Deciding Properties of Contract-Signing Protocols

Detlef Kähler, Ralf Küsters, and Thomas Wilke

Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität zu Kiel
24098 Kiel, Germany
{kaehler,kuesters,wilke}@ti.informatik.uni-kiel.de

Abstract. We show that for infinite transition systems induced by cryptographic protocols in the Rusinowitch/Turuani style certain fundamental branching properties are decidable. As a consequence, we obtain that crucial properties of contract-signing protocols such as balance are decidable.

1 Introduction

There has been intensive research on the automatic analysis of cryptographic protocols in the recent past (see, e.g., [8, 13] for an overview) which led to industrial-strength debugging tools (see, e.g., [2]). One of the central results of the area is that security of cryptographic protocols is decidable when analyzed w.r.t. a finite number of sessions, without a bound on the message size, and in presence of the so-called Dolev-Yao intruder (see, e.g., [15, 1, 5, 14]). This result (and all the related ones) is, however, restricted to security properties such as authenticity and secrecy, which are reachability properties of the transition system associated with a given protocol: Is a state, in which the intruder possesses a certain secret, such as a session key, reachable? In contrast, crucial properties required of contract-signing and related protocols [10, 3, 4, 17], for instance abuse-freeness [10] and balance [6], are properties of the structure of the transition system associated with a protocol. Balance, for instance, requires that in no stage of a protocol run, the intruder or a dishonest party has both a strategy to abort the run and a strategy to successfully complete the run and thus obtain a valid contract.

The main goal of this paper is to show that the central result mentioned above extends to branching properties, such as balance, and similar properties of contract-signing protocols. In other words, we want to show that these branching properties are decidable w.r.t. a finite number of sessions, without a bound on the message size, and in presence of the so-called Dolev-Yao intruder. This can potentially lead to fully automatic analyzers for contract-signing protocols that are much more precise than the existing ones, which consider only drastically scaled down finite-state versions of the protocols in question.

The protocol and intruder model that we suggest to use is a "finite session" version of a model proposed in [6].[1] It contains different features important for contract-signing protocols which are absent in the models for authentication and key exchange protocols referred to above. First, as in [6], we include private contract signatures in our model as an important example of a cryptographic primitive used in contract-signing protocols, such as the protocol proposed in [10]. Second, as in [6], we model write-protected channels which are *not* under the control of the intruder. In this paper, we call these channels *secure channels* for simplicity, although this notion is also used in cryptography with a different meaning. Third, for protocols expressed in our model we explicitly define the induced transition systems. These transition systems have infinitely many states and are infinitely branching, but have paths of bounded length, and allow us to state crucial properties of contract-signing properties.

Our main technical result is that for the transition systems induced by cryptographic protocols certain game-theoretic properties—expressing the existence of certain strategies of the intruder—are decidable. From this we obtain that balance is decidable for contract-signing protocols. We also show that reachability properties slightly more general than those mentioned above are decidable and conclude that other important properties of contract signing protocols, namely effectiveness and fairness, are decidable as well.

The basic technique used in our proofs is the same as the one first introduced in [15], where they show that to find an attack on a protocol only reasonably small substitutions have to be considered. For our framework, we extend and modify this idea appropriately.

In several papers, contract-signing and related protocols have been analyzed using finite-state model checkers (see, e.g., [16, 12]). Due to the restriction to a finite state set, the Dolev-Yao intruder is, however, only approximated. A much more detailed model has been considered by Chadha et al. [6], who analyzed the contract-signing protocol proposed by Garay, Jakobsson, and MacKenzie [10], even taking into account an unbounded number of sessions. However, the analysis was carried out by hand and without tool support. As mentioned, our model is the "finite session" version of the model by Chadha et al. Hence, our results show that when restricted to a finite number of sessions, the analysis carried out in [6] can be fully automated (given a specification of the protocols) without resorting to finite-state models as in [16, 12]. Drielsma and Mödersheim [9] apply an automatic tool originally intended for authentication and key exchange protocols in the analysis of the Asokan-Shoup-Waidner (ASW) protocol [3]. Their analysis is, however, restricted to reachability properties as branching properties cannot be handled by their tool. Also, secure channels are not modeled explicitly in that paper.

Structure of this paper. In Section 2, we introduce our protocol and intruder model, with an example provided in Section 3. In Section 4, we present our main

[1] Our technical exposition though is closer to the term-rewriting approach [15] than to the multi-set rewriting framework employed in [6].

technical result, stating that certain game-theoretic properties of transition systems induced by cryptographic protocols are decidable. In Section 5, this result is applied to contract-signing protocols. Due to the page limit, private contract signatures and reachability properties are only dealt with in our technical report [11], which also includes the full proofs of our results.

2 The Protocol and Intruder Model

As mentioned in the introduction, in essence, our model is the "finite session" version of the model proposed in [6]. When it comes to the technical exposition, our approach is, however, inspired by the term-rewriting approach of [15] rather than the multi-set rewriting approach of [6].

In our model, a protocol is a finite set of principals and every principal is a finite tree, which represents all possible behaviours of the principal. Each edge of such a tree is labeled by a rewrite rule, which describes the receive-send action that is performed when the principal takes this edge in a run of the protocol.

When a principal carries out a protocol, it traverses its tree, starting at the root. In every node, the principal takes its current input, chooses one of the edges leaving the node, matches the current input with the left-hand side of the rule the edge is labeled with, sends out the message which is determined by the right-hand side of the rule, and moves to the node the chosen edge leads to. Whether or not a principal gets an input and which input it gets is determined by the secure channel and the intruder (see below), who receives every message sent by a principal, can use all the messages he has received to construct new messages, and can provide input messages to any principal he wants—this is the usual Dolev-Yao model (see, e.g., [15]).

The above is very similar to the approach in [15]. There are, however, three important ingredients that are not present in [15]: secure channels, explicit branching structure, and certain cryptographic primitives relevant to contract-signing protocols, such as private contract signatures. In the following, we deal with secure channels and the branching structure. Private contract signatures are only dealt with in the technical report [11].

Secure channels. Unlike in the standard Dolev-Yao model, in our model the input of a principal may not only come from the intruder but also from a so-called secure channel. While a secure channel is not read-protected (the intruder can read the messages written onto this channel), the intruder does not control this channel. That is, he cannot delay, duplicate, or remove messages, or write messages onto this channel under a fake identity (unless he has corrupted a party).

Branching structure. As mentioned in the introduction, unlike authentication and key-exchange protocols, properties of contract-signing and related protocols cannot be stated as reachability properties, i.e., in terms of single runs of a protocol alone. One rather has to consider branching properties. We therefore describe the behavior of a protocol as an infinite-state transition system which comprises all runs of a protocol. To be able to express properties of contract-

signing protocols we distinguish several types of transitions: there are intruder transitions (just as in [15]), there are ε-transitions, which can be used to model that a subprotocol is spawned without waiting for input from the intruder, and secure channel transitions, which model communication via secure channels.

2.1 Terms and Messages

We have a finite set $\mathcal{V}$ of variables, a finite set $\mathcal{A}$ of atoms, a finite set $\mathcal{K}$ of public and private keys, an infinite set $\mathcal{A}_I$ of intruder atoms, and a finite set $\mathcal{N}$ of principal addresses. All of them are assumed to be disjoint.

The set $\mathcal{K}$ is partitioned into a set $\mathcal{K}_{pub}$ of public keys and a set $\mathcal{K}_{priv}$ of private keys. There is a bijective mapping $\cdot^{-1}\colon \mathcal{K} \to \mathcal{K}$ which assigns to every public key the corresponding private key and to every private key its corresponding public key.

Typically, the set $\mathcal{A}$ contains names of principals, atomic symmetric keys, and nonces (i.e., random numbers generated by principals). We note that we will allow non-atomic symmetric keys as well. The atoms in $\mathcal{A}_I$ are the nonces, symmetric keys, etc. the intruder may generate. The elements of $\mathcal{N}$ are used as addresses of principals in secure channels.

We define two kinds of terms by the following grammar, namely *plain terms* and *secure channel terms*:

$$\begin{aligned} \textit{plain-terms} ::= {} & \mathcal{V} \mid \mathcal{A} \mid \mathcal{A}_I \mid \langle \textit{plain-terms}, \textit{plain-terms} \rangle \mid \{\textit{plain-terms}\}^s_{\textit{plain-terms}} \mid \\ & \{\textit{plain-terms}\}^a_{\mathcal{K}} \mid \mathsf{hash}(\textit{plain-terms}) \mid \mathsf{sig}_{\mathcal{K}}(\textit{plain-terms}) \\ \textit{sec-terms} ::= {} & \mathsf{sc}(\mathcal{N}, \mathcal{N}, \textit{plain-terms}) \\ \textit{terms} ::= {} & \textit{plain-terms} \mid \textit{sec-terms} \mid \mathcal{N} \end{aligned}$$

Plain terms, secure channel terms, and terms without variables (i.e., ground terms) are called *plain messages*, *secure channel messages*, and *messages*, respectively. As usual, $\langle t, t' \rangle$ is the pairing of t and t', the term $\{t\}^s_{t'}$ stands for the symmetric encryption of t by t' (note that the key t' may be any plain term), $\{t\}^a_k$ is the asymmetric encryption of t by k, the term $\mathsf{hash}(t)$ stands for the hash of t, and $\mathsf{sig}_k(t)$ is the signature on t which can be verified with the public key k.

A secure channel term of the form $\mathsf{sc}(n, n', t)$ stands for feeding the secure channel from n to n' with t. A principal may only generate such a term if he knows n and t (but not necessarily n'). This guarantees that a principal cannot impersonate other principals on the secure channel. Knowing n grants access to secure channels with sender address n.

A *substitution* assigns terms to variables. The *domain* of a substitution is denoted by $\mathsf{dom}(\sigma)$ and defined by $\mathsf{dom}(\sigma) = \{x \in \mathcal{V} \mid \sigma(x) \neq x\}$. Substitutions are required to have finite domains and it is required that $\sigma(x)$ is a ground term for each $x \in \mathsf{dom}(\sigma)$. Given two substitutions σ and σ' with disjoint domains, their union $\sigma \cup \sigma'$ is defined in the obvious way. Given a term t, the term $t\sigma$ is obtained from t by simultaneously substituting each variable x occurring in t by $\sigma(x)$.

2.2 Principals and Protocols

Principal rules are of the form $R \Rightarrow S$ where R is a term or ε and S is a term.

A *rule tree* $\Pi = (V, E, r, \ell)$ is a finite tree rooted at $r \in V$ where ℓ maps every edge $(v, v') \in E$ of Π to a principal rule $\ell(v, v')$. A rule tree $\Pi = (V, E, r, \ell)$ is called a *principal* if every variable occurring on the right-hand side of a principal rule $\ell(v, v')$ also occurs on the left-hand side of $\ell(v, v')$ or on the left-hand side of a principal rule on the path from r to v.

For $v \in V$, we write $\Pi{\downarrow}v$ to denote the subtree of Π rooted at v. For a substitution σ, we write $\Pi\sigma$ for the principal obtained from Π by substituting all variables x occurring in the principal rules of Π by $\sigma(x)$.

A *protocol* $P = ((\Pi_1, \ldots, \Pi_n), \mathcal{I})$ consists of a finite set of principals and a finite set $\mathcal{I}$ of messages, the *initial intruder knowledge*. We require that each variable occurs in the rules of only one principal, i.e., different principals must have disjoint sets of variables. We assume that intruder atoms, i.e., elements of $\mathcal{A}_I$, do not occur in P.

2.3 Intruder

Given a set $\mathcal{I}$ of general messages, the (infinite) set $d(\mathcal{I})$ of general messages the intruder can derive from $\mathcal{I}$ is the smallest set satisfying the following conditions:

1. $\mathcal{I} \subseteq d(\mathcal{I})$.
2. *Composition and decomposition*: If $m, m' \in d(\mathcal{I})$, then $\langle m, m' \rangle \in d(\mathcal{I})$. Conversely, if $\langle m, m' \rangle \in d(\mathcal{I})$, then $m \in d(\mathcal{I})$ and $m' \in d(\mathcal{I})$.
3. *Symmetric encryption and decryption*: If $m, m' \in d(\mathcal{I})$, then $\{m\}^s_{m'} \in d(\mathcal{I})$. Conversely, if $\{m\}^s_{m'} \in d(\mathcal{I})$ and $m' \in d(\mathcal{I})$, then $m \in d(\mathcal{I})$.
4. *Asymmetric encryption and decryption*: If $m \in d(\mathcal{I})$ and $k \in d(\mathcal{I}) \cap \mathcal{K}$, then $\{m\}^a_k \in d(\mathcal{I})$. Conversely, if $\{m\}^a_k \in d(\mathcal{I})$ and $k^{-1} \in d(\mathcal{I})$, then $m \in d(\mathcal{I})$.
5. *Hashing*: If $m \in d(\mathcal{I})$, then $\mathsf{hash}(m) \in d(\mathcal{I})$.
6. *Signing*: If $m \in d(\mathcal{I}), k^{-1} \in d(\mathcal{I}) \cap \mathcal{K}$, then $\mathsf{sig}_k(m)$. (The signature contains the public key but can only be generated if the corresponding private key is known.)
7. *Writing onto and reading from the secure channel:* If $m \in d(\mathcal{I})$, $n \in d(\mathcal{I}) \cap \mathcal{N}$, and $n' \in \mathcal{N}$, then $\mathsf{sc}(n, n', m) \in d(\mathcal{I})$. If $\mathsf{sc}(n, n', m) \in d(\mathcal{I})$, then $m \in d(\mathcal{I})$.
8. *Generating fresh constants*: $\mathcal{A}_I \subseteq d(\mathcal{I})$.

Each of the above rules only applies when the resulting expression is a term according to the grammar stated above. For instance, a hash of a secure channel term is not a term, so rule 5 does not apply when m is of the form $\mathsf{sc}(n, n', m')$.

Intuitively, $n \in d(\mathcal{I}) \cap \mathcal{N}$ means that the intruder has corrupted the principal with address n and therefore can impersonate this principal when writing onto the secure channel. Also, the intruder can extract m from $\mathsf{sc}(n, n', m)$ since, just as in [6], the secure channel is not read-protected. (However, our results hold independent of whether or not the secure channel is read-protected.)

2.4 The Transition Graph Induced by a Protocol

We define the transition graph $\mathcal{G}_P$ induced by a protocol P and start with the definition of the states and the transitions between these states.

A *state* is of the form $((\Pi_1, \ldots, \Pi_n), \sigma, \mathcal{I}, \mathcal{S})$ where

1. σ is a substitution,
2. for each i, Π_i is a rule tree such that $\Pi_i\sigma$ is a principal,
3. $\mathcal{I}$ is a finite set of messages, the *intruder knowledge*, and
4. $\mathcal{S}$ is a finite multi-set of secure channel messages, the *secure channel*.

The idea is that when the transition system gets to such a state, then the substitution σ has been performed, the accumulated intruder knowledge is what can be derived from $\mathcal{I}$, the secure channels hold the messages in $\mathcal{S}$, and for each i, Π_i is the "remaining protocol" to be carried out by principal i. This also explains why $\mathcal{S}$ is a multi-set: messages sent several times should be delivered several times.

Given a protocol $P = ((\Pi_1, \ldots, \Pi_n), \mathcal{I})$ the *initial state of* P is set to be $((\Pi_1, \ldots, \Pi_n), \sigma, \mathcal{I}, \emptyset)$ where σ is the substitution with empty domain.

We have three kinds of transitions: intruder, secure channel, and ε-transitions. In what follows, let $\Pi_i = (V_i, E_i, r_i, \ell_i)$ and $\Pi_i' = (V_i', E_i', r_i', \ell_i')$ denote rule trees. We define under which circumstances there is a transition

$$((\Pi_1, \ldots, \Pi_n), \sigma, \mathcal{I}, \mathcal{S}) \xrightarrow{\tau} ((\Pi_1', \ldots, \Pi_n'), \sigma', \mathcal{I}', \mathcal{S}')$$

with τ an appropriate label.

1. *Intruder transitions:* The above transition with label i, m, I exists if there exists $v \in V_i$ with $(r_i, v) \in E_i$ and $\ell_i(r_i, v) = R \Rightarrow S$, and a substitution σ'' of the variables in $R\sigma$ such that
   (a) $m \in d(\mathcal{I})$,
   (b) $\sigma' = \sigma \cup \sigma''$,
   (c) $R\sigma' = m$,
   (d) $\Pi_j' = \Pi_j$ for every $j \neq i$, $\Pi_i' = \Pi_i{\downarrow}v$,
   (e) $\mathcal{I}' = \mathcal{I} \cup \{S\sigma'\}$,
   (f) $\mathcal{S}' = \mathcal{S}$ if $S \neq \mathsf{sc}(\cdot, \cdot, \cdot)$, and $\mathcal{S}' = \mathcal{S} \cup \{S\sigma'\}$ otherwise.
   This transition models that principal i reads the message m from the intruder (i.e., the public network).
2. *Secure channel transitions:* The above transition with label i, m, sc exists if there exists $v \in V_i$ with $(r_i, v) \in E_i$ and $\ell_i(r_i, v) = R \Rightarrow S$, and a substitution σ'' of the variables in $R\sigma$ such that $m \in \mathcal{S}$, (b)–(e) from 1., and $\mathcal{S}' = \mathcal{S} \setminus \{m\}$ if $S \neq \mathsf{sc}(\cdot, \cdot, \cdot)$, and $\mathcal{S}' = (\mathcal{S} \setminus \{m\}) \cup \{S\sigma'\}$ otherwise.
   This transition models that principal i reads message m from the secure channel.
3. *ε-transitions:* The above transition with label i exists if there exists $v \in V_i$ with $(r_i, v) \in E_i$ and $\ell_i(r_i, v) = \varepsilon \Rightarrow S$ such that $\sigma' = \sigma$ and (d), (e), (f) from above.
   This transition models that i performs a step where neither a message is read from the intruder nor from the secure channel.

If $q \xrightarrow{\tau} q'$ is a transition where the first component of the label τ is i, then the transition is called an *i-transition* and q' an *i-successor* of q.

Given a protocol P, the *transition graph $\mathcal{G}_P$ induced by P* is the tuple (S_P, E_P, q_P) where q_P is the initial state of P, S_P is the set of states reachable from q_P by a sequence of transitions, and E_P is the set of all transitions among states in S_P. Formally, a transition $q \xrightarrow{\tau} q'$ is a tuple (q, τ, q').

We write $q \in \mathcal{G}_P$ if q is a state in $\mathcal{G}_P$ and $q \xrightarrow{\tau} q' \in \mathcal{G}_P$ if $q \xrightarrow{\tau} q'$ is a transition in $\mathcal{G}_P$.

Remark 1. The transition graph $\mathcal{G}_P$ of P is a DAG since by performing a transition the size of the first component of a state decreases. While the graph may be infinite branching, the maximal length of a path in this graph is bounded by the total number of edges in the principals Π_i of P.

For a state q, we denote the subgraph of $\mathcal{G}_P$ consisting of all states reachable from q by $\mathcal{G}_{P,q}$.

3 Modeling the Originator of the ASW Protocol

To demonstrate that our framework can actually be used to analyze contract-signing protocols, we show how the originator of the Asokan-Shoup-Waidner (ASW) protocol [3] can be modeled. In a similar fashion, other contract-signing protocols, such as the Garay-Jakobsson-MacKenzie protocol [10], can be dealt with.

3.1 Overview of the Protocol

Our informal description of the ASW protocol follows [16] (see this work or [3] for more details). For ease in notation, we will write $\mathsf{sig}[m, k]$ instead of $\langle m, \mathsf{sig}_k(m)\rangle$.

The ASW protocol enables two principals O (originator) and R (responder) to obtain each other's commitment on a previously agreed contractual text, say text, with the help of a trusted third party T, which, however, is only invoked in case of problems. In other words, the ASW protocol is an optimistic two-party contract-signing protocol.

There are two kinds of valid contracts specified in the ASW protocol: the standard contract, $\langle \mathsf{sig}[m_O, k_O], N_O, \mathsf{sig}[m_R, k_R], N_R\rangle$, and the replacement contract, $\mathsf{sig}[\langle \mathsf{sig}[m_O, k_O], \mathsf{sig}[m_R, k_R]\rangle, k_T]$, where k_T is the key of the trusted third party, $m_O = \langle k_O, k_R, k_T, \mathsf{text}, \mathsf{hash}(N_O)\rangle$, and $m_R = \langle m_O, \mathsf{hash}(N_R)\rangle$. The keys k_O, k_R, and k_T are used for identifying the principals. Note that a signed contractual text ($\mathsf{sig}[\mathsf{text}, k_O]$ or $\mathsf{sig}[\mathsf{text}, k_R]$) is not considered a valid contract.

The ASW protocol consists of three subprotocols: the exchange, abort, and resolve protocol. However, we can describe every principal—O, R, and T—in terms of a single tree as introduced in Section 2.2.

The basic idea of the exchange protocol is that O first indicates his/her interest to sign the contract. To this end, O hashes a nonce N_O and signs it together

with text and the keys of the principals involved. The resulting message is the message m_O from above. By sending it to R, O commits to the contract. Then, similarly, R indicates his/her interest to sign the contract by hashing a nonce N_R and signing it together with text and the keys of the involved principals. This is the message m_R from above. By sending it to O, R commits to the contract. Finally, first O and then R reveal N_O and N_R, respectively. This is why a standard contract is only valid if N_O and N_R are included.

If, after O has sent the first message, R does not respond, O may contact T to abort. At any point, if one of O and R does not respond, the other may contact T to resolve. In case the protocol is successfully resolved, the replacement contract $\mathsf{sig}[\langle m_O, m_R\rangle, k_T]$ is issued. While this version of the contract only contains the message indicating O's and R's intention to sign the contract (and neither N_O nor N_R), the signature of T validates the contract.

In the next subsection, the model of O is presented. The models for R and T as well as the security properties for the ASW protocol can be found in the technical report [11].

3.2 The Principal O

The principal O is defined by the tree Π_O depicted in Figure 1 where the numbers stand for the principal rules defined below. Rules 1, 2, and 3 belong to the exchange protocol, rules 4, 5, and 6 belong to the abort protocol, and rules 7, 8, and 9 belong to the resolve protocol.

Exchange protocol. The actions performed in the exchange protocol have informally been discussed above.

Abort protocol. If, after the first step of the exchange protocol, O does not get an answer back from R, the principal O may start the abort protocol, i.e., send an abort request via a secure channel to T (rule 4). Then, T will either confirm the abort of the protocol by returning an abort token—in this case O will continue with rule 5—or send a resolve token—in this case O will continue with rule 6. (The trusted third party T sends a resolve token if R previously contacted T to resolve the protocol run.)

Resolve protocol. If after rule 2, i. e., after sending N_O, the principal O does not get an answer back from R, then O can start the resolve protocol by sending a resolve request to T via the secure channel (rule 7). After that, depending on the answer returned from T (which again will return an abort or resolve token), one of the rules 8 or 9 is performed.

We now present the principal rules for O where the numbering corresponds to the one in Figure 1. Any occurrence of _ should be substituted by a new fresh variable, that is, the term which is matched is not used afterwards.

1. $\varepsilon \Rightarrow me_1$ where

$$me_1 = \mathsf{sig}[me_2, k_O] \quad \text{and} \quad me_2 = \langle k_O, k_R, k_T, \mathsf{text}, \mathsf{hash}(N_O)\rangle\,.$$

2. $\mathsf{sig}[me_3, k_R] \Rightarrow N_O$ where $me_3 = \langle me_1, \mathsf{hash}(x)\rangle$.
3. $x \Rightarrow \mathsf{OHasValidContract}$.

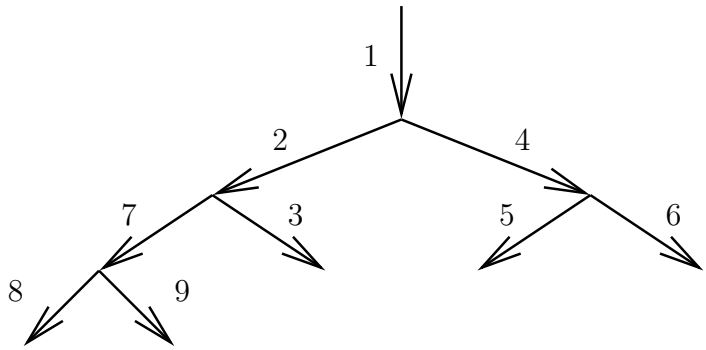

Fig. 1. The Originator O

4. $\varepsilon \Rightarrow \mathsf{sc}(O, T, ma_1)$ where $ma_1 = \mathsf{sig}[\langle \mathsf{aborted}, me_1 \rangle, k_O]$.
5. $\mathsf{sc}(T, O, ma_2) \Rightarrow \mathsf{OHasValidContract}$ where

$$ma_2 = \mathsf{sig}[\langle me_1, me_4 \rangle, k_T] \quad \text{and} \quad me_4 = \mathsf{sig}[\langle me_1, \_ \rangle, k_R].$$

6. $\mathsf{sc}(T, O, \mathsf{sig}[\langle \mathsf{aborted}, ma_1 \rangle, k_T]) \Rightarrow \mathsf{OHasAbortToken}$.
7. $\varepsilon \Rightarrow \mathsf{sc}(O, T, \langle me_1, \mathsf{sig}[me_3, k_R] \rangle)$.
8. $\mathsf{sc}(T, O, \mathsf{sig}[\langle \mathsf{aborted}, ma_1 \rangle, k_T]) \Rightarrow \mathsf{OHasAbortToken}$.
9. $\mathsf{sc}(T, O, mr_1) \Rightarrow \mathsf{OHasValidContract}$ where

$$mr_1 = \mathsf{sig}[\langle me_1, mr_2 \rangle, k_T] \quad \text{and} \quad mr_2 = \mathsf{sig}[\langle me_1, \_ \rangle, k_R].$$

4 Main Result

As indicated in the introduction, our main result states that for the transition graphs induced by cryptographic protocols certain game-theoretic properties—expressing the existence of certain strategies of the intruder—are decidable. In what follows we formalize this.

We first define the notion of a strategy graph, which captures that the intruder has a way of acting such that regardless of how the other principals act, he achieves a certain goal, where goal in our context means that a state will be reached where the intruder can derive certain constants and cannot derive others.

A q-strategy graph $\mathcal{G}_q$ is a sub-transition system of $\mathcal{G}_P$ where q is the initial state of $\mathcal{G}_q$ and such that for all states q' in $\mathcal{G}_q$, the following conditions, which are explained below, are satisfied.

1. If $q' \xrightarrow{j} q'' \in \mathcal{G}_P$, then $q' \xrightarrow{j} q'' \in \mathcal{G}_q$ for every j and q''.
2. If $q' \xrightarrow{j,m,\mathsf{sc}} q'' \in \mathcal{G}_P$, then $q' \xrightarrow{j,m,\mathsf{sc}} q'' \in \mathcal{G}_q$ for every m, j, and q''.
3. If $q' \xrightarrow{j,m,I} q'' \in \mathcal{G}_q$ and $q' \xrightarrow{j,m,I} q''' \in \mathcal{G}_P$, then $q' \xrightarrow{j,m,I} q''' \in \mathcal{G}_q$ for every m, j, q'', and q'''.

The first condition says that every ε-transition of the original transition system must be present in the strategy graph; this is because the intruder should not be able to prevent a principal from performing an ε-rule. The second condition is similar: the intruder should not be able to block secure channels. The third condition says that although the intruder can choose to send a particular message to a particular principal, he cannot decide which transition this principal uses (if the message matches two rules).

A *strategy property* is a tuple $((C_1, C_1'), \ldots, (C_s, C_s'))$, where $C_i, C_i' \subseteq \mathcal{A} \cup \mathcal{K} \cup \mathcal{N}$. A state q satisfies $((C_1, C_1'), \ldots, (C_s, C_s'))$ if there exist q-strategy graphs $\mathcal{G}_1, \ldots, \mathcal{G}_s$ such that every $\mathcal{G}_i$ satisfies (C_i, C_i'), where $\mathcal{G}_i$ satisfies (C_i, C_i') if for all leaves v_i of $\mathcal{G}_i$ all elements from C_i can be derived by the intruder and all elements from C_i' cannot.

The decision problem Strategy asks, given a protocol P and a strategy property $((C_1, C_1'), \ldots, (C_s, C_s'))$, whether there exists a state q that satisfies the property.

Theorem 1. Strategy *is decidable.*

To prove this theorem, we show that given a possibly large state q and large q-strategy graphs satisfying the properties, we can reduce the sizes of the state and the strategy graphs, i.e., the sizes of the substitutions in the state and the graphs. For this purpose, we need to deal with all substitutions in all of the strategy graphs at the same time. The challenge is then to guarantee that the reduced strategy graphs are in fact strategy graphs, i.e., satisfy the required conditions. We make use of the fact that the intruder can generate new constants.

5 Balance

In this section, we formalize a fundamental property of contract-signing protocols, namely balance, and explain why this property is decidable in our framework.

As in [6] and most other works on the formal analysis of contract-signing protocols we formulate balance for two-party optimistic contract-signing protocols (see, however, [7]). Besides the two parties (the contractual partners), say A and B, who want to sign the contract, a trusted third party T is involved in the protocol, and is consulted in case a problem occurs. We assume in what follows that the protocols are modeled in such a way that if A has a valid contract, it writes the atom AHasValidContract into the network, similarly for B.

We formulate balance under the assumption that one of the contractual parties is dishonest and the other party is honest, i.e., follows the protocol. The actions of the dishonest party are performed by the intruder, and hence, are arbitrary. All private information the dishonest party has, such as private keys and addresses for the secure channel, are given to the intruder as part of his initial knowledge. The trusted third party is assumed to be honest. We denote the honest party by H and the dishonest party by I.

Informally speaking, balance for H means that at no stage of the protocol run I has both a strategy to prevent H from obtaining a valid contract with I on the previously agreed contractual text, say text, and a strategy to obtain a valid contract with H on text.

In order to formulate that I has a strategy to obtain a valid contract, we assume that the protocol description contains a specific principal, a *watch dog*, which when receiving a valid contract for I outputs $\mathsf{IHasValidContract}$. Such a watch dog can easily be derived from a protocol specification. For the ASW protocol, for example, the watch dog would be a principal with two edges leaving the root of the tree. The first edge would be labeled with the rule

$$\langle \mathsf{sig}[m_O, k_H], N_H, \mathsf{sig}[m_I, k_I], x \rangle \Rightarrow \mathsf{IHasValidContract}$$

where $m_H = \langle k_H, k_I, k_T, \mathsf{text}, \mathsf{hash}(N_H) \rangle$ and $m_I = \langle m_H, \mathsf{hash}(x) \rangle$. The nonce N_H could be replaced by a variable y in case no specific instance of H is considered, and hence, no specific nonce generated in such an instance. This rule indicates that I has a standard contract with H. Analogously, the rule for the second edge would check whether I has a replacement contract with H.

Now, formally a protocol P is *not* balanced for H if P satisfies the strategy property $((\{\mathsf{IHasValidContract}\}, \emptyset), (\emptyset, \{\mathsf{HHasValidContract}\}))$. By Theorem 1, this can be decided. In [11], following [6], we also consider a formulation of balance involving abort tokens.

6 Conclusion

In this paper we have shown that effectiveness, fairness, and balance, a branching property of contract-signing protocols, is decidable when there is no bound on the message size for a Dolev-Yao intruder and when there are only a finite number of sessions. This extends known results on the decidability of reachability problems for cryptographic protocols in a natural way. Our approach is fairly generic; it should therefore be a good starting point for analyzing other game-theoretic properties of cryptographic protocols. From a practical point of view, our result may also be a good starting point for developing more precise analyzers for contract-signing protocols.

Acknowledgment. We thank the anonymous referees for their comments, which helped, in particular, to improve the presentation in Section 5.

References

1. R.M. Amadio, D. Lugiez, and V. Vanackere. On the symbolic reduction of processes with cryptographic functions. *TCS*, 290(1):695–740, 2002.
2. A. Armando, D. Basin, M. Bouallagui, Y. Chevalier, L. Compagna, S. Mödersheim, M. Rusinowitch, M. Turuani, L. Viganò, and L. Vigneron. The AVISS Security Protocol Analysis Tool. In CAV 2002, volume 2404 of *LNCS*, pages 349–353. Springer, 2002.

3. N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *Proc. of the IEEE Symposium on Research in Security and Privacy*, pages 86–99, 1998.
4. M. Ben-Or, O. Goldreich, S. Micali, and R.L. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1):40–46, 1990.
5. M. Boreale. Symbolic trace analysis of cryptographic protocols. In *ICALP 2001*, volume 2076 of *Lecture Notes in Computer Science*, pages 667–681. Springer-Verlag, 2001.
6. R. Chadha, M.I. Kanovich, and A.Scedrov. Inductive methods and contract-signing protocols. In *CCS 2001*, pages 176–185. ACM Press, 2001.
7. R. Chadha, S. Kremer, and A. Scedrov. Formal analysis of multi-party contract signing. In *CSFW-17*, pages 266–279. IEEE Computer Society Press, 2004.
8. H. Comon and V. Shmatikov. Is it possible to decide whether a cryptographic protocol is secure or not? *Journal of Telecommunications and Information Technolog*, 2002.
9. P. H. Drielsma and S. Mödersheim. The ASW protocol revisited: A unified view. In *Workshop on Automated Reasoning for Security Protocol Analysis (ARSPA)*, 2004.
10. J.A. Garay, M. Jakobsson, and P. MacKenzie. Abuse-free optimistic contract signing. In *CRYPTO'99*, volume 1666 of *LNCS*, pages 449–466. Springer-Verlag, 1999.
11. D. Kähler, R. Küster, T. Wilke. Deciding Properties of Contract-Signing Protocols. Technical Report IFI 0409, CAU Kiel, 2004. Available from http://www.informatik.uni-kiel.de/reports/2004/0409.html
12. S. Kremer and J.-F. Raskin. Game analysis of abuse-free contract signing. In *Computer Security Foundations Workshop 2002 (CSFW 2002)*. IEEE Computer Society, 2002.
13. C. Meadows. Formal Methods for Cryptographic Protocol Analysis: Emerging Issues and Trends. *IEEE Journal on Selected Areas in Communication*, 21(1):44–54, January 2003.
14. J. K. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *CCS 2001*, pages 166–175. ACM Press, 2001.
15. M. Rusinowitch and M. Turuani. Protocol insecurity with a finite number of sessions, composed keys is NP-complete. *Theoretical Computer Science*, 299(1–3):451–475, 2003.
16. V. Shmatikov and J.C. Mitchell. Finite-state analysis of two contract signing protocols. *Theoretical Computer Science*, 283(2):419–450, 2002.
17. J. Zhou and D. Gollmann. A fair non-repudiation protocol. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 55–61. IEEE Computer Society Press, 1996.