F.: Some remarks on a fair exchange protocol. In: Imai, H., Zheng, Y. (eds.) PKC 2000. LNCS, vol. 1751, pp. 46–57. Springer, Heidelberg (2000)