

Inductive Methods and Contract-Signing Protocols*

R. Chadha M. Kanovich A. Scedrov
{rchadha@math,maxkanov@central.cis,scedrov@cis}.upenn.edu
University of Pennsylvania

ABSTRACT

Garay, Jakobsson and MacKenzie introduced the notion of *abuse-free* distributed contract-signing: at any stage of the protocol, no participant A has the ability to prove to an outside party, that A has the power to choose between completing the contract and aborting it. We study a version of this property, which is naturally formulated in terms of game strategies, and which we formally state and prove for a two-party, optimistic contract-signing protocol. We extend to this setting the formal inductive proof methods previously used in the formal analysis of simpler, trace-based properties of authentication protocols.

1. INTRODUCTION

Distributed contract signing over a network involves many challenges in mimicking the features of paper contract signing. For instance, a paper contract is usually signed by both parties at the same time and at the same place, but distributed electronic transactions over a network are inherently asymmetric in that someone has to send the first message.

Several digital contract-signing protocols have been devised in order to overcome this basic asymmetry and to achieve symmetric properties such as *fairness*, namely, neither party signs the contract without receiving the other party's signature. Even and Yacobi [8] showed that no deterministic two-party contract-signing protocol can achieve fairness. Even [7] introduced contract-signing protocols with a third party. A trusted third party can enforce the contract after it witnesses a partial completion of the protocol. In *optimistic* contract-signing protocols [1, 10] the trusted third party is contacted only in case of a dispute, otherwise the protocol can be completed without involving the third party. Such protocols involve several subprotocols that al-

low a contract to be signed normally or aborted or resolved by the trusted third party. This makes reasoning about such protocols complicated. A formal analysis of such protocols was initiated in [18, 15] using a finite state verification tool. More recently, Das and Dill [5] used predicate abstraction and a finite-state model checker to prove that the protocol in [10] is fair.

Another kind of symmetry desirable in distributed contract signing was identified in [10]: a fair protocol is said to be *abuse-free* if, at any stage of the protocol, it is impossible for any participant, say A , to be able to prove to an outside challenger that A has the power to choose between completing the contract and aborting it. (If the intruder takes only finitely many steps, then it is a consequence of fairness that there are only two possible outcomes: either no one receives a valid contract or both parties receive a valid contract.) Abuse-freeness is an important property of contract signing. Indeed, suppose that Alice and Bob use a protocol that is *not* abuse-free. Then it is possible for one party, say Alice, at some point to convince another party, Charlie, that Bob is committed to the contract, whereas Alice is not yet. This would put Bob at a distinct disadvantage, because there is a risk that Alice does not really want to sign the contract with Bob but only use Bob's willingness to sign to gain leverage for another contract with Charlie.

Our aim in this paper is to study a version of abuse-freeness for a two-party optimistic contract-signing protocol slightly revised from [10, 15]. We adopt the multiset-rewriting formalism for protocol analysis [3], which reflects the two basic assumptions of the Dolev-Yao model [16, 6, 21], perfect cryptography coupled with nondeterministic computation on the part of the intruder. These assumptions provide an idealized setting in which protocol analysis becomes relatively tractable.

One of the decisions is how to model dishonest participants. We model two kinds of dishonest participants. One is a *weakly dishonest* participant, which behaves normally but is always ready to accept messages from the network and take no further part in the protocol. We also discuss a *strongly dishonest* participant which shares its private keys with the Dolev-Yao intruder prior to the execution of the protocol. On the other hand, we assume that the trusted third party is well-behaved and does not play the role of a party interested in signing the contract.

We formally state and prove a version of abuse-freeness similar to the version suggested in [15]. We call our version of the property the *balance* property. In particular, assuming that the intruder takes only finitely many steps

*Chadha and Scedrov were partially supported by DoD MURI "Semantic Consistency in Information Exchange" as ONR Grant N00014-97-1-0505, NSF Grant CCR-9800785 and by NSF grant CCR 0098096.

and that any one protocol participant and the trusted third party are honestly following the protocol, we prove that, at any stage of the protocol, the other protocol participant, even a strongly dishonest one, does not have both the power to complete the contract as well as the power to abort it. In this way, if Charlie requests any particular outcome, Alice would not be able to guarantee it.

More precisely, we view A 's power to abort as A 's ability to prevent the successful completion of the contract in coalition with the intruder, regardless of the actions of the other (honest) participant and the actions of the trusted third party. Similarly, we view A 's power to complete the contract as A 's ability to prevent the abort in coalition with the intruder, regardless of the actions of the other (honest) participant and the actions of the trusted third party.

Because our setting is asynchronous and a participant cannot control the actions of other parties, we take the view that the only ability that a participant has is that it may choose to fire a certain subset of its allowable actions at any stage of the protocol and hope to determine the outcome in that way. Our version of abuse-freeness, which assumes fairness, may be stated equivalently in terms of certain recursive properties of finite trees, which we prove by inductive methods. (Fairness itself is proved by inductive proof methods as well.) Inductive methods were introduced in [17] and used more recently in [4], both in the analysis of simpler properties of authentication protocols [21].

Let us emphasize that balance is not a trace-based property in that it refers to the entire execution tree rather than to any single branch. In particular, balance is not a correspondence property in the sense of [21]. Nevertheless, we show that balance may be represented as formal provability in a logical system in which formal derivations correspond to protocol execution trees.

In Section 2 we describe our revised version of the Garay-Jakobsson-MacKenzie (GJM) two-party contract-signing protocol [10, 15]. Section 3 contains a brief description of the multiset-rewriting framework [3], which is used in Section 4 to formalize the protocol. Section 5 contains the discussion of fairness of the protocol. In Section 6 we discuss the balance property and give an inductive proof of this property. In Section 7 we indicate how balance may be represented using provability in linear logic [11].

We thank Karthikeyan Bhargavan, Iliano Cervesato, David Dill, Satyaki Dutta, Carl Gunter, Joshua Guttman, Steve Kremer, Philip Mackenzie, Dahlia Mahlki, Cathy Meadows, Jonathan Millen, John Mitchell, Sylvan Pinsky, Vitaly Shmatikov, Scott Stoller and Paul Syverson for interesting and helpful discussions. We would also like to thank the anonymous referees for their useful comments.

2. THE GARAY-JAKOBSSON-MACKENZIE PROTOCOL

We now describe a protocol obtained by a slight revision of the two-party optimistic contract-signing protocol of Garay, Jakobsson, and MacKenzie in [10, 15]. We discuss the differences in our protocol and these protocols in Section 2.3. In the protocol, the two parties may contact a trusted third party, henceforth referred to as T . We start by describing the purpose, cryptographic assumptions, communication model under which the protocol is meant to be executed.

2.1 Purpose and assumptions of the protocol

The purpose of the protocol is to enable two parties, O and R , to exchange signatures on a previously agreed upon contractual text, m with the help of a trusted third party, T .

Each protocol participant is assumed to have a private signing key and a corresponding public verification key. Each participant is identified with this private signing/public verification key pair. In particular, for the rest of a section if we say that " A can..", we mean anyone that possesses the private signing key of A .

The protocol uses a cryptographic primitive, *private contract signature* (PCS) introduced in [10]. We write $PCS_O(m, R, T)$, for the private contract signature of O on a contract text m , intended for R with respect to T . The main properties of private contract signatures are:

- 1) $PCS_O(m, R, T)$ can be computed by O .
- 2) There is a probabilistic polynomial-time algorithm $PCS-Ver$ such that $PCS-Ver(m, O, R, T, S) = true$ if $S = PCS_O(m, R, T)$. This algorithm can be run by anybody who knows the public verification keys of O , R and T .
- 3) R can compute $S = FakeSign_R(m, O, T)$ such that $PCS-Ver(m, O, R, T, S) = true$. Nobody other than O or R can compute S such that $PCS-Ver(m, O, R, T, S) = true$. O can verify whether it was created by R and similarly R can verify whether it was created by O . T can also verify whether it was generated by O or R .
- 4) O can convert $PCS_O(m, R, T)$ into a conventional digital signature, $sig_O(m)$. There is a probabilistic-polynomial time algorithm, $S-Ver$ such that $S-Ver(m, O, T, S) = true$ iff $S = sig_O(m)$. Nobody other than O can compute S such that $S-Ver(m, O, T, S) = true$.
- 5) T can convert $PCS_O(m, R, T)$ into a digital signature, $TP-Sig_O(m)$. There is a probabilistic-polynomial time algorithm, $T-Ver$ such that $T-Ver(m, O, T, S) = true$ iff $S = TP-Sig_O(m)$. Nobody other than T can compute S such that $T-Ver(m, O, T, S) = true$.

Our formalization of the cryptographic primitives will involve an abstract version of these primitives.

Usually, no assumption is made on the communication channel between the two parties. An *intruder* may assume full control over the communication channel between these two parties. However the protocol participants have separate communication channels to T . These channels are assumed to be *write-protected*, i.e., nobody except the T and the participant (or anybody who possesses the private signing key of the participant) can write on this channel. Furthermore these channels are assumed to be *transparent*; the channel never loses a message, unless the intended party reads it; and the intruder can observe messages from this channel without blocking or delaying them. We assume an asynchronous communication model. We have relaxed the condition of private channels as stated in [10].

T is assumed to maintain a permanent database of each of the protocol instances that it has acted upon before. In our analysis, we further assume that T does not misbehave and does not act as a party to a contract. In [15], the authors allowed T to be accidentally corrupt, namely the database of T was visible to the intruder. Since we have relaxed the notion of private channels, we do allow the accidentally corrupt behavior of T , by making all communication between it and the participants observable to the intruder.

2.2 Protocol Description

The protocol consists of three subprotocols: *exchange*, *abort*, and *resolve* subprotocols. Usually the parties would try to achieve the contract by executing the exchange subprotocol. They would contact T using one of the other two subprotocols when they think something is amiss. Once they contact it, they no longer take part in the exchange subprotocol.

In the protocol description, when a participant A sends a message, $mssg$ intended for B , it will be abbreviated as $A \rightarrow B : mssg$.

In the description, the two parties will be designated as O , the originator and R , the respondent. Before executing the protocol, the participants are assumed to have agreed upon each other's identity, the contractual text m and the identity of the trusted third party, T . As in [10, 15], we do not consider how this is achieved, in our analysis. In addition, they also agree on a globally unique identifier n , before they execute the protocol. Different instances of the protocol would have different identifiers. We further assume that the intruder knows this identifier. Let $pd = \langle m, n, O, R, T \rangle$. We begin by describing the exchange subprotocol.

Exchange Subprotocol: O displays its commitment to sign by sending $PCS_O(pd, R, T)$, intended for R . If R receives it, then it displays its commitment by sending $PCS_R(pd, O, T)$, intended for O . If O receives it, it sends $sig_O(pd)$, intended for R . If R receives it, it sends $sig_R(pd)$ intended for O and the protocol finishes for R . The protocol finishes for O when it receives $sig_R(pd)$. The protocol steps are described as

$$\begin{aligned} O \rightarrow R : me_1 &= PCS_O(\langle m, n, O, R, T \rangle, R, T) \\ R \rightarrow O : me_2 &= PCS_R(\langle m, n, O, R, T \rangle, O, T) \\ O \rightarrow R : me_3 &= sig_O(\langle m, n, O, R, T \rangle) \\ R \rightarrow O : me_4 &= sig_R(\langle m, n, O, R, T \rangle) \end{aligned}$$

Abort Subprotocol: O may request T to abort the protocol after it sends $PCS_O(pd, R, T)$ and before it receives $PCS_R(pd, O, T)$. This it does by sending $ma_1 = sig_O(abort, pd)$ on the private $O-T$ channel. T on receiving the abort request, checks its database and checks if it has ever answered a request for pd that it received before on the $O-T$ Channel. If it has, it will not send back anything, otherwise it checks if it has resolved pd before. If it has, it sends the stored resolution $TP-Sig_O(pd)$, $TP-Sig_R(pd)$. Otherwise it issues an `abort_token`, $sig_T(ma_1)$ and sends it on the $O-T$ channel. It raises its abort flag for pd and stores the `abort_token`. An `abort_token` is a promise by T that it has not and will not resolve pd in future.

While R is not allowed to abort, it is allowed to quit before it receives the first message.

Resolve Subprotocol: O may request T to resolve the protocol after it sends $sig_O(pd)$ and before it receives $sig_R(pd)$. R may run the subprotocol after it sends $PCS_R(pd, O, T)$ and before it receives $sig_O(pd)$.

R requests T to resolve the protocol instance by sending $(PCS_O(pd, R, T), PCS_R(pd, O, T))$. T on receiving the resolve request, checks if it has ever answered a request for pd that it received before on the $R-T$ channel. If it has, it will not send back anything, otherwise it checks if it has aborted or resolved pd before. If it has been aborted or resolved before then it sends the stored decision on the $R-T$ channel. If it has neither aborted nor resolved the protocol, it converts the PCS to $TP-Sig_O(pd)$, $TP-Sig_R(pd)$ and sends it on the $R-T$ channel. It raises its resolved flag for

pd and stores the resolution.

O 's resolve subprotocol is similar.

DEFINITION 2.1. *A participant (either O or R) is said to have a valid contract if it has either $sig_O(pd)$ or $TP-sig_O(pd)$, and either $sig_R(pd)$ or $TP-sig_R(pd)$*

The authors of the original protocol [10] claimed several properties, including fairness and abuse-freeness. We discuss these properties in sections 5 and 7.

2.3 Differences from the original protocol

In the original protocol [10], the resolve request consisted of the requesting party's signature on the contract and the private contract signature of the other party. In [15], the authors showed that an accidentally corrupt T may lead to loss of fairness. They suggested a fix in which the resolve request consisted of private contract signatures of both the parties. We take this fixed protocol as our reference point and discuss our differences.

We discuss briefly the differences in our protocol from the versions of the GJM protocol as defined in [10, 15]:

- 1) We assume perfect cryptography, while in [10], the cryptographic primitives, are defined in terms of probabilistic polynomial time computations.
- 2) We assume write-protected and transparent channels between the participants and T , instead of private channels.
- 3) We explicitly include the identities of the parties involved in the contract in every message that is being sent. This is in accordance with the well-established practice of including the participants' identities in each step of a cryptographic protocol. Also, now each contract includes a globally unique identifier. In the previous two versions, there was no such identifier. We did this because both [10, 15] allow the following scenario:

O and R try to sign a contract with contract text m and trusted third party, T . O sends $PCS_O(m, R, T)$ intended for R . The intruder, henceforth called I listens on the channel between O and R .

$$O \rightarrow R : me_1 = PCS_O(m, R, T)$$

I intercepts it. R times out waiting for a reply, and quits. O asks T for an `abort_token`, who not having an entry for the contract before, issues the `abort_token`.

$$O \rightarrow T : ma_1 = sig_O(abort, m, O, R, T)$$

$$T \rightarrow O : ma_2 = sig_T(sig_O(abort, m, O, R, T))$$

Now, suppose O and R decide to sign m again with T as the trusted third party. They start by using the exchange protocol.

$$O \rightarrow R : me_1 = PCS_O(m, R, T)$$

$$R \rightarrow O : me_2 = PCS_R(m, O, T)$$

$$O \rightarrow R : me_3 = sig_O(m)$$

$$R \rightarrow O : me_4 = sig_R(m)$$

I intercepts.

O does not receive $sig_R(m)$ and times out. Then O asks T for a resolution, which having an `abort_token` in its database for this (m, O, R, T) sends the `abort_token` to O . Hence R has a valid contract and O only an `abort_token`, thus violating fairness.

- 4) The protocol as presented in [10] seems to allow T to be contacted multiple times by each party. This seems to violate an optimistic setting since a malicious participant could contact T multiple times forcing it to do expensive database searches. We revise the protocol so that T acts on only one request from each participant for each run of the

protocol. This should be sufficient because according to the protocol definition each participant is allowed to contact T only once. O may contact it for an abort or a resolve. R may contact it for a resolve.

3. MULTISSET-REWRITING FORMALISM

The protocol formalism we use is multiset-rewriting with existential quantification, MSR, as described in [3]. We outline it here only briefly. Its syntax involves terms, facts and rules. If one wants to represent a system in this notation, one begins by choosing a vocabulary, or *first-order signature*. This is a standard notion from many-sorted algebra or first-order logic. As usual, the *terms* over a signature are the well-formed expressions produced by applying functions to arguments of the correct sort. A *fact* is a first-order atomic formula over the chosen signature, without free variables. This means that a fact is the result of applying a predicate symbol to ground terms of the correct sorts. A *state* is a multiset of facts (all over the same signature).

A state transition is a *rule* written using two multisets of first-order atomic formulas, and existential quantification, in the syntactic form

$F_1, \dots, F_k \longrightarrow \exists x_1 \dots \exists x_j. G_1, \dots, G_n$. The meaning of this rule is that if some state S contains facts obtained by a ground substitution σ from first-order atomic formulas F_1, \dots, F_k , then one possible next state is the state S' that is similar to S , but with facts obtained by σ from F_1, \dots, F_k removed and facts obtained by σ from G_1, \dots, G_n added, where $x_1 \dots x_j$ are replaced by new symbols. If there are free variables in the rule $F_1, \dots, F_k \longrightarrow \exists x_1 \dots \exists x_j. G_1, \dots, G_n$, these are treated as universally quantified throughout the rule. In an application of a rule, these variables may be replaced by any ground terms.

As an example, consider the state, $\{P(f(a)), P(b)\}$, and rule, $(P(x) \longrightarrow \exists z. Q(f(x), z))$. A possible next state is obtained by instantiating the rule to $P(f(a)) \longrightarrow \exists z. Q(f(f(a)), z)$. Applying this rule, we choose a new value, c , for z and replace $P(f(a))$ by $Q(f(f(a)), c)$. This gives us the state $\{Q(f(f(a)), c), P(b)\}$.

As presented in [3], a protocol theory consists of three parts: a bounded phase describing protocol initialization that distributes keys or establishes other shared information, a role generation theory that designates possibly multiple roles that each principal may play in a protocol (such as initiator, responder, client, or server), and a disjoint union of bounded subtheories that each characterize a possible role, and which ensure that each role is finite. Furthermore, the multiset-rewriting formalism allows us to formulate one standard intruder theory per given signature, which describes any intruder for any protocol formalized in the language of the given signature. Looping is prevented by certain technical conditions discussed in [3].

In our protocols, the participants would be identified with their private digital signature and the corresponding public verification keys. Usually a digital signature on a message is a pair consisting of a message and the signature on the message. The signature is used to verify the person who is supposed to have generated it. In our formalism a message x , signed with a private key k_s is denoted by $sig(k_s, x)$ where k_v is the corresponding public verification key. This notation, allows us to model the verification by pattern matching. So, for example a protocol participant, A , who is waiting for a message signed under a key whose

public verification key is k_v when presented with a message, $sig(k_v, x)$ on the network would accept it. This rule expressed as $A(k_v), N(sig(k_v, x)) \rightarrow A(k_v, sig(k_v, x))$ would abstract away the verification process. This verification by pattern matching is similar to the encryption by pattern matching as presented in [3].

Given a participant O with public verification key k_o , a participant R with public verification key k_r and a trusted third party, T with public verification key k_t , $PCS_O(m, R, T)$ will be denoted by $PCS(k_o, m, k_r, k_t)$. $FakeSign_R(m, O, T)$ by $FakeSign(k_r, m, k_o, k_t)$ and $TP-Sig_O(m)$ by $tsig(k_t, k_o, m)$.

4. PROTOCOL DEFINITION IN MSR

We now discuss the precise definition of the protocol in MSR. For the sake of convenience, we use some abbreviations:

$pd = \langle m, n, k_o, k_r, k_t \rangle$, $me_1 = PCS(k_o, pd, k_r, k_t)$, $me_2 = PCS(k_r, pd, k_o, k_t)$, $me_3 = sig(k_o, pd)$, $me_4 = sig(k_r, pd)$, $ma_1 = sig_o(abort, pd)$, $mr_1 = \langle me_1, me_2 \rangle$, $ab_tok = sig(k_t, ma_1)$, and $res_cn = \langle tsig(k_t, k_o, pd), tsig(k_t, k_r, x) \rangle$.

pd identifies the protocol instance. me_i is the i -th message of the exchange protocol. ma_1 is the abort request, ab_tok is the abort_token, mr_1 is the resolve request and res_cn is the resolution from T .

We assume an initial finite set (not a multiset) of facts, Σ . Σ contains many $KP(k_s, k_v)$ predicates. A $KP(k_s, k_v)$ predicate identifies a protocol participant, whose private/public key pair is (k_s, k_v) . Honest participants amongst these will be identified by $HonestGuy(k_s, k_v)$ predicates and weakly dishonest participants will be identified by $WDisHonestGuy(k_s, k_v)$. Strongly dishonest participants would be identified by $BadKey(k_s, k_v)$ predicates. Both keys in the $BadKey$ predicate are known to the intruder. Σ also contains $TTP(k_{ts}, k_t)$ predicates which identifies trusted third parties with private/public signing key pairs (k_{ts}, k_t) . Furthermore, the key pairs amongst the KP and TTP predicates are pairwise disjoint. The publicly announced verification keys of the participants are denoted by $AnnK(k_v)$ predicates and the publicly announced verification keys of trusted third parties are denoted by $AnnT(k_t)$ predicates.

Σ also contains $contract(m)$ predicates which identifies contract texts. We further assume that the intruder knows all these contract texts.

The MSR definition consists of a role generation, protocol theories of O, R, T and the intruder theory.

4.1 Role Generation Theory

The role generation theory consists of a single rule:

$KP(k_{os}, k_o), KP(k_{rs}, k_r), TTP(k_{ts}, k_t), contract(m) \rightarrow \exists n. O_0(pd), R_0(pd), T_0(pd), KP(k_{os}, k_o), KP(k_{rs}, k_r), TTP(k_{ts}, k_t), contract(m), M(n)$

This rule shall be henceforth referred to as RG . In the role generation theory, two principals with key pairs (k_{os}, k_s) and (k_{or}, k_r) agree upon a contract text m , identity of the trusted third party k_t , and a globally unique identifier, n . The uniqueness of n is guaranteed because existential quantification means generation of a fresh value. The trusted third party's state is initialized to $T_0(pd)$. The state $T_0(pd)$ should be thought of as the state of T in which T has never heard about pd and hence has no entry for it in its database. This intuition makes sense because of the freshness of n , T

could not have heard of pd before this rule is applied. This rule abstracts away the agreement of contract text, identity of T and the globally unique identifier. The predicate M indicates the intruder's memory. Now the participants are ready to commence the protocol.

4.2 Protocol Theory

The protocol theory for O is shown in Table 1.

In the protocol definition, O has 10 states. The numbered states O_0, O_1, O_2, O_3 denotes O 's state during the exchange subprotocol execution. O_0 is the state of O at the start of the protocol. O_1 corresponds to the state in which it has sent me_1 , O_2 corresponds to the state in which it has received me_2 , O_3 to the state in which it has sent me_3 . O_{com} corresponds to a state in which it has received me_4 . $O_{ab?}$ corresponds to the state in which O has issued an abort request to T and O_{ab1} corresponds to the state in which it has an abort_token for this request and O_{res1} corresponds to the state in which it receives a resolution for this request. $O_{res?}$ corresponds to the state in which O has issued a resolve request to T and O_{ab2} and O_{res2} correspond to the states in which it has received an abort_token or a resolution, respectively for this request.

The network between O and R is modeled by N_i predicates. The write-protected transparent channel between O and T is modeled by $Rn_i(k_t, k_o, -)$ predicates.

A weakly dishonest O , in addition to all of the above states, may accept any messages from the network in any of the 10 states and take no further part in the protocol. We have given the templates of the rules that allow us to model weakly dishonest O . The template $O_{\langle 4, i, j \rangle}$ allows O in state $i \in \{0, 1, 2, 3, ab?, res?, ab1, ab2, res1, res2\}$ to accept a message from the network state j , $1 \leq j \leq 4$ and go into the state $O_{\langle 4, i, j \rangle}$ and remember what was learned before. So, for example one rule that gets instantiated by the use of this template is $O_0(pd), N_1(x) \rightarrow O_{\langle 4, 0, 1 \rangle}(pd, x)$.

As mentioned before, a strongly dishonest O is modeled by a *BadKey*(k_s, k_v) predicate. Both k_s, k_v are known to the intruder.

The protocol theory of T is given in Table 2, for R in Table 3 and the intruder theory is given in table 4.

5. FAIRNESS

DEFINITION 5.1. *Any state that is reached from the initial set of facts, Σ by the application of zero or more transition rules in the role generation theory and the theories of O, R, T and I is called a **reachable configuration**.*

We now state fairness in our formalism. Concurrent runs of the protocol are modeled by different instantiations of the role generation theory. We show that each instance in concurrent runs is fair and hence the protocol is fair. Assume that in configuration S_0 , two principals O and R , whose private/public key pairs are (k_{os}, k_o) , and (k_{rs}, k_r) , agree to sign a contract with contractual text m , and a trusted third party, T whose private/public key pair is (k_{ts}, k_t) by using rule *RG*. A fresh globally unique identifier, n gets generated and we have a new protocol instance identified by $pd = \langle m, n, k_o, k_r, k_t \rangle$. Let the resulting configuration be S_1 . For the rest of the section, unless otherwise stated, reachable configurations would mean configurations reachable from S_1 .

Assume O is honest and R strongly dishonest. By $O_i(pd, -)$ we mean O in its i -th state with first argument as pd .

DEFINITION 5.2. *Let $pd = \langle m, n, k_o, k_r, k_t \rangle$ identify a protocol instance. For all reachable S , an honest O has completed pd in S if $O_i(pd, -) \in S$ for $i \in \{ab1, ab2, res1, res2, com\}$. An honest O has an abort_token for pd in S if $O_i(pd, -) \in S$ for $i \in \{ab1, ab2\}$. An honest O has a valid contract for pd in S if $O_i(pd, -) \in S$ for $i \in \{res1, res2, com\}$.*

Similar definitions can be given for T having an abort_token for pd in S , T having a resolution for pd , T having answered a request for pd on the $O - T$ channel.

DEFINITION 5.3. *For a set of terms, Tms ,*

1. *$Analz(Tms)$ is the smallest superset of Tms , such that $X \in analz(Tms)$ and $Y \in analz(Tms)$ if $\langle X, Y \rangle \in analz(Tms)$; $x \in analz(Tms)$ if $sig(k, x)$ or $PCS(k, x, k_1, k_t)$ or $FakeSign(k, x, k_1, k_t)$ or $tsig(k, k_1, x) \in analz(Tms)$.*
2. *$Synth(Tms)$ is the smallest superset of Tms , such that $\langle X, Y \rangle \in synth(Tms)$ if $X \in synth(Tms)$ and $Y \in synth(Tms)$; $PCS(k, x, k_1, k_t), FakeSign(k, x, k_1, k_t) \in synth(Tms)$ if $x, k^{-1} \in Synth(Tms)$; $sig(k, x), tsig(k, k_1, x) \in synth(Tms)$ if $x, k^{-1}, PCS(k, x, k_1, k_t) \in synth(Tms)$; where k^{-1} is the signature key corresponding to the verification key k .*
3. *For a given configuration S , the set of messages available to the intruder is defined to be $A(S) = \{x | C(x), D(x), M(x), N_i(x), Rn_i(k_1, k_2, x) \in S\}$. The **VIEW** of the intruder is defined to be $W(S) = synth(analz(AM(S)))$. If $x \in W(S)$ then we say that x is in the view of intruder in S .*
4. *A strongly dishonest R is said to have a valid contract if $sig(k_o, pd)$ or $tsig(k_t, k_o, pd)$ is in the view of the intruder. It is said to have an abort_token if abort_token is in the view of the intruder.*

Because of the freshness of n in pd , it can be shown by induction on the length of derivations of reachability, that in any reachable configuration O cannot be in two conflicting states for same pd .

Now we define, fairness for honest O . We break the definition in [10] into two definitions. The definitions for an honest R can be stated similarly.

DEFINITION 5.4. Effectiveness for honest O : *Let $pd = \langle m, n, k_o, k_r, k_t \rangle$ identify a protocol instance and let O be an honest participant.*

1. *There is a reachable configuration S , such that O has completed pd in S and has a valid contract.*
2. *For all reachable configurations S , there exists S' reachable from S , involving only the rules of O and T , such that O has completed pd in S' and has either a valid contract or an abort token for pd in S' .*

Because of the nondeterminism in our system, we cannot prove the stronger version stated in [10].

DEFINITION 5.5. Fairness for honest O : *Let $pd = \langle m, n, k_o, k_r, k_t \rangle$ identify a protocol instance. For all reachable configurations S ,*

Protocol Theory for O :

$$\begin{aligned} O_1 &: O_0(pd) \rightarrow O_1(pd, me_1), N_1(me_1) \\ O_{ab?} &: O_1(pd, me_1) \rightarrow O_{ab?}(pd, me_1, ma_1), Rn_1(k_t, k_o, ma_1) \\ O_2 &: O_1(pd, me_1), N_2(me_2) \rightarrow O_2(pd, me_1, me_2) \\ O_3 &: O_2(pd, me_1, me_2) \rightarrow O_3(pd, me_1, me_2, me_3), N_3(me_3) \\ O_{res?} &: O_3(pd, me_1, me_2, me_3) \rightarrow O_{res?}(pd, me_1, me_2, me_3, mr_1), Rn_3(k_t, k_o, mr_1) \\ O_{com} &: O_3(pd, me_1, me_2, me_3), N_4(me_4) \rightarrow O_{com}(pd, me_1, me_2, me_3, me_4) \\ O_{ab1} &: O_{ab?}(pd, me_1, ma_1), Rn_2(k_t, k_o, ab_tok) \rightarrow O_{ab1}(pd, me_1, ma_1, ab_tok) \\ O_{res1} &: O_{ab?}(pd, me_1, ma_1), Rn_2(k_t, k_o, res_cn) \rightarrow O_{res1}(pd, me_1, ma_1, res_cn) \\ O_{ab2} &: O_{res?}(pd, me_1, me_2, me_3, mr_1), Rn_4(k_t, k_o, ab_tok) \rightarrow O_{ab2}(pd, me_1, me_2, me_3, mr_1, ab_tok) \\ O_{res2} &: O_{res?}(pd, me_1, me_2, me_3, mr_1), Rn_4(k_t, k_o, res_cn) \rightarrow O_{res2}(pd, me_1, me_2, me_3, mr_1, res_cn) \end{aligned}$$

Additional Protocol theory for weakly dishonest O :

$$O_{<4,i,j>} : WDisHonestGuy(k_{os}, k_o), O_i(pd, -), N_j(x) \rightarrow WDisHonestGuy(k_{os}, k_o), O_{<4,i,j>}(pd, -, x)$$

Table 1: Protocol theory for O and weakly dishonest O

Protocol Theory for T :

$$\begin{aligned} T_{ab} &: Rn_1(k_t, k_o, ma_1), T_0(pd) \rightarrow T_{ab}(pd, aborted, ab_tok), Rn_2(k_t, k_o, ab_tok) \\ T_{or} &: Rn_3(k_t, k_o, mr_1), T_0(pd) \rightarrow T_{or}(pd, resolved, res_cn), Rn_4(k_t, k_o, res_cn) \\ T_{rr} &: Rn_5(k_t, k_r, mr_1), T_0(pd) \rightarrow T_{rr}(pd, resolved, res_cn), Rn_6(k_r, k_t, res_cn) \\ T_{abf} &: Rn_5(k_t, k_r, ma_1), T_{ab}(pd, aborted, ab_tok) \rightarrow \\ &T_{abf}(pd, aborted, ab_tok), Rn_6(k_t, k_r, ab_tok) \\ T_{orf} &: Rn_5(k_t, k_r, mr_1), T_{or}(pd, resolved, res_cn) \rightarrow \\ &T_{orf}(pd, resolved, res_cn), Rn_6(k_t, k_r, res_cn) \\ T_{rrf1} &: Rn_1(k_t, k_o, ma_1), T_{rr}(pd, resolved, res_cn) \rightarrow \\ &T_{rrf1}(pd, resolved, res_cn), Rn_2(k_t, k_o, res_cn) \\ T_{rrf2} &: Rn_3(k_t, k_o, mr_1), T_{rr}(pd, resolved, res_cn) \rightarrow \\ &T_{rrf2}(pd, resolved, res_cn), Rn_4(k_t, k_o, res_cn) \end{aligned}$$

Table 2: Protocol theory for T

Protocol Theory for R :

$$\begin{aligned} R_{quit} &: R_0(pd) \rightarrow R_{quit}(pd) \\ R_1 &: R_0(pd), N_1(me_1) \rightarrow R_1(pd, me_1) \\ R_2 &: R_1(pd, me_1) \rightarrow R_2(pd, me_1, me_2), N_2(me_2) \\ R_{res?} &: R_2(pd, me_1, me_2) \rightarrow R_{res?}(pd, me_1, me_2, mr_1), Rn_5(k_t, k_r, mr_1) \\ R_3 &: R_2(pd, me_1, me_2), N_3(me_3) \rightarrow R_3(pd, me_1, me_2, me_3) \\ R_{com} &: R_3(pd, me_1, me_2, me_3) \rightarrow R_{com}(pd, me_1, me_2, me_3, me_4), N_4(me_4) \\ R_{ab} &: R_{res?}(pd, me_1, me_2, mr_1), Rn_6(k_t, k_r, ab_tok) \rightarrow R_{ab}(pd, me_1, me_2, mr_1, ab_tok) \\ R_{res} &: R_{res?}(pd, me_1, me_2, me_3, mr_1), Rn_6(k_t, k_r, res_cn) \rightarrow R_{res2}(pd, me_1, me_2, me_3, mr_1, res_cn) \end{aligned}$$

Additional Protocol theory for weakly dishonest R :

$$R_{<4,i,j>} : WDisHonestGuy(k_{rs}, k_r), R_i(pd, -), N_j(x) \rightarrow WDisHonestGuy(k_{rs}, k_r), R_{<4,i,j>}(pd, -, x)$$

Table 3: Protocol theory for R and weakly dishonest R

Protocol Theory for the intruder:

I/O Rules :

$REC : N_{S_i}(x) \rightarrow D(x)$
 $SND : C(x) \rightarrow N_{R_i}(x)$
 $REC_R : Rn_i(k_1, k_2, x) \rightarrow D(x), Rn_i(k_1, k_2, x)$
 $SND_R : C(x), M(k_{2s}), KP(k_{2s}, k_2) \rightarrow Rn_i(k_1, k_2, x), M(k_{2s}), KP(k_{2s}, k_2)$

Decomposition Rules :

$DCMP : D(\langle x, y \rangle) \rightarrow D(x), D(y)$
 $LRN : D(x) \rightarrow M(x)$
 $ReadPCS : D(PCS(k_o, x, k_r, k_t)) \rightarrow D(x)$
 $ReadSig : D(sig(k_s, x)) \rightarrow D(x)$
 $ReadFakeSig : D(FakeSign(k_r, x, k_o, k_t)) \rightarrow D(x)$
 $ReadTPSig : D(tsig(k_t, k_o, x)) \rightarrow D(x)$

Composition Rules :

$COMP : C(x), C(y) \rightarrow C(\langle x, y \rangle)$
 $USE : M(x) \rightarrow C(x), M(x)$
 $PCS : M(k_{os}), C(x), KP(k_{os}, k_o), ANNK(k_r), ANNT(k_t) \rightarrow$
 $M(k_{os}), C(PCS(k_o, x, k_r, k_t)), KP(k_{os}, k_o), ANNK(k_r), ANNT(k_t)$
 $FakeSign : M(k_{rs}), C(x), KP(k_{rs}, k_r), ANNK(k_o), ANNT(k_t) \rightarrow$
 $M(k_{rs}), C(FakeSign(k_r, x, k_o, k_t)), KP(k_{rs}, k_r), ANNK(k_o), ANNT(k_t)$
 $SIG : M(k_{os}), C(PCS(k_o, x, k_r, k_t)), KP(k_{os}, k_o) \rightarrow M(k_{os}), C(sig(k_o, x)), KP(k_{os}, k_o)$
 $TPSIG : M(k_{ts}), C(PCS(k_o, x, k_r, k_t)), KP(k_{ts}, k_t) \rightarrow M(k_{ts}), C(tsig(k_t, k_o, x)), KP(k_{ts}, k_t)$
 $GEN : \rightarrow \exists x.M(x)$

Table 4: Two-Phase Intruder Theory

1. If a strongly dishonest R has a valid contract for pd in S , then there exists S' reachable from S such that an honest O has a valid contract for pd in S' .
2. If an honest O has an `abort_token` for pd in S , then for all S' reachable from S , a strongly dishonest R does not have a valid contract for pd .

5.1 Database Properties

The proof of fairness depends on the following database properties :

LEMMA 5.1. Database Persistence : For all reachable configurations S , either T has no entry for pd , or has an `abort` or a resolution for pd . If T has an `abort`, then for all configurations S' reachable from S , T has an `abort` for pd in S' . If it has a resolution, then for all configurations S' reachable from S , T has a resolution for pd in S' .

Proof By Induction on reachability of S .

LEMMA 5.2. Database Consistency : For all reachable configurations S , if T has no entry for pd in S then it does not have an `abort` or a resolution for pd in S ; if it has an entry for pd in S then if it has an `abort` for pd in S it does not have a resolution for pd in S , and if it has a resolution for pd then it does not have an `abort` for pd in S .

Proof By Induction on reachability of S and freshness of nonce n in pd .

5.2 Effectiveness for Honest O

PROPOSITION 5.3. If in S , O is in one of the states $O_i, i \in \{0, 1\}$, and if T has answered a request on the O - T channel for pd then an `abort_token` or a resolution for pd is on the O - T channel.

Proof By Induction on reachability of S , uniqueness of identifier n in pd and transparency of the $O - T$ channel.

LEMMA 5.4. If in S , O is in a state in which it has requested for an `abort_token` from T for pd and is waiting for a reply, then

- 1) either T has answered a request on the O - T channel and an `abort_token` or a resolution is on the O - T channel.
- 2) T has yet to answer a request on the O - T channel and the request is on the O - T channel.

Proof By Induction on reachability of S , Proposition 5.3, uniqueness of identifier n in pd and the transparency of the $O - T$ channel.

LEMMA 5.5. If in S , O is in a state in which it has requested for a resolution from T for pd and is waiting for a reply, then

- 1) either T has answered a request on the O - T channel and an `abort_token` or a resolution is on the O - T channel.
- 2) T has yet to answer a request on the O - T channel and the request is on the O - T channel.

THEOREM 5.6. For all reachable configurations S , there exists S' reachable from S , involving only the rules of O and T , such that O has completed pd in S' and has either a valid contract or an `abort token` for pd in S' .

Proof For all reachable S , O is in exactly one of the 10 states. Hence the following cases arise:

1. $O_i \in S$ for $i \in \{ab1, ab2, res1, res2, com\}$: O has completed.
2. $O_{res?} \in S$: By lemma 5.5 either T has answered the resolution request in which case O reads the answer and completes the protocol; or else the request is still on the $O-T$ channel. In the latter case T answers the request and O reads the answer.
3. $O_3 \in S$: O requests T for a resolution. The result follows from case 2.
4. $O_2 \in S$: O sends me_3 and the result follows from case 3.
5. $O_{ab?} \in S$: Similar to case 2.
6. $O_1 \in S$: O requests T for an abort_token and the result follows from case 5.
7. $O_0 \in S$: O sends me_1 and the result follows from case 6.

THEOREM 5.7. *There is a reachable configuration S , such that O has completed pd in S and has a valid contract.*

Proof The results follows when both O and R follow the exchange subprotocol with no rules of intruder and T being used.

THEOREM 5.8. *Effectiveness for honest O holds.*

Proof By theorems 5.7 and 5.6.

5.3 Fairness for Honest O

PROPOSITION 5.9. *For all reachable S , k_{os}, k_{ts} is not in the view of the intruder.*

Proof By Induction.

LEMMA 5.10. *For all reachable S , an abort_token for pd is in the view of the intruder only if T has an abort_token for pd . A resolution for pd is in the view of the intruder only if T has a resolution for me_1 .*

Proof By Induction, using Proposition 5.9, database persistence and consistence.

LEMMA 5.11. *For all reachable S ,*

1. *If O has an abort_token for pd then T has an abort for pd and in all configurations S' reachable from S , a resolution for pd is not in the view of the intruder in S' .*
2. *If a resolution for pd is in the view of the intruder in S then for all states S' reachable from S , O does not have an abort_token.*

Proof By induction using lemma 5.10, database consistency and database persistence.

LEMMA 5.12. *For all reachable S ,*

1. *If O is in one of the states, $O_i(pd, -)$ where $i \in \{0, 1, 2, 3, res?\}$ then neither an abort request nor an abort_token is in the view of the intruder.*
2. *S does not contain $O_{ab2}(pd, -)$ i.e. the only way O has an abort_token is if it is in the state O_{ab1} .*

Proof By induction on the reachability of S .

LEMMA 5.13. *For all reachable S ,*

1. *If O is in one of the states, O_i where $i \in \{0, 1, 2, ab?, ab1, res1\}$, then me_3 is not in the view of the intruder.*
2. *If O has a abort_token for pd then for all S' reachable from S , me_3 is not in the view of the intruder. If me_3 is in the view of the intruder then for all S' reachable from S , O does not have an abort_token in S' .*

Proof By induction.

THEOREM 5.14. Fairness for honest O : *For all reachable configurations S ,*

1. *If either me_3 or a resolution from T is in the view of the intruder in S , then for all S' reachable from S , O cannot have an abort_token. By effectiveness for honest O , for all configurations S' reachable from S , there exists a reachable configuration S'' , (depending on S') s.t O is completed for pd in S'' . Since O cannot have an abort_token in S'' , O must have a resolution from T or a valid contract.*
2. *If O has an abort_token, in S then for all S' reachable from S , then neither a resolution nor me_3 is in the view of the intruder in S .*

Proof By lemma 5.13 and lemma 5.11.

COROLLARY 5.15. *Effectiveness and fairness for honest O holds even if the intruder takes a bounded number of steps and the role generation rule is used a bounded number of times.*

6. EFFECTIVENESS AND FAIRNESS FOR HONEST R

We now discuss briefly fairness for honest R . Assume that in configuration S_0 , two principals O and R , whose private/public key pairs are (k_{os}, k_o) , and (k_{rs}, k_r) , agree to sign a contract with contractual text m , and a trusted third party, T whose private/public key pair is (k_{ts}, k_t) by using rule RG . A fresh globally unique identifier, n gets generated and we have a new protocol instance identified by $pd = \langle m, n, k_o, k_r, k_t \rangle$. Let the resulting configuration be S_1 . For the rest of the section, unless otherwise stated, reachable configurations would mean configurations reachable from S_1 . Assume R is honest and O strongly dishonest. By $R_i(pd, -)$ we mean R in its i -th state with first argument as pd .

LEMMA 6.1. *If in S , R is in a state in which it has requested a resolution from T for pd and is waiting for a reply, then*

- 1) *either T has answered a request on the $R-T$ channel and an abort_token or a resolution is on the $R-T$ channel.*
- 2) *T has yet to answer a request on the $R-T$ channel and the request is on the $R-T$ channel.*

Using the above lemma, we obtain

THEOREM 6.2. Effectiveness for honest R :

1. There is a reachable configuration S , such that R has completed pd in S and has a valid contract.
2. For all reachable configurations S , there exists S' reachable from S , involving only the rules of R and T , such that R has completed pd in S' and has either quit or has obtained either a valid contract or an `abort_token` for pd in S' .

LEMMA 6.3. For all reachable S ,

1. If R is in one of the states, R_i where $i \in \{0, quit\}$, then neither me_2 nor a resolution is in the view of the intruder.
2. If R quits for pd then for all S' reachable from S , a resolution is not in the view of the intruder.

LEMMA 6.4. For all reachable S , an `abort_token` for pd is in the view of the intruder only if T has an `abort_token` for pd . A resolution for pd is in the view of the intruder only if T has a resolution for me_1 .

LEMMA 6.5. For all reachable S ,

1. If R has an `abort_token` for pd then T has an `abort` for pd and in all configurations S' reachable from S , a resolution for pd is not in the view of the intruder in S' .
2. If a resolution for pd is in the view of the intruder in S then for all states S' reachable from S , R does not have an `abort_token`.

LEMMA 6.6. For all reachable S ,

1. If R is in one of the states, R_i where $i \in \{0, quit, 1, 2, 3, ab?, ab, res\}$, then me_4 is not in the view of the intruder.
2. If R quits or R has an `abort_token` for pd then for all S' reachable from S , me_4 is not in the view of the intruder. If me_4 is in the view of the intruder then for all S' reachable from S , R is in state R_{com} i.e. it has a valid contract.

THEOREM 6.7. Fairness for honest R : For all reachable configurations S ,

1. If a strongly dishonest O has a valid contract for pd in S , then there exists S' reachable from S such that an honest R has a valid contract for pd in S' .
2. If an honest R quits or has an `abort_token` for pd in S , then for all S' reachable from S , a strongly dishonest O does not have a valid contract for pd .

Once again, we have the corollary:

COROLLARY 6.8. Effectiveness and fairness for honest R holds even if the intruder takes a bounded number of steps and the role generation rule is used a bounded number of times.

Fairness can also be shown for an honest O and an honest R or a weakly dishonest R ; an honest R and an honest O or a weakly dishonest O .

7. BALANCE

We now state a version of abuse-freeness in our formalism. Recall that Σ is the initial finite set of facts. Assume that in Σ , two principals O and R whose private/public key pairs are (k_{os}, k_o) , and (k_{rs}, k_r) , agree to sign a contract with contractual text m , unique identifier n , and a trusted third party T , whose private/public key pair is (k_{ts}, k_t) . This they do by the use of role generation rule RG . Let the resulting configuration be S_0 . For our analysis, we assume that O is honest and R is strongly dishonest. Since O is honest, it can be shown by using induction that for all configurations reachable from S_0 , O is in exactly one of the states, O_i , and that if once it has received an `abort_token`, it does not get a valid contract and vice-versa.

For the rest of the section, by a reachable configuration S , we mean a configuration reachable from Σ without the use of the role generation rule. Note that since we do not allow the use of role generation, we are considering only a single instance of the protocol in isolation and not a concurrent run of several instances. We have: if the intruder takes only a bounded number of steps, and no role generation rules are used, then the number of traces is finite.

By $S \xrightarrow{r} S'$, we mean that S goes to S' by the application of rule r .

DEFINITION 7.1. Let S be a reachable configuration and assume that the intruder takes a bounded number of steps.

For each $0 \leq i$, we define recursively a finite tree tr_i of height i , whose nodes are labeled by configurations and whose edges are labeled by rules of the protocol and the intruder theories:

1) Define tr_0 to be the tree consisting of a single node labeled by S .

2) Assume that we have defined the tree tr_i . Let X be the set of nodes at height i in tr_i . Pick a node N in X and fix it. Let N be labeled by S_i and let the nodes in the path from the root to N be labeled as S, S_1, S_2, \dots, S_i and the edges in the path be labeled as r_1, r_2, \dots, r_i in that order. For each configuration S' and a rule r (different from RG), such that $S \xrightarrow{r_1} S_1 \xrightarrow{r_2} \dots \xrightarrow{r_i} S_i \xrightarrow{r} S'$ is a derivation, if any, add a node as a child of N . Label this node S' and the edge from N to this node as r . Repeat this process for each node in X . Since X is finite and the number of traces is finite, this process terminates. Define tr_{i+1} to be the resulting tree.

Continuation tree of S is defined to be tr_M , such that $tr_M = tr_{M+1}$.

Because of effectiveness for honest O (Corollary 5.15), all the leaf nodes in the continuation tree would be labeled by configurations in which O has either an `abort_token` or a valid contract. Also since O is honest, O cannot have both an `abort_token` and a valid contract. By fairness for honest O , if O has an `abort_token` then R cannot have a valid contract. We take the view expressed in [10] and say that a contract is completed even if one party possesses a valid contract. (By fairness, if the dishonest party has a valid contract in some configuration, and if the intruder takes only a bounded number of steps, then in all the leaf nodes of the continuation tree, the honest party must have a valid contract).

Hence, in our analysis we would say that the protocol instance is aborted if O has an `abort_token` and is completed if O has a valid contract. The following definitions are motivated by game strategies.

DEFINITION 7.2. Let tr be the continuation tree of S . An edge of tr is said to be a **removable edge** if it is labeled by a rule in the theory of R or by a rule of the intruder or a rule that indicates a message being read from the network (that is a N_i predicate occurs in the precedent of the rule). Otherwise, the edge is said to be **non-removable**.

DEFINITION 7.3. Let S be a reachable configuration and assume that the intruder takes a bounded number of steps. Let tr be the continuation tree of S . Let E be a set of removable edges in tr . Define $tr \setminus E$, a **strategy**, to be the tree obtained from tr by removing the edges in E and all of their descendants (including descendants further down).

- 1) At S , R has the power to abort if there is an E such that the leaf nodes of $tr \setminus E$ are labeled by configurations in which O has an abort token. We call $tr \setminus E$ an **abort-tree**.
- 2) At S , R has the power to complete if there is an E such that the leaf nodes of $tr \setminus E$ are labeled by a configurations in which O has a valid contract. We call $tr \setminus E$ a **contract-tree**.

DEFINITION 7.4. The protocol is said to be **balanced for honest O** if for all reachable S and for all bounds on the number of steps that an intruder can take, at S , R does not have both the power to abort and the power to complete.

Note that for the protocol not to be balanced, all we need is to show that R has the power to abort and complete for one bound on the number of steps that the intruder could take (the intruder is in coalition with R).

7.1 Balance for Honest O

We state the following recursive definition:

DEFINITION 7.5. Let S be a reachable configuration and assume that the intruder takes only finitely many steps. Let tr be the continuation tree of S . Let N be a node in tr and X a set of nodes that are children of N such that any edge between N and a member of X is a removable edge. Define N_X to be the set of children of N that are either in X or connected to N by a non-removable edge. N is an **abort-power node** if

- 1) Either it is labeled by a configuration in which O has an abort token, or
- 2) There is an X such that N_X is nonempty and each node in N_X is an abort-power node.

We can define **contract-power node** similarly.

Note that in the definition above, X may be empty, (choosing not to do anything is a valid strategy). Also note that in the above definition, the condition N_X is non-empty to rule out cases such as in which we are at a leaf node and O has a valid contract. Because tr is a finite tree, the above recursive definition makes sense.

For the rest of the section we assume, that S is a reachable configuration, the intruder takes a bounded number of steps and tr is the continuation tree of S with respect to this bound. We have the following connection between the power to abort and abort-nodes.

LEMMA 7.1. At S , R has the power to abort if the root of tr is an abort-power node. Also at S , R has the power to complete if the root of tr is a contract-power node.

Proof By induction on the height of a node in the tree tr , we show that if at S , R has the power to abort then all the nodes in an abort-tree are abort-power nodes. For the other direction, we show how to choose a set of edges in tr , inductively such that $tr \setminus E$ is an abort-power tree.

PROPOSITION 7.2. We have

- 1) If a node N , in tr is labeled by a configuration in which O has an abort_token then it is not a contract-power node.
- 2) If a node N , in tr is labeled by a configuration in which O has a valid contract then it is not an abort-power node.
- 3) Suppose there is a configuration S' , such that every node in tr labeled by S' is not an abort-power node. If N is a node in tr , such that it is labeled by S'' and $S'' \rightarrow S'$ by an application of a rule that labels a non-removable edge, then N is not an abort-power node.
- 4) Suppose there is a configuration S' , such that every node in tr labeled by S' is not a contract-power node. If N is a node in tr , such that it is labeled by S'' and $S'' \rightarrow S'$ by an application of a rule that labels a non-removable edge, then N is not a contract-power node.

PROPOSITION 7.3. Let S be a reachable configuration, and tr the continuation tree of S . Let N be a node in tr , labeled by a configuration S . Suppose further that in S , O is in a state in which it has requested for an abort_token from T and is waiting for a reply, and T has answered a request on the $O-T$ channel. Then either N is not an abort-power node or N is not a contract-power node.

Proof: By lemma 5.4, in S' either an abort_token is on the $O-T$ Channel or a res_cn is on the $O-T$ channel. If the abort_token is on the $O-T$ channel then O can read it by the use of rule O_{abi} . Hence by proposition 7.2, N is not an abort-power node. Similarly, if a resolution from T is on the $O-T$ channel, we have T is not an abort-power token.

PROPOSITION 7.4. Let S be a reachable configuration, and tr the continuation tree of S . Let N be a node in tr , labeled by a configuration S' . Suppose further that in S' , O is in a state in which it has requested for an abort_token from T and is waiting for a reply, and T has yet to answer a request on the $O-T$ channel. Then either N is not an abort-power node or N is not a contract-power node.

Proof: By lemma 5.4 in S , the abort request is on the $O-T$ Channel. Since T has yet not answered a request on the $O-T$ channel in S' , it can answer it and the whole configuration would go into a configuration S'' , in which T has answered a request on the $O-T$ channel by the use of one of the rules in T . The proposition now follows from proposition 7.2 and prop 7.3.

LEMMA 7.5. Let N be a node in tr which is labeled by a configuration S' . Suppose further that in S' , O is in a state in which it has requested for an abort_token or a resolution from T and is waiting for a reply, then either N is not an abort-power node or N is not a contract-power node.

Proof: By propositions 7.3 and 7.4.

LEMMA 7.6. Let N be a node in tr which is labeled by a configuration S' . Suppose further that in S' , O is in a state in which it has requested for a resolution from T and is waiting for a reply, then either N is not an abort-power node or N is not a contract-power node.

Proof: Similar to lemma 7.5 using lemma 5.5 and proposition 7.2.

LEMMA 7.7. *Let N be a node in tr . Then either N is not an abort-power node or N is not a contract-power node.*

Proof: Let N be labeled S' . Since S' is a reachable configuration, we have O must be in one of its 10 states in S' . So the following cases arise:

1. $O_{ab?}(pd, -) \in S'$: By lemma 7.5.
2. $O_1(pd, -) \in S'$: O can request an abort_token from T by the use of rule $O_{ab?}$ and go to a state S'' , such that $O_{ab?}(pd, -) \in S''$. The result follows by case 1 and proposition 7.2.
3. $O_0(pd, -) \in S'$: O can send me_1 by the use rule O_1 and go to a state S'' , such that $O_1(pd, -) \in S''$. The result follows by case 2 and proposition 7.2.
4. $O_{res?}(pd, -) \in S'$: By lemma 7.5.
5. $O_3(pd, -) \in S'$: O can request a resolution from T by the use of rule $O_{res?}$ and go to a state S'' , such that $O_{res?}(pd, -) \in S''$. The result follows by case 4 and proposition 7.2.
6. $O_2(pd, -) \in S'$: O can send me_2 by the use of rule O_3 and go to a state S'' , such that $O_3(pd, -) \in S''$. The result follows by case 5 and proposition 7.2.
7. $O_{ab1}(pd, -) \in S'$ or $O_{ab2}(pd, -) \in S'$: O has an abort_token and the result follows from proposition 7.2.
8. $O_{res1}(pd, -) \in S'$ or $O_{res2}(pd, -) \in S'$ or $O_{com}(pd, -) \in S'$: O has a valid contract and the result follows from proposition 7.2.

THEOREM 7.8. *The protocol is balanced for honest O .*

The above definitions and proofs of balance can be extended to multiple runs of the protocol. There are the following important distinctions in the definition:

- 1) Instead of initial set of facts, Σ , we start with an arbitrary reachable configuration.
- 2) For the construction of a finite continuation tree, along with the number of steps that the intruder steps, we also put a bound the number of times the role generation rule can be used.
- 3) Each edge of the continuation tree is now labeled by the rule being used along with the key of the principal involved.
- 4) Any edge that is labeled by R 's key or a key known to the intruder is also a removable edge.
- 5) The protocol is said to be balanced for honest O , if for all reachable configurations S , and for all bounds on the number of steps that the intruder takes and the number of times the role generation rule is used, at S , R does not have both the power to abort and the power to complete.

7.2 Balance for Honest R

We discuss briefly balance for honest R . Assume that in Σ , two principals O and R whose private/public key pairs are (k_{os}, k_o) , and (k_{rs}, k_r) , agree to sign a contract with contractual text m , unique identifier n , and a trusted third party T , whose private/public key pair is (k_{ts}, k_t) . This they do by the use of role generation rule RG . Let the

resulting configuration be S_0 . For our analysis, we assume that R is honest and O is strongly dishonest. For the rest of the section, by a reachable configuration S , we mean a configuration reachable from Σ without the use of the role generation rule.

Because of effectiveness for honest R , all the leaf nodes in the continuation tree would be labeled by configurations in which R has either quit or has obtained either an abort_token or a valid contract. As before, we say that the protocol has successfully completed if the honest party R obtains a valid contract and aborted if R either quits or obtains an abort_token. Please note that, a dishonest O can obtain both a valid contract and an abort_token. This it can do by first exchanging signatures with R by using the exchange protocol and then requesting T for an abort_token. However, by fairness since R will always obtain a valid contract if O obtains a valid contract, R can always claim that the contract is valid.

DEFINITION 7.6. *Let tr be the continuation tree of S . An edge of tr is said to be a **removable edge** if it is labeled by a rule in the theory of O or by a rule of the intruder or a rule that indicates a message being read from the network (that is a N_i predicate occurs in the precedent of the rule). Otherwise, the edge is said to be **non-removable**.*

DEFINITION 7.7. *Let S be a reachable configuration and assume that the intruder takes a bounded number of steps. Let tr be the continuation tree of S . Let E be a set of removable edges in tr . Define $tr \setminus E$, a **strategy**, to be the tree obtained from tr by removing the edges in E and all of their descendants (including descendants further down).*

1) *At S , O has the power to abort if there is an E such that the leaf nodes of $tr \setminus E$ are labeled by configurations in which either R has quit or has an abort token. We call $tr \setminus E$ an **abort-tree**.*

2) *At S , O has the power to complete if there is an E such that the leaf nodes of $tr \setminus E$ are labeled by a configurations in which R has a valid contract. We call $tr \setminus E$ a **contract-tree**.*

DEFINITION 7.8. *The protocol is said to be **balanced for honest R** if for all reachable S and for all bounds on the number of steps that an intruder can take, at S , O does not have both the power to abort and the power to complete.*

DEFINITION 7.9. *Let S be a reachable configuration and assume that the intruder takes only finitely many steps. Let tr be the continuation tree of S . Let N be a node in tr and X a set of nodes that are children of N such that any edge between N and a member of X is a removable edge. Define N_X to be the set of children of N that are either in X or connected to N by a non-removable edge. N is an **abort-power node** if*

- 1) *Either it is labeled by a configuration in which either R has quit or has an abort token, or*
- 2) *There is an X such that N_X is nonempty and each node in N_X is an abort-power node. We can define **contract-power node** similarly.*

LEMMA 7.9. *At S , R has the power to abort if the root of tr is an abort-power node. Also at S , R has the power to complete if the root of tr is a contract-power node.*

We have

PROPOSITION 7.10. *We have*

- 1) *If a node N , in tr is labeled by a configuration in which either R or has an abort_token then it is not a contract-power node.*
- 2) *If a node N , in tr is labeled by a configuration in which R has a valid contract then it is not an abort-power node.*
- 3) *Suppose there is a configuration S' , such that every node in tr labeled by S' is not an abort-power node. If N is a node in tr , such that it is labeled by S'' and $S'' \rightarrow S'$ by an application of a rule that labels a non-removable edge, then N is not an abort-power node.*
- 4) *Suppose there is a configuration S' , such that every node in tr labeled by S' is not a contract-power node. If N is a node in tr , such that it is labeled by S'' and $S'' \rightarrow S'$ by an application of a rule that labels a non-removable edge, then N is not a contract-power node.*

LEMMA 7.11. *Let N be a node in tr which is labeled by a configuration S' . Suppose further that in S' , R is in a state in which it has requested for a resolution from T and is waiting for a reply, then either N is not an abort-power node or N is not a contract-power node.*

LEMMA 7.12. *Let N be a node in tr . Then either N is not an abort-power node or N is not a contract-power node.*

Proof: Let N be labeled S' . Since S' is a reachable configuration, we have R must be in one of its 8 states in S' . So the following cases arise:

1. $R_{quit}(pd, -) \in S'$: By proposition 7.10, R is not a contract-power node.
2. $R_0(pd, -) \in S'$: R can quit by the use of rule R_{quit} and the result follows from proposition 7.10
3. $R_{res?}(pd, -) \in S'$: By lemma 7.11.
4. $R_2(pd, -) \in S'$: R can ask for a resolution by the use of rule $R_{res?}$ and the result follows from case 3 and proposition 7.10.
5. $R_1(pd, -) \in S'$: R can send me_2 by the use of rule R_2 and the result then follows from case 3 and proposition 7.10.
6. $R_{ab}(pd, -) \in S'$: By proposition 7.10 R is not a contract power node.
7. $R_{res}(pd, -) \in S'$ or $R_{com}(pd, -) \in S'$: R has a valid contract and the result follows from proposition 7.10.
8. $R_3(pd, -) \in S'$: R can send me_4 by the use of the rule the use of rule R_{com} and go to a state S'' , such that $R_{com}(pd, -) \in S''$. The result follows by case 7 and proposition 7.10.

THEOREM 7.13. *The protocol is balanced for honest R .*

Once again the definition and proof can be extended for concurrent runs.

Although we do not discuss it here, balance for honest O in case R is honest or weakly dishonest, and balance for honest R in case O is honest, weakly dishonest can be similarly stated and proved.

7.3 Transparent Channels and Balance

If we relax the condition of transparent channels and allow the intruder to delay messages on the channels between the participants and T , the protocol is no longer balanced. This is because of the following scenario relayed to us by Steve Kremer and Olivier Markowitch:

- 1) O sends its PCS to R . R does not respond.
- 2) O asks T for an abort request, and R sends a resolve request to T .
- 3) To achieve an abort, R delays its resolve request until T acts on the abort request; and to achieve a valid contract, R delays the abort request.

8. REPRESENTATION IN LINEAR LOGIC

Let us indicate how strategies, $tr \setminus E$, considered in the previous section may be faithfully represented as formal derivations in linear logic [11], extending the representation of MSR traces considered in [3].

A *pure Horn axiom* is defined as: $X(\bar{x}, \bar{a}) \multimap \exists \bar{y} Y(\bar{x}, \bar{y}, \bar{b})$. Any of its ground instances of the form: $X(\bar{t}, \bar{a}) \multimap \exists \bar{y} Y(\bar{t}, \bar{y}, \bar{b})$, is conceived of as an *instruction* to transform a configuration of the form: $\dots \otimes X(\bar{t}, \bar{a}) \otimes \dots$ into the configuration: $\dots \otimes Y(\bar{t}, \bar{d}, \bar{b}) \otimes \dots$ where \bar{d} is the vector of fresh constants.

A configuration of a system is represented as a tensor of finitely many atomic formulas. The case where all possible actions of a system should be taken into account is described with the help of *branching Horn formulas* of the form $X \multimap (Y_1 \oplus Y_2 \oplus \dots \oplus Y_m)$. The *firing* of such an instruction results in the change of the “state” X either into the “state” Y_1 , or into the “state” Y_2 , or \dots , or into the “state” Y_m . But we do not know *in advance* which of these m alternatives will be chosen at a given occasion [12, 13]. The case where all possible actions Y_2, \dots, Y_m should be taken into account, with providing our own choice between $Y_{1,1}, \dots, Y_{1,k}$ will be described by means of a *Horn-like formula* of the form $X \multimap ((Y_{1,1} \& \dots \& Y_{1,k}) \oplus Y_2 \oplus \dots \oplus Y_m)$.

DEFINITION 8.1. *Let Γ be a set of Horn clauses. A strategy \mathcal{S} in accordance with Γ is a labeled rooted tree of configurations such that each of its edges (v, w) is labeled by an ‘instruction’ specified within Γ , which transforms v into w .*

More precisely, for each vertex v with exactly $k \geq 1$ sons w_1, w_2, \dots, w_k , the outgoing edges $(v, w_1), (v, w_2), \dots, (v, w_k)$ are labeled by pure Horn clauses with one and the same antecedent: $X \multimap Y_{1,i}, X \multimap Y_2, \dots, X \multimap Y_k$, resp., - these clauses being such that there exists a Horn clause of the form $X \multimap ((Y_{1,1} \& \dots \& Y_{1,i} \& \dots \& Y_{1,k}) \oplus Y_2 \oplus \dots \oplus Y_m)$, which belongs to Γ . (Thus we represent all possible effects at v).

We extend the *comprehensive* computational interpretation for Horn linear logic introduced in [12, 13].

THEOREM 8.1 (STRATEGIES \implies PROOFS). *Let Γ be a set consisting of Horn-like formulas. Let $W(a_1, \dots, a_p)$ and $Z(b_1, \dots, b_q)$ be closed elementary products, where $a_1, \dots, a_p, b_1, \dots, b_q$ are constants, and each of the constants b_1, \dots, b_q occurs either in Γ or in $W(a_1, \dots, a_p)$.*

Then any strategy \mathcal{S} in accordance with Γ , which, whatever its branch b we take, leads from $W(a_1, \dots, a_p)$ to $Z(b_1, \dots, b_q)$, can be transformed into a linear logic derivation of a sequent of the form: $!\Gamma, W(a_1, \dots, a_p) \vdash$

$Z(b_1, \dots, b_q)$. Here $!\Gamma$ stands for the set resulting from putting $!$ before each formula in Γ .

THEOREM 8.2 (PROOFS \implies STRATEGIES). *Let Γ be a set consisting of Horn formulas. Suppose there are elementary products $W(a_1, \dots, a_p)$ and $Z(b_1, \dots, b_q)$, such that a sequent of the form $!\Gamma, W(a_1, \dots, a_p) \vdash Z(b_1, \dots, b_q)$, is derivable in linear logic, then there exists a strategy \mathcal{S} in accordance with Γ , such that its root is $W(a_1, \dots, a_p)$, and each of the terminal vertices is of the form $Z(b_1, \dots, b_q)$.*

Let us apply this approach to the case of the previous section where a strongly dishonest participant R is playing against the group G of “others”, which consists of O , T . Suppose that $\alpha_1, \alpha_2, \dots, \alpha_k$ is a list of all R 's rules; the intruder rules; and all rules in which N_i predicates occur on the list; and the list $\beta_2, \beta_3, \dots, \beta_m$ includes all the rules of the opponents from G .

Because of space constraints, we sketch the main points of our linear logic encoding. We describe the case of a single run with an honest O and a strongly dishonest R . The case of a single run with an honest O and weakly dishonest R can be similarly described.

Auxiliary control variables: $q, q_{1,1}, \dots, q_{1,k}, q_2, \dots, q_m$, are introduced. We form Γ in the following way. Each of the above α_i of the form $(X \multimap Y)$ is “guarded” as: $((q_{1,i} \otimes X) \text{mbox}\multimap (q \otimes Y))$. Each β_j of the form $(X' \multimap Y')$ is “guarded” as: $((q_j \otimes X') \multimap (q \otimes Y'))$. And we add the following axiom: $q \multimap ((q_{1,1} \& \dots \& q_{1,k}) \oplus q_2 \oplus \dots \oplus q_m)$. In addition, we have to introduce a number of Horn axioms to handle the cases where certain enabling conditions are violated (a similar situation with the zero-test simulation is handled, for instance, in [13]).

Theorems 8.1 and 8.2 provide us with the following

THEOREM 8.3. *Given a “goal” Z_0 , R has a power to get Z_0 if and only if a sequent of the form: $!\Gamma, q \vdash Z_0$, is derivable in linear logic.*

9. CONCLUSIONS AND FURTHER WORK

We have studied an optimistic two-party contract-signing protocol derived from the Garay-Jakobsson-MacKenzie (GJM) protocol [10] and the notion of abuse-free contract signing introduced in [10]. In a version suggested in [15], a fair contract-signing protocol is abuse-free if, at any stage of the protocol, any protocol participant does not have both the power to complete the contract as well as the power to abort it. We have used a multiset-rewriting formalism for protocol analysis [3] to formally state this property in terms of a certain recursive property of the protocol execution tree, which we then proved for our version of the GJM protocol by inductive methods. Our proof relies on a strong notion of fairness adopted from [10], which itself we formally stated in the multiset-rewriting formalism and proved by inductive methods. We have also showed that a version of abuse-freeness may be represented in terms of provability in a logical system, in which formal derivations correspond to full execution trees and vice versa.

Other possible forms of abuse-freeness remain to be investigated. Other desirable properties of the GJM protocol, such as non-repudiation and the trusted third party accountability, still remain to be investigated as well. The latter

will involve formalizing potentially dishonest or careless actions of the trusted third party. We also plan to apply our techniques to analyze other, multi-party optimistic contract-signing protocols presented in [10] and to prove their properties, including abuse-freeness.

10. REFERENCES

- [1] N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *IEEE Symposium on Security and Privacy*, pages 86–99, 1998.
- [2] I. Cervesato. Typed MSR: Syntax and examples. In *WITS'00. Workshop on Issues in the Theory of Security, 2000.*, 2000.
- [3] I. Cervesato, N.A. Durgin, P.D. Lincoln, J.C. Mitchell, and A. Scedrov. A meta-notation for protocol analysis. In *Proc. 12-th Annual IEEE Computer Security Foundations Workshop — CSFW'99*, pages 55–69, Mordano, Italy, 1999. IEEE Computer Society Press.
- [4] V. Cortier, J. Millen, and H. Ruess. Proving secrecy is easy enough. In *14-th Annual IEEE Computer Security Foundations Workshop'01*, pages 97–108, 2001.
- [5] S. Das and D. Dill. Successive approximation of abstract transition relations. In *Sixteenth Annual IEEE Symposium on Logic in Computer Science*, pages 51–58, 2001.
- [6] D. Dolev and A. Yao. On the security of public-key protocols. In *Proc. 22nd Annual IEEE Symp. Foundations of Computer Science*, pages 350–357, 1981.
- [7] S. Even. Protocol for signing contracts. In *CRYPTO*, pages 148–153, 1981.
- [8] S. Even and Y. Yacobi. Tr 175. pages 148–153, Computer Science Dept, Technion, Israel, March, 1980.
- [9] M.J. Fischer, N.A. Lynch, and M.S. Paterson. Impossibility of distributed consensus with one faulty process. *JACM*, 32(2):374–382, 1985.
- [10] J. Garay, M. Jakobsson, and P. MacKenzie. Abuse-free optimistic contract signing. In *Advances in Cryptology – CRYPTO'99*, pages 449–466. Springer Lecture Notes in Computer Science, vol. 1666, 1999.
- [11] J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
- [12] M. Kanovich. Linear logic as a logic of computations. *Annals of Pure and Applied Logic*, 67:183–212, 1994.
- [13] M. Kanovich. The direct simulation of Minsky machines in linear logic. In *Advances in Linear Logic*, volume 222, pages 123–145, London Mathematical Society Lecture Notes, 1995. Cambridge University Press.
- [14] S. Kremer and J.-F. Raskin. Formal verification of non-repudiation protocols - a game approach. In *Formal Methods for Computer Security (FMCS 2000)*, Chicago, USA, July 2000.
- [15] J.C. Mitchell and V. Shmatikov. Analysis of abuse-free contract signing. In *Financial Cryptography '00*, 2000.
- [16] R.M. Needham and M.D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

- [17] L. Paulson. Proving properties of security protocols by induction. In *Proc. 10th Computer Security Foundations Workshop*, pages 70–83. IEEE Computer Society Press, 1997.
- [18] V. Shmatikov and J.C. Mitchell. Analysis of a fair exchange protocol. In *FLoC Workshop on Formal Methods and Security Protocols*, 1999.
- [19] P. Syverson, C. Meadows, and I. Cervesato. Dolev-yao is no better than machiavelli. In *WITS'00. Workshop on Issues in the Theory of Security, 2000.*, 2000.
- [20] V.Shmatikov and J.C.Mitchell. Finite-state analysis of two contract signing protocols. In *To appear in special issue of TCS on computer security*, 2001.
- [21] T.Y.C. Woo and S.S. Lam. A semantic model for authentication protocols. In *Proc. IEEE Symposium on Research in Security and Privacy*, 1993.