Simulation and Analysis of Multi-Party Contract Signing Protocol

Yadana Htun, and Win Zaw

Abstract—A contract signing protocol allows a set of participants to exchange messages with each other with a view to arriving in a state in which each of them has a pre-agreed contract text signed by all the others. There are multiple negotiation styles ranging from two-party to multi-party contract signing protocol. A number of two-party contract signing protocols have been proposed with various features. Nevertheless, in some applications, a contract may need to be signed by multiple parties. In this paper, the current main analysis techniques using Colored Petri Nets (CP-nets) for simulation and analysis of multi-party contract signing protocol are introduced. An important property of contract signing protocols is fairness: no participant should be left in the position of having sent another participant his signature on the contract, but not having received signatures from the other participants. Based on the techniques, a new method using CP-Nets for the simulation and analysis of multi-party contract signing protocol is presented. Furthermore, the automated analysis tools CPN Tools is used to analyze the fairness property. Simulation can be used for both validation and performance analysis, while state-space analysis can be used to discover anomalies in various multi-party negotiation protocols. It shows how such a protocol can be modeled as a colored Petri net (CPN) and simulated using CPN Tools.

Keywords—Contract signing protocol, fairness property, state-space analysis, colored Petri net (CP-nets).

I. INTRODUCTION

Contracts are documents that states the mutual obligations and authorizations that reflect the agreements between two trading partners [1]. An e-contract is a contract modeled, specified and enacted by a software system. Contract signing is a fundamental service for business transactions, and has been well practiced in the traditional paper-based business model [2]. Nowadays, it is necessary to find appropriate mechanisms for contract signing in the digital world. Fair exchange is a problem of exchanging data in a way that guarantees either all participants obtain what they want, or none does. From a designing point of view, contract signing is a particular form of fair exchange, in which the parties exchange commitments to a contract. If several signers are involved, then it is a multi-party contract signing protocol (MPCS).

With the rapid growth of security applications, network security property has become an important issue, and security protocols are the basis of security in networks. Therefore, it is essential to ensure these protocols correctly. Unfortunately, it is difficult to design a robustness and effective security protocol for networks. Not only because of the characteristics of networks, but also because good analysis techniques are lacking.

In this paper, the current main analysis techniques using Colored Petri Nets (CP-nets) [3] for simulation and analysis of multi-party contract signing protocol are introduced. An important property of contract signing protocols is fairness: no participant should be left in the position of having sent another participant his signature on the contract, but not having received signatures from the other participants. Based on the techniques, a new method using CP-Nets for the simulation and analysis of multi-party contract signing protocol is presented. Furthermore, how to use the automated analysis tools CPN Tools [4] is shown in the construction and model checking of the net models. Model checking is performed at last. In the model checking, two methods are used. One exploits the provided state space exploration functions and another is simulation implementation.

The paper is organized as follows: in Section 2, review the previous work related to contract signing, outline the properties to be satisfied when designing an optimistic contract signing protocol, and give explicit definitions for some terms used along the descriptions of these protocols. In Section 3, describe an optimal multi-party contract signing protocol. After that, the proposed security property design is described and how the main parts of the multi-party contract signing protocol were modeled in terms of colored Petri nets and presents some of its subnets in Section 4. After model checking and the state space analysis, the fairness properties are found as a fair transition or just transition or no fairness condition. Finally, the work and suggest future research are concluded in Section 5.

II. RELATED WORK

As contract signing is a particular case of fair exchange, any fair exchange protocol found in the literature in which digital signatures are exchanged can be considered as the related work [2]. In all practical schemes, contract signing involves an additional player, called trusted third party (TTP). This party is (at least to some extent) trusted to be have correctly, thus playing the role of a notary in paper-based contract signing and somehow sharing the legal duties the former one shaves. In fact, designing and implementing a contract signing protocol using a non-line TTP should not be a complicated task. In this
case, if Alice and Bob wish to enter into a contract, they each sign a copy of the contract and send it to the TTP through a secure channel. The TTP will forward the signed contracts only when it has received valid signatures from both Alice and Bob.

Additionally, if the TTP is not involved, the notary fee could be avoided. Some protocols appear in the literature trying to eliminate the TTP's involvement using gradual exchange of signatures. But these solutions are not deterministic, thus may not be accepted by signatories. Our objective is to focus on contract signing protocols that necessarily use a TTP only in those cases in which an exception occurs (i.e., a network communication failure or a dishonest party's misbehaviour). Otherwise (all-honest-case), the TTP will not be contacted, and parties will bring the protocol to its end by themselves. In the literature, these protocols are called optimistic contract signing protocols.

Some properties extracted from the different previous work on optimistic contract signing are summarized as follows [2].

Effectiveness - if each party behaves correctly, the TTP will not be involved in the protocol.

Fairness - no party will be in an advantageous situation at the end of the protocol.

Timeliness - any party can decide when to finish a protocol run without losing fairness.

Non-repudiation - no party can deny its action.

Transparency of TTP - if the TTP is contacted to resolve the protocol, the resulting contract will be similar to the one obtained in case the TTP is not involved.

Abuse-Freeness - it is not possible for an attacker (either a legitimate participant or an outsider) to show a third party that the TTP misbehaves.

Verifiability of TTP - if the TTP misbehaves, all harmed parties will be able to prove it.

In summary, the CPN methodology offers an advantage over those formal methods discussed previously in that it provides a simpler and more intuitive way to model a protocol by using the graph representation.

III. AN OPTIMAL MULTI-PARTY CONTRACT SIGNING PROTOCOL

There are n signing parties P_1, ..., P_n. Protocol proceeds at least (n+1) round [2]. The following simple solution uses verifiable encryption of signatures based on ring architecture for achieving transparency of the TTP. Assume that the channel between any participant and the TTP is functional and not disrupted. The following notation is used in the protocol description.

- \( C = [M, P, \text{id}, t] \): a contract text M to be signed by each party Pi \( \in \) P (i = 1, ..., n), a unique identifier id for the protocol run, and a deadline t agreed by all parties to contact the TTP.
- \( e_{P} (X) \): encryption of message X with P's public key.
- \( S_{P} (X) \): P's digital signature on X.
- \( \text{Cert}_{i} \): a certificate with which anyone can verify that the cipher text is the correct signature of the plaintext, and can be decrypted by the TTP (see CEMBS-Certificate of an Encrypted Message Being a Signature).

An optimal synchronous protocol for multi-party contract signing is outlined as follows [2]:

1. \( P_{1} \rightarrow P_{2} : m_{1} = [C, e_{TPP} (S_{P_{1}} (C)), \text{Cert}_{1}] \)
2. \( P_{2} \rightarrow P_{3} : m_{2} = [C, e_{TPP} (S_{P_{2}} (C)), \text{Cert}_{2}] \)
3. \( P_{n} \rightarrow P_{1} : m_{n} = [C, e_{TPP} (S_{P_{n}} (C)), \text{Cert}_{n}] \)
4. \( P_{n} \rightarrow P_{n-1} : m_{n-1} = [C, e_{TPP} (S_{P_{n-1}} (C)), \text{Cert}_{n-1}] \)
5. \( P_{n} \rightarrow P_{n-1} : m_{n-2} = [C, e_{TPP} (S_{P_{n-2}} (C)), \text{Cert}_{n-2}] \)
6. \( P_{n} \rightarrow P_{n-1} : m_{n-3} = [C, e_{TPP} (S_{P_{n-3}} (C)), \text{Cert}_{n-3}] \)
7. \( P_{n} \rightarrow P_{n-1} : m_{n-4} = [C, e_{TPP} (S_{P_{n-4}} (C)), \text{Cert}_{n-4}] \)
8. \( P_{n} \rightarrow P_{n-1} : m_{n-5} = [C, e_{TPP} (S_{P_{n-5}} (C)), \text{Cert}_{n-5}] \)
9. \( P_{n} \rightarrow P_{n-1} : m_{n-6} = [C, e_{TPP} (S_{P_{n-6}} (C)), \text{Cert}_{n-6}] \)
10. \( P_{n} \rightarrow P_{n-1} : m_{n-7} = [C, e_{TPP} (S_{P_{n-7}} (C)), \text{Cert}_{n-7}] \)
11. \( P_{n} \rightarrow P_{n-1} : m_{n-8} = [C, e_{TPP} (S_{P_{n-8}} (C)), \text{Cert}_{n-8}] \)
12. \( P_{n} \rightarrow P_{n-1} : m_{n-9} = [C, e_{TPP} (S_{P_{n-9}} (C)), \text{Cert}_{n-9}] \)
13. \( P_{n} \rightarrow P_{n-1} : m_{n-10} = [C, e_{TPP} (S_{P_{n-10}} (C)), \text{Cert}_{n-10}] \)
14. \( P_{n} \rightarrow P_{n-1} : m_{n-11} = [C, e_{TPP} (S_{P_{n-11}} (C)), \text{Cert}_{n-11}] \)
15. \( P_{n} \rightarrow P_{n-1} : m_{n-12} = [C, e_{TPP} (S_{P_{n-12}} (C)), \text{Cert}_{n-12}] \)
16. \( P_{n} \rightarrow P_{n-1} : m_{n-13} = [C, e_{TPP} (S_{P_{n-13}} (C)), \text{Cert}_{n-13}] \)
17. \( P_{n} \rightarrow P_{n-1} : m_{n-14} = [C, e_{TPP} (S_{P_{n-14}} (C)), \text{Cert}_{n-14}] \)
18. \( P_{n} \rightarrow P_{n-1} : m_{n-15} = [C, e_{TPP} (S_{P_{n-15}} (C)), \text{Cert}_{n-15}] \)
19. \( P_{n} \rightarrow P_{n-1} : m_{n-16} = [C, e_{TPP} (S_{P_{n-16}} (C)), \text{Cert}_{n-16}] \)
20. \( P_{n} \rightarrow P_{n-1} : m_{n-17} = [C, e_{TPP} (S_{P_{n-17}} (C)), \text{Cert}_{n-17}] \)
21. \( P_{n} \rightarrow P_{n-1} : m_{n-18} = [C, e_{TPP} (S_{P_{n-18}} (C)), \text{Cert}_{n-18}] \)
22. \( P_{n} \rightarrow P_{n-1} : m_{n-19} = [C, e_{TPP} (S_{P_{n-19}} (C)), \text{Cert}_{n-19}] \)
23. \( P_{n} \rightarrow P_{n-1} : m_{n-20} = [C, e_{TPP} (S_{P_{n-20}} (C)), \text{Cert}_{n-20}] \)

The protocol has two clearly differentiated phases: exchange of commitments and exchange of digital signatures [2]. The parties first exchange their commitments in an “in-and-out” manner. Note that, \( P_{i} \) can choose deadline t in the first message (and others can halt if they do not agree). Only after the first phase is finished at step n, the final signatures are exchanged. Following this simple approach, only 3(n - 1) steps are needed.

If there is no exception (e.g., network failure or misbehaving party), the protocol will not need the TTP's help [2]. Otherwise, the following resolve sub-protocol helps to drive the contract signing process to its end. \( P_{i} \) can contact the TTP before the deadline t.

1. \( P_{i} \rightarrow \text{TTP}: \text{resolve}_{P_{i}} = C, m_{i_{1}}, ..., m_{i_{n}} \)
2. \( \text{TTP} \): IF \( \text{resolve}_{P_{i}} \) is received before t THEN decrypts \( m_{i_{1}}, ..., m_{i_{n}} \), publishes \( S_{P_{i}} (C), \ldots, S_{P_{n}} (C) \)

If the main protocol is not completed successfully, some parties may not hold all the commitments (\( m_{i_{1}}, ..., m_{i_{n}} \)). Then, the parties just wait until the deadline t and check with the TTP whether the contract has been resolved by other parties. If not, the contract is cancelled. Otherwise, the parties get the valid contract (\( S_{P_{1}} (C), \ldots, S_{P_{n}} (C) \)) from the TTP.

If a party has all the commitments when the main protocol is terminated abnormally, it could initiate the above sub-protocol [2]. Then the TTP will help to resolve the contract if the request is received before the deadline t and the contract will be available to all the participants (even after the deadline t). After the deadline, the TTP will not accept such requests anymore. In other words, the status of the contract will be determined the latest by the deadline t. This optimal version permits overlapping the dispatch of promises with real signatures without losing fairness.

IV. MULTI-PARTY CONTRACT SIGNING PROTOCOL BY USING CPN TOOLS

In this section, the definitions of Petri Nets and Colored Petri Nets are presented first. Then, the current main techniques using CP-Nets to analyze the multi-party contract signing protocols are compared. Based on the techniques, a new method using CP-Nets is presented to analyze the multi-party contract signing protocols.

A. Petri Nets

Petri Nets are presented by Carl Adam Petri during his Ph.D. thesis in 1962. A Petri Net is a graphical and mathematical tool to verify systems and protocols. Petri Nets in the graphical forms are like flowcharts and network
diagrams, while in mathematical forms, they are like algebra and logic subjects.

Definition 1:
In a formal way, A Petri Net is a tuple [5]:
\[ PN=(P,T,A,N) \]

In the tuple,
(1) \( P \) is a finite set of Places.
(2) \( T \) is a finite set of Transitions.
(3) \( A \) is a finite set of Arcs such that:
\[ P \cap T = P \cap A = T \cap A = \emptyset \]
(4) \( N \) is a set of Token.

B. Colored Petri Nets (CPN Tools)

CPN is a formal method based on graph theory to analyse distributed systems and communication protocols [6]. In fact, CPN is a model checking technique where a system is first modelled by a kind of graphs, called a net and then a state space of all possible executions of the system is generated and analysed to search for errors in the system. However, only a single error trace can be detected by the built-in mechanism in CPN.

CPN extends Petri nets with programming abilities. The programming language provided in CPN is a functional programming language called CPN-ML, and it is based on standard ML. A graph model in CPN contains four main components: places, transitions, arcs and tokens. Places, represented by circles, and transitions, represented by rectangles, are used to describe states and actions of the system, respectively. Arcs represented by arrows are used to link between place and transition. Tokens mean data and they are just a rich set of data types in standard ML. Arc expressions which are CPN-ML programs attached to arcs describe how a transition produces output tokens from input tokens. A transition can occur, called enabled, only when there are sufficient tokens that match the arc expressions on its input places. When an enabled transition is executed, called occurred, tokens from input places are removed and new tokens are added into output places according to corresponding arc expressions. Thus, the main programming functionality in CPN is to perform token processing at transitions. Furthermore, CPN provides a software tool called CPN Tools which facilitates the creation, the modification and the analysis of nets.

The original Petri nets, called place/transition nets, are considered as low-level nets. It is widely known that low-level nets are not practical to deal with real-world applications due to their unmanageable size of the net specification. Thus, many high-level nets, for example CPN and predicate/transition nets, have been developed to solve this problem. In fact, CPN is very much similar to predicate/transition nets and they are considered as two different dialects.

There are other kinds of high-level nets, for example algebraic Petri nets, CPN with algebraic specifications, etc. Most of them are similar to CPN, but the difference is on the inscription language which is the language for arc expressions and tokens. While the inscription language in CPN is based on functional programming, the inscription language in algebraic Petri nets and CPN with algebraic specifications is based on algebraic specifications. Further discussion on the difference between CPN and other high-level nets can be found.

C. Fairness Of Multi-Party Contract Signing Protocol

Fairness is only relevant if there are Infinite Firing Sequences (IFS), otherwise CPN Tools reports that "no infinite occurrence sequences". Given a transition ‘t’ it is often desirable that t appears infinitely often in an IFS. The proposed security property design by using CPN Tools for achieving fairness is shown in Fig.1.

![Fig. 1 The Proposed Security Property Design by using CPN Tools for achieving fairness.](image)

**Statistics**

<table>
<thead>
<tr>
<th>State Space</th>
<th>Nodes: 5</th>
<th>Arcs: 10</th>
<th>Secs: 0</th>
<th>Status: Full</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scc Graph</td>
<td>Nodes: 1</td>
<td>Arcs: 0</td>
<td>Secs: 0</td>
<td></td>
</tr>
</tbody>
</table>

**Boundedness Properties**

<table>
<thead>
<tr>
<th>Best Integer Bounds</th>
<th>Upper</th>
<th>Lower</th>
</tr>
</thead>
<tbody>
<tr>
<td>New_Page'p1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>New_Page'p2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>New_Page'p3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>New_Page'p4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>New_Page'p5</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Best Upper Multi-set Bounds</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>New_Page'p1</td>
<td>1’ ()</td>
<td></td>
</tr>
<tr>
<td>New_Page'p2</td>
<td>1’ ()</td>
<td></td>
</tr>
<tr>
<td>New_Page'p3</td>
<td>1’ ()</td>
<td></td>
</tr>
<tr>
<td>New_Page'p4</td>
<td>1’ ()</td>
<td></td>
</tr>
</tbody>
</table>

http://dx.doi.org/10.15242/IIE.E0314045
V. CONCLUSION

Contract signing is a fundamental service for business transactions. Previous work mainly focused on two-party contract signing. In some applications, however, a contract may need to be signed by multiple parties. In this paper, the multi-party contract signing protocol with the additional feature like fairness property is analyzed by using CPN Tools. The simulation results produce the state space report and generate the simulation report in step by step. CPN Tools allows an efficient way to analyze multi-party contract signing protocol execution with various properties. As a future work, the other properties of multi-party contract signing protocol can analyze and simulate with CPN Tools and includes the formal security analysis of optimal multi-party contract signing protocol.

ACKNOWLEDGMENT

The author wishes to her special thanks to her principal, Dr. Aung Win, Principal of University of Technology (Yatanarpon Cyber city) for his invaluable suggestion, leading and helpful guidance during the period of study. The author would like to heartfelt thanks to all her teachers.