A Dolev-Yao-based Definition of Abuse-free Protocols

Detlef Kähler, Ralf Küsters, and Thomas Wilke

Christian-Albrechts-Universität zu Kiel
{kaehler,kuesters,wilke}@ti.informatik.uni-kiel.de

Abstract. We propose a Dolev-Yao-based definition of abuse freeness for optimistic contract-signing protocols which, unlike other definitions, incorporates a rigorous notion of what it means for an outside party to be convinced by a dishonest party that it has the ability to determine the outcome of the protocol with an honest party, i.e., to determine whether it will obtain a valid contract itself or whether it will prevent the honest party from obtaining a valid contract. Our definition involves a new notion of test (inspired by static equivalence) which the outside party can perform. We show that an optimistic contract-signing protocol proposed by Asokan, Shoup, and Waidner is abusive and that a protocol by Garay, Jakobsson, and MacKenzie is abuse-free according to our definition. Our analysis is based on a synchronous concurrent model in which parties can receive several messages at the same time. This results in new vulnerabilities of the protocols depending on how a trusted third party reacts in case it receives abort and resolve requests at the same time.

1 Introduction

Abuse freeness is a security property introduced in [9] for optimistic contract-signing protocols: An optimistic (two-party) contract-signing protocol is a protocol run by A (Alice), B (Bob), and a trusted third party T (TTP) to exchange signatures on a previously agreed upon contractual text with the additional property that the TTP will only be involved in a run in case of problems. Such a protocol is *not* abuse-free for (honest) Alice if at some point during a protocol run (dishonest) Bob can "convince" an outside party, Charlie, that he is in an unbalanced state, where, following the terminology of [4], *unbalanced* means that Bob has both (i) a strategy to prevent Alice from getting a valid contract and (ii) a strategy to obtain a valid contract. In other words, Alice can be misused by Bob to get leverage for another contract (with Charlie). Obviously, abuse-free contract-signing protocols are highly desirable.

The main goal of the present work is to present a formal definition of abuse freeness which is as protocol-independent as possible. The crucial issue with such a formal definition is that it needs to specify what it means for Bob to *convince* Charlie. One of the first proposals for this was presented by Kremer and Raskin [12]. Roughly, their proposal is the following: To convince Charlie a message is presented to Charlie from which he can deduce that "a protocol run has been started between Alice and Bob". What that means is, however, not specified in a general fashion in [12]. Instead, this is decided on a case by case basis. The objective of this paper is to give a generic definition. The only part which needs to be decided on a case by case basis in our definition is what it means for Alice (or Bob) to have received a valid contract—something which can hardly be described in a generic way—and what the assumptions are that Charlie makes.

Before we explain our approach and the contribution of our work we need to explain the following crucial point: Whether or not Charlie is convinced should be based on evidence provided by Bob. Following [9], we model this evidence as a message that Bob presents to Charlie. (In [9], this is called an off-line attack.) This, however, has an important implication. Since Bob can hold back any message he wants to (he can himself decide which messages he shows to Charlie) and since Charlie is assumed to be an outside party not involved in the protocol, if Bob could convince Charlie to be in some state of the protocol at some point, at any later point he would be able to convince Charlie that he was in the same state, simply by providing the same evidence. Therefore, Bob can only convince Charlie that he is or was and still might be in an unbalanced state. We employ this notion of abuse freeness for our work. (Note that it is stronger than the one described above as Charlie is more easily convinced.)

Contribution of this Work. We provide a formal definition of the version of abuse freeness just explained, apply our definition to the optimistic contract-signing protocols by Asokan, Shoup, and Waidner [3] (ASW protocol) and by Garay, Jakobsson, and MacKenzie [9] (GJM protocol), and show that the ASW protocol is abusive while the GJM protocol is abuse-free according to our definition.

The idea behind our definition of abuse freeness is that Bob presents a message to Charlie and Charlie performs a certain test on this message. If the message passes the test, then Charlie is convinced that Bob is or was and still might be in an unbalanced state. The test is such that from the point of view of Charlie, Bob can only generate messages passing the test in states where Bob is or was in an unbalanced state and where at least one of these states is in fact unbalanced. To describe the power Bob has, we adopt a Dolev-Yao style approach [8] (see also [2, 1, 7]). Our definition of test is inspired by the notion of static equivalence [2].

We use a synchronous concurrent communication model in which principals and the (Dolev-Yao-style) intruder may send several messages to different parties at the same time. This rather realistic model requires to specify the behavior of protocol participants in case several messages are received at the same time (or within one time slot). This leads to new effects that have not been observed in previous works. In the ASW and GJM protocols, one needs to specify the behavior of the TTP in case an abort and a resolve request are received at the same time (from different parties). The question arises whether the TTP should answer with an abort or a resolve acknowledgment. We show that if the TTP does the former, then the ASW and the GJM protocol are unbalanced for the responder, and if it does the latter, the two protocols are unbalanced for the initiator.

Related Work. As mentioned above, Kremer et al. [12] analyzed the ASW and GJM protocol based on finite-state alternating transition systems, using an automatic analysis tool. They explicitly needed to specify the behavior of dishonest principals and which states are the ones that are convincing to Charlie (they use a propositional variable $\mathsf{prove2C}$, which they set manually). This is what our definition makes obsolete.

Chadha et al. [4] introduce a stronger notion than abuse freeness, namely *balance*: For a protocol to be *unbalanced* one does not require Bob to convince Charlie that he is in an unbalanced state. The fact that an unbalanced state exists is sufficient for a protocol to be unbalanced. Hence, balance is a formally stronger notion than abuse freeness. Unfortunately, this notion is too strong in some cases. In fact, as shown by Chadha et al. [6] in an interleaving (rather than real concurrent) model, if principals are optimistic, i.e., they are willing to wait for messages of other parties, balance is impossible to achieve; in this paper, Chadha et al. also sketch a definition of abuse freeness based on epistemic logic, but without going into details. In [5], Chadha et al. study multi-party contract signing protocols.

Shmatikov and Mitchell [13] employ the finite-state model checker Murφ to automatically analyze contract-signing protocols. They, too, approximate the notion of abuse freeness by a notion similar to balance.

Structure of the Paper. The technical part of the paper starts with an informal description of the ASW protocol in Sect. 2, which then serves as a running example for the further definitions. In Sect. 3, we describe our communication and protocol model, with the new definition of abuse freeness presented in Sect. 4. We then treat the ASW and the GJM protocol in our framework in Sect. 5 and Sect. 6. We conclude in Sect. 7. A full version of our paper is available, see [11].

2 The ASW Protocol

In this section, we recall the Asokan-Shoup-Waidner (ASW) protocol from [3], which will serve as a running example; the Garay-Jakobsson-MacKenzie (GJM) protocol from [9] will be explained in Section 6.

The ASW protocol assumes the following scenario: Alice and Bob want to sign a contract and a TTP is present. Further, it is agreed upon that the following two types of messages, the *standard contract (SC)* and the *replacement contract (RC)*, will be recognized as valid contracts between Alice and Bob with contractual text text: $SC = \langle me_1, N_A, me_2, N_B\rangle$ and $RC = \mathsf{sig}_T(\langle me_1, me_2\rangle)$ where $me_1 = \mathsf{sig}_A(\langle A, B, \mathsf{text}, \mathsf{hash}(N_A)\rangle)$ and $me_2 = \mathsf{sig}_B(\langle me_1, \mathsf{hash}(N_B)\rangle)$, and as usual, N_A and N_B stand for nonces. In addition to SC and RC, the variants of SC and RC which one obtains by exchanging the roles of A and B are regarded as valid contracts.

There are three interdependent parts to the protocol: an exchange protocol, an abort protocol, and a resolve protocol. The *exchange protocol* consists of four steps, which, in Alice-Bob notation, are displayed in Fig. 1. The first two messages, me_1 and me_2, serve as respective *promises* of Alice and Bob to sign the contract, and N_A and N_B serve as *contract authenticators*: After they have been revealed, Alice and Bob can compose the standard contract, SC.

$$A \to B : me_1$$
$$B \to A : me_2$$
$$A \to B : N_A$$
$$B \to A : N_B$$

Fig. 1: ASW exchange protocol

The *abort protocol* is run between Alice and the TTP and is used by Alice to abort the contract signing process when she does not receive Bob's promise. Alice will obtain (from the TTP) an abort receipt or, if the protocol instance has already been resolved (see below), a replacement contract. The first step is $A \to T\colon ma_1$ where $ma_1 = \mathsf{sig}_A(\langle \mathsf{aborted}, me_1\rangle)$ is Alice's *abort request*; the second step is the TTP's reply, which is either $\mathsf{sig}_T(\langle \mathsf{aborted}, ma_1\rangle)$, the *abort receipt*, if the protocol has not been resolved, or the replacement contract, RC.

Similarly, the *resolve protocol* can be used by Alice and Bob to resolve the protocol, which either results in a replacement contract or, if the protocol has already been aborted, in an abort receipt. When Bob runs the protocol (because Alice has not sent her contract authenticator yet), the first step is $B \to T\colon \langle me_1, me_2\rangle$; the second step is the TTP's reply, which is either the abort receipt $\mathsf{sig}_T(\langle \mathsf{aborted}, ma_1\rangle)$, if the protocol has already been aborted, or the replacement contract, RC. The same protocol (with roles of A and B exchanged) is also used by Alice.

It is assumed that the communication between Alice and the TTP and between Bob and the TTP goes through a channel that is not under the control of the intruder (the dishonest party), i.e., the intruder cannot delay, modify, or insert messages. We refer to such a channel as secure. Whether or not the intruder can read messages sent on this channel does not effect the results shown in this paper.

3 The Concurrent Protocol and Intruder Model

In this section, we introduce our protocol and intruder model, which, unlike most other Dolev-Yao-based models, captures real concurrent computation. Given sets S, T, U with $U \subseteq T$, we denote by S^T the set of functions from T to S and for $f \in S^T$ we denote by $f|_U$ the restriction of f to U.

3.1 Concurrent System Model

A concurrent system in our framework is made up of several components, which are automata provided with input and output ports for inter-component communication. Each such port can either carry a message from a given set $\mathcal{M}$ of messages or the special symbol '$\circ$' (no message). We use $\mathcal{M}_\circ$ to denote $\mathcal{M} \cup \{\circ\}$. A run of such a system proceeds in rounds: In every round, every component reads the input on all of its input ports, and then, depending on its current state, writes output on its output ports (possibly $\circ$), and goes into a new state.

A message written on an output port is read in the next round by the component with the corresponding input port. Note that all components perform their "receive-send action" at the same time and that a component may receive and send several messages at the same time.

Formally, a *component* of a concurrent system over a set $\mathcal{M}$ of messages is a tuple $\mathcal{A} = (S, \mathbf{In}, \mathbf{Out}, I, \Delta)$ where S is a (possibly infinite) set of *local states*, $\mathbf{In}$ is the set of *input ports*, $\mathbf{Out}$ is the set of *output ports*, disjoint from $\mathbf{In}$, $I \subseteq S$ is the set of *initial states*, and $\Delta \subseteq \mathcal{M}_\circ^{\mathbf{In}} \times S \times S \times \mathcal{M}_\circ^{\mathbf{Out}}$ is the *transition relation*, which, w.l.o.g., is required to be complete: for each $(m, s) \in \mathcal{M}_\circ^{\mathbf{In}} \times S$ there exist s' and m' with $(m, s, s', m') \in \Delta$. A transition (m, s, s', m') is meant to model that if $\mathcal{A}$ is in state s and reads the messages m on its input ports, then it writes m' on its output ports and goes into state s'.

A *concurrent system* over a set $\mathcal{M}$ of messages is a finite family $\{\mathcal{A}_i\}_{i\in P}$ of components over $\mathcal{M}$ of the form $(S_i, \mathbf{In}_i, \mathbf{Out}_i, I_i, \Delta_i)$ such that $\mathbf{In}_i \cap \mathbf{In}_j = \mathbf{Out}_i \cap \mathbf{Out}_j = \emptyset$ for every i and $j \neq i$. Note that an output port of one component may coincide with the input port of another component, which allows the former component to send messages to the latter component.

Given a concurrent system $\mathcal{G} = \{\mathcal{A}_i\}_{i\in P}$ as above, its set of input and output ports is determined by $\mathbf{In} = \bigcup_{i\in P} \mathbf{In}_i$ and $\mathbf{Out} = \bigcup_{i\in P} \mathbf{Out}_i$, respectively, while its state set and its initial state set are defined by $S = \prod_{i\in P} S_i$ and $I = \{s \in S \mid s(i) \in I_i \text{ for } i \in P\}$ where $s(i)$ denotes the entry with index i in s. We set $\mathbf{P} = \mathbf{In} \cup \mathbf{Out}$.

A *concurrent transition* is a tuple of the form (m, s, s', m') satisfying $(m|_{\mathbf{In}_i}, s(i), s'(i), m'|_{\mathbf{Out}_i}) \in \Delta_i$ for every $i \in P$. Note that if $p \in \mathbf{Out}_l \cap \mathbf{In}_r$ for $l \neq r$, then this means that by applying the transition, component $\mathcal{A}_l$ sends message $m'(p)$ to component $\mathcal{A}_r$. A *global state* of $\mathcal{G}$ is a pair (m, s) with $m \in \mathcal{M}_\circ^{\mathbf{P}}$ and $s \in S$, i.e., it contains all current messages on ports and all local states.

An (m, s)-*computation* of $\mathcal{G}$ is an infinite sequence $\rho = m_0 s_0 m_1 s_1 \ldots$ of global states such that $(m_i, s_i, s_{i+1}, m_{i+1})$ is a concurrent transition for every i and $(m_0, s_0) = (m, s)$. *Finite* (m, s)-*computations* are defined in the same way. An infinite (m, s)-computation is called a *run* of $\mathcal{G}$ if $m(p) = \circ$ for every $p \in \mathbf{P}$ and $s \in I$. A finite prefix of a run is called a *run segment*.

A global state (m, s) is called *reachable* if there is a run segment $\rho = m_0 s_0 m_1 s_1 \ldots m_{k-1} s_{k-1}$ such that $(m_{k-1}, s_{k-1}) = (m, s)$. Let (m, s) and (m', s') be global states. We call (m', s') a *descendant of* (m, s) if there is an (m, s)-computation $\rho = m_0 s_0 m_1 s_1 \ldots$ such that $(m', s') = (m_i, s_i)$ for some $i \geq 0$, in particular, (m, s) is a descendant of (m, s).

3.2 Dolev-Yao Systems

To model protocols and the execution of protocols in presence of an intruder, we consider specific concurrent systems, called Dolev-Yao systems. We first introduce messages and terms, along the lines of [1, 2, 7].

Given a signature Σ and a set of variables $\mathcal{V}$, the set of *terms* $\mathcal{T}(\Sigma, \mathcal{V})$ and the set of *ground terms* $\mathcal{T}(\Sigma)$ are defined as usual. Given a set $\mathcal{S} \subseteq \mathcal{T}(\Sigma, \mathcal{V})$, called a set of *basic operations*, we call a term t an $\mathcal{S}$-*term* if $t \in \mathcal{S}$ or if it is obtained from a term in $\mathcal{S}$ by substituting $\mathcal{S}$-terms for variables and renaming variables. We also consider an equational theory $\mathcal{H}$ over Σ, which we assume is convergent, implying that every term t has a unique normal form, which we denote by $t{\downarrow}$.

For example, to model the ASW protocol we consider the signature Σ_{ASW} consisting of the following symbols: $\mathsf{sig}(\cdot,\cdot)$, $\mathsf{sigcheck}(\cdot,\cdot,\cdot)$, $\mathsf{pk}(\cdot)$, $\mathsf{sk}(\cdot)$, $\langle\cdot,\cdot\rangle$, $\pi_1(\cdot)$, $\pi_2(\cdot)$, $\mathsf{hash}(\cdot)$, A, B, T, text, ok, $\mathsf{initiator}$, $\mathsf{responder}$, $\mathsf{aborted}$, and an infinite number of constants. Further, we choose operations that model pairing, projections, checking a signature, signing, and hashing, that is, $\mathcal{S}_{ASW}$ consists of the following basic operations: $\langle x_1, x_2\rangle$, $\pi_1(x_1)$, $\pi_2(x_1)$, $\mathsf{sigcheck}(x_1, x_2, x_3)$, $\mathsf{sig}(x_1, x_2)$, and $\mathsf{hash}(x_1)$. The semantics of these operations is determined by the equational theory $\mathcal{H}_{ASW}$, which consists of the following three identities: $\pi_1(\langle x, y\rangle) = x$, $\pi_2(\langle x, y\rangle) = y$, and $\mathsf{sigcheck}(x, \mathsf{sig}(\mathsf{sk}(y), x), \mathsf{pk}(y)) = \mathsf{ok}$.

Using the set $\mathcal{S}$ of basic operations, an intruder can derive messages from a given set $\mathcal{K}$ of messages by forming $(\mathcal{S} \cup \mathcal{K})$-terms. We define $d_{\mathcal{S}}(\mathcal{K}) = \{m{\downarrow} \mid m \text{ is an } (\mathcal{S} \cup \mathcal{K})\text{-term without variables}\}$ to be the set of messages (in normal form) that can be derived from $\mathcal{K}$ using $\mathcal{S}$. In the ASW example, with $\mathcal{K} = \{\langle\mathsf{contract}, \mathsf{sig}(\mathsf{sk}(A), \mathsf{contract})\rangle, \mathsf{sk}(B)\}$, the following term is an $(\mathcal{S}_{ASW} \cup \mathcal{K})$-term: $m = \mathsf{sig}(\mathsf{sk}(B), \pi_1(\langle\mathsf{contract}, \mathsf{sig}(\mathsf{sk}(A), \mathsf{contract})\rangle))$. The normal form $m{\downarrow} = \mathsf{sig}(\mathsf{sk}(B), \mathsf{contract})$ of m belongs to $d_{\mathcal{S}_{ASW}}(\mathcal{K})$.

To specify a Dolev-Yao system, we partition a given (finite) set ALL of all principals into a set HON of *honest principals* and a set $\mathit{DIS} = \mathit{ALL} \setminus \mathit{HON}$ of *dishonest principals*. In the Dolev-Yao system, we have a component $\mathcal{A}_\pi$ for every honest principal (*honest components*) and one component $\mathcal{A}_\mathcal{I}$, the *intruder component*, subsuming all dishonest principals. Each honest component $\mathcal{A}_\pi$ has ports (i) $\mathrm{netin}^\pi_{\pi'}$ and $\mathrm{sec}^\pi_{\pi'}$ for sending messages to π' for every π' through the network and the secure channel, respectively, and (ii) ports $\mathrm{netout}^{\pi'}_\pi$ and $\mathrm{sec}^{\pi'}_\pi$ for receiving messages coming from the network (supposedly from π') and from the secure channel (definitely from π') for every π'. The input and output port sets of the intruder component $\mathcal{A}_\mathcal{I}$ are $\mathbf{In}_\mathcal{I} = \{\mathrm{netin}^\pi_{\pi'} \mid \pi \in \mathit{HON}, \pi' \in \mathit{ALL}\} \cup \{\mathrm{sec}^\pi_{\pi'} \mid \pi \in \mathit{HON}, \pi' \in \mathit{DIS}\}$ and $\mathbf{Out}_\mathcal{I} = \{\mathrm{netout}^\pi_{\pi'} \mid \pi' \in \mathit{HON}, \pi \in \mathit{ALL}\} \cup \{\mathrm{sec}^\pi_{\pi'} \mid \pi \in \mathit{DIS}, \pi' \in \mathit{HON}\}$, respectively. Note that one end of a network port is always connected to the intruder (since he controls the network), while secure channel ports directly connect two honest principals or an honest principal and a dishonest principal (i.e., the intruder). Instead of connecting two honest principals directly through a secure channel, one could plug between two honest principals a secure channel component for more flexible scheduling. However, for simplicity and since this does not change our results (if secure channel components between honest principals are not controlled by the adversary), we choose direct secure channel links.

The intruder component acts as a Dolev-Yao intruder in that it may derive arbitrary messages from its initial knowledge and the messages received so far using $\mathcal{S}$-terms as described above. Note, however, that the intruder component (as all other components) may receive and send several messages at the same time.

Given the set *HON* of honest and the set *DIS* of dishonest principals, a family $\{\mathcal{A}_\pi\}_{\pi\in HON}$ of honest components (with ports as specified above), and a set $\mathcal{K}$ of messages (the initial intruder knowledge), we denote by $\mathrm{DY}[\{\mathcal{A}_\pi\}_{\pi\in HON}, HON, DIS, \mathcal{K}]$ the induced Dolev-Yao system where the set $\mathcal{S}$ of operations the intruder may use to derive new messages is understood from the context. If (m, s) is a global state of a run of such a system, we denote by $\mathcal{K}(m, s)$ the initial knowledge of the intruder plus the messages he has seen so far on his input ports (including the messages currently on his input ports). We say that the intruder can *deduce message* m' *at state* (m, s) if $m' \in d_{\mathcal{S}}(\mathcal{K}(m, s))$.

4 Balanced and Abuse-free Protocols

In this section, we present our formal definition of abuse freeness, based on the notion of balance, which, in turn, is based on the notion of strategy.

4.1 Balanced Protocols

Throughout this subsection, we assume a concurrent system $\mathcal{G} = \{\mathcal{A}_i\}_{i\in P}$ with set of ports $\mathbf{P}$ and state set S to be given.

Strategies in the context of abuse freeness need to be defined with respect to partial information, since Bob will not necessarily know the global state of the entire protocol at any point of the protocol execution. In addition, strategies can be carried out jointly by several components. This motivates the following definitions.

A function with domain $(\mathcal{M}_\circ^{\mathbf{P}} \times S)^+$ is called a *view function* for $\mathcal{G}$. Given a view function view and a run segment ρ, we say that $\mathsf{view}(\rho)$ is the *view of* ρ *w. r. t.* view. Any subset of P is called a *coalition*. Given a coalition J, we write $\mathbf{Out}_J$ for $\bigcup_{j\in J} \mathbf{Out}_j$.

Given both, a coalition J and a view function $\mathsf{view}\colon (\mathcal{M}_\circ^{\mathbf{P}} \times S)^+ \to W$, a *view-strategy for* J is a function σ which determines how the components of J act depending on their current view $w \in W$, which itself is determined by view. More precisely, σ assigns to each $w \in W$ successor states $s_j \in S_j$ (for $j \in J$) and messages m_p (for $p \in \mathbf{Out}_J$) to be written to the output ports of the components of the coalition. Clearly, these choices are required to be consistent with the individual transition relations Δ_j. Given a strategy σ and a global state (m, s), we denote by $\mathsf{out}((m, s), \sigma)$ the set of all infinite (m, s)-computations in which the components of the coalition J follow the strategy σ.

In our formal definition of balance, path properties are used to define what exactly it means to prevent Alice from getting a valid contract or to obtain one. Formally, a set $\varphi \subseteq (\mathcal{M}_\circ^{\mathbf{P}} \times S)^\omega$ is called a $\mathcal{G}$*-property*.

The notion of balance will be defined w.r.t. what we call a *balance specifier*, i.e., a tuple β of the form $(I, \mathsf{view}, \varphi_1, \varphi_2)$ where I is a coalition, view is a view function, and φ_1 and φ_2 are path properties. For instance, assume we want to describe balance for Alice in a concrete contract signing setting. Then we need to check whether there exist certain strategies for Bob, so we may choose $I = \{B\}$.

More precisely, we want to know whether Bob has a strategy for preventing that Alice gets a valid contract and a strategy for making sure Bob gets a valid contract. So we define φ_1 as the set of all runs of the protocol where Alice does not get a valid contract and φ_2 as the set of all runs where Bob gets a valid contract. Finally, we choose view in such a way that at any given point in a run, view returns everything Bob has observed of the system thus far. Similarly to [4], balance is now defined as follows:

Definition 1 (balance). *Let $\mathcal{G}$ be a concurrent system with index set P, (m, s) a reachable state of $\mathcal{G}$, and $\beta = (I, \mathsf{view}, \varphi_1, \varphi_2)$ a balance specifier.*

The state (m, s) is β-unbalanced *if there are view-strategies σ_1 and σ_2 for I such that $\rho \in \varphi_i$ for every $\rho \in \mathsf{out}((m, s), \sigma_i)$ and $i \in \{1, 2\}$. The system $\mathcal{G}$ is* β-unbalanced *if there is a reachable state (m, s) of $\mathcal{G}$ that is β-unbalanced.*

4.2 Abuse-free Protocols

As already explained earlier, when a protocol is considered abuse-free, then this means that from Charlie's point of view Bob has no way of convincing him that he is in an unbalanced state. That is, the property of being abuse-free is relative to the view that Charlie has of the protocol. Technically, such a view is determined by a Dolev-Yao system and a balance specifier. This motivates the following definition. A pair $(\mathcal{G}^e, \beta^e)$ consisting of a Dolev-Yao system $\mathcal{G}^e$ and a balance specifier β^e is called an *external view (with respect to abuse freeness).*

We use a specific but natural notion of test that Charlie can make use of to verify that Bob is in fact in the position he claims to be in. As a parameter it uses a set $\mathcal{X} \subseteq \mathcal{M}$ of messages, which should be thought of as Charlie's a-priori knowledge, such as his private key.

A pair (M, M') of $(\mathcal{S} \cup \mathcal{X})$-terms (containing exactly one variable x) is called an *atomic $\mathcal{X}$-test.* A message $m \in \mathcal{M}$ *passes the test* (M, M'), denoted $m \models (M, M')$, if $M[m/x] \equiv_{\mathcal{H}} M'[m/x]$. The message m *fails the test* (M, M') if m does not pass it. This is extended to *boolean* and *ω-tests* in a straightforward fashion, where in ω-tests conjunctions and disjunctions with a denumerable number of arguments are allowed (our results hold for both boolean and ω-tests). For instance, if Charlie wants to check whether a message has the form $\langle c, \mathsf{sig}(\mathsf{sk}(A), c)\rangle$, then he can use the boolean test $(\pi_1(x), c) \wedge (\mathsf{sigcheck}(c, \pi_2(x), \mathsf{pk}(A)), \mathsf{ok})$.

As explained above, in our definition Charlie uses a test to distinguish between messages that give evidence for an unbalanced state and messages which don't. In other words, Charlie considers a state unbalanced when Bob could possibly deduce a message in that state which passes the test. Therefore, we say that for a given $\mathcal{X}$-test θ, a state (m, s) of a Dolev-Yao system $\mathcal{G}^e$ is *θ-possible* if there exists $m' \in d_{\mathcal{S}}(\mathcal{K}(m, s))$ such that $m' \models \theta$.

The next definition puts everything together. A protocol is not abuse-free if there exists a convincing test which indicates unbalanced states as explained in the introduction:

Definition 2 (abuse freeness). *Let $\mathcal{X} \subseteq \mathcal{M}$. An external view $(\mathcal{G}^e, \beta^e)$ is $\mathcal{X}$-* abusive *if there exists an $\mathcal{X}$-test θ such that the following two conditions are satisfied:*

1. *There exists a θ-possible and β-unbalanced state in $\mathcal{G}^e$.*
2. *Each θ-possible state (m, s) of $\mathcal{G}^e$ is a descendant of a θ-possible and β-unbalanced state in $\mathcal{G}^e$.*

*Such a test is called $(\mathcal{G}^e, \beta^e)$-*convincing*. The external view $(\mathcal{G}^e, \beta^e)$ is called* $\mathcal{X}$-abuse-free *if $(\mathcal{G}^e, \beta^e)$ is not $\mathcal{X}$-abusive.*

5 The ASW Protocol Analyzed

In this section, we present our results concerning the analysis of the ASW contract signing protocol. For our formal analysis of the ASW protocol, we let $\sigma = \sigma_{ASW}$, $\mathcal{S} = \mathcal{S}_{ASW}$, and $\mathcal{H} = \mathcal{H}_{ASW}$, as explained in Sect. 3.

5.1 The ASW Protocol is not Balanced

First, we note that the ASW protocol (without an optimistic honest party) can be shown to be balanced in an interleaving (as opposed to a real concurrent) model; the proof is along the same lines as the one presented in [4] for the GJM protocol. By contrast, if we consider a concurrent setting and make the assumptions that Bob (the intruder) is (1) as fast as Alice in sending messages and (2) the TTP handles a resolve request first when an abort request is received at the same time (or in the same time slot), we can argue (informally) that the protocol is unbalanced: Bob has (i) a strategy to prevent Alice from getting a valid contract, namely by simply doing nothing, and (ii) a strategy to resolve the contract signing process after Alice has sent the first message of the exchange protocol, namely by sending a resolve request to the TTP. Even if Alice sends an abort request to the TTP at the same time, because of assumption (1) her request cannot reach the TTP before Bob's resolve request, and with assumption (2), we know that Bob's resolve request takes priority over Alice's abort request.

Assumption (2) from above shows that we need to be careful when implementing the TTP, because of simultaneous requests. If the TTP receives a resolve request and an abort request at the same time, it could first serve the resolve request and then the abort request or vice versa. As a consequence, we distinguish two models of the TTP, denoted T and T', with corresponding components $\mathcal{A}_T$ and $\mathcal{A}_{T'}$, respectively, and we show that for both variants of the TTP, the ASW protocol is unbalanced.

We consider two scenarios. In the first one, we have honest Alice, dishonest Bob, and T, and in the second one, we have dishonest Alice, honest Bob, and T'. More precisely, we consider $\mathcal{G}_{ASW} = \mathrm{DY}[\{\mathcal{A}_i\}_{i \in \{A,T\}}, \{A, T\}, \{B\}, \mathcal{K}]$ and $\mathcal{G}'_{ASW} = \mathrm{DY}[\{\mathcal{A}_i\}_{i \in \{B,T'\}}, \{B, T'\}, \{A\}, \mathcal{K}']$ where $\mathcal{K} = \mathcal{K}_0 \cup \{\mathsf{sk}(B)\}$, $\mathcal{K}' = \mathcal{K}_0 \cup \{\mathsf{sk}(A)\}$, and $\mathcal{K}_0 = \{A, B, T, \mathsf{text}, \mathsf{pk}(A), \mathsf{pk}(B), \mathsf{pk}(T), \mathsf{initiator}, \mathsf{responder}, \mathsf{ok}\}$, which means that among other things the intruder's initial knowledge comprises

Bob's and Alice's private key, respectively. The components $\mathcal{A}_A$ and $\mathcal{A}_B$ are easily obtained from the informal description in Sect. 2.

We define, in a straightforward fashion, path properties $\bar{\varphi}_A$, $\bar{\varphi}_B$, and $\varphi_{\mathcal{I}}$ to describe that Alice does not get a valid contract, that Bob does not get a valid contract, and that the intruder does get a valid contract, respectively.

We assume that the intruder's view of the system is limited to his own history, that is, we use an appropriate view function $\mathsf{view}_{\mathcal{I}}$, which removes anything the intruder cannot observe.

Finally, we define the balance specifiers $\beta_{ASW} = (\{\mathcal{I}\}, \mathsf{view}_{\mathcal{I}}, \bar{\varphi}_A, \varphi_{\mathcal{I}})$ and $\beta'_{ASW} = (\{\mathcal{I}\}, \mathsf{view}_{\mathcal{I}}, \bar{\varphi}_B, \varphi_{\mathcal{I}})$, which are designed in such a way that they describe being unbalanced for Bob and for Alice, respectively.

Following the informal reasoning from above, we prove that the ASW protocol is unbalanced for either Alice or Bob, depending on which version of the TTP is used:

Theorem 1 (ASW is unbalanced). *The Dolev-Yao system $\mathcal{G}_{ASW}$ is β_{ASW}-unbalanced, and, similarly, $\mathcal{G}'_{ASW}$ is β'_{ASW}-unbalanced.*

5.2 The ASW Protocol is not Abuse-free

For abuse freeness, we imagine that Charlie assumes that there is only one instance of the ASW protocol running, but that he does not know whether Alice is the initiator or responder, which is a realistic assumption. Formally, we replace $\mathcal{A}_A$ by a variant of it, denoted $\mathcal{A}_{A'}$, which in the beginning decides whether it wants to play the role of the initiator or the responder and then sends a corresponding message to Bob. We set $\mathcal{G}^e_{ASW} = \mathrm{DY}[\{\mathcal{A}_i\}_{i\in\{A',T\}}, \{A', T\}, \{B\}, \mathcal{K}]$ with $\mathcal{K}$ as above, $\beta^e_{ASW}{=}(\{\mathcal{I}^e\}, \mathsf{view}_{\mathcal{I}}, \bar{\varphi}_A, \varphi_{\mathcal{I}})$, and $\mathcal{X}{=}\{A, B, C, T, \mathsf{pk}(A), \mathsf{pk}(B), \mathsf{pk}(C), \mathsf{pk}(T), \mathsf{sk}(C), \mathsf{text}, \mathsf{ok}\}$. Here, $\mathcal{I}^e$ denotes the intruder of $\mathcal{G}^e$. We prove:

Theorem 2 (ASW not abuse-free). *The external view $(\mathcal{G}^e_{ASW}, \beta^e_{ASW})$ is $\mathcal{X}$-abusive. This remains true when T is replaced by T'.*

In the proof we identify a test for checking whether a message is Alice's promise of signature in an instance initiated by her and show that this test is convincing.

6 The GJM Protocol Analyzed

We show that, in a concurrent setting, the GJM protocol is unbalanced but abuse-free. The structure of the GJM protocol is exactly as for the ASW protocol. However, the actual messages exchanged are different. In particular, in the version of the exchange protocol of the GJM protocol the first two messages are so-called private contract signatures [9] and the last two messages are actual signatures (obtained by converting the private contract signatures into universally verifiable signatures).

For the GJM protocol we consider the signature Σ_{GJM} which contains the following individual symbols: $\mathsf{sig}(\cdot,\cdot,\cdot)$, $\mathsf{sigcheck}(\cdot,\cdot,\cdot)$, $\mathsf{pk}(\cdot)$, $\mathsf{sk}(\cdot)$, $\langle\cdot,\cdot\rangle$, $\pi_1(\cdot)$, $\pi_2(\cdot)$,

$\mathsf{fake}(\cdot,\cdot,\cdot,\cdot,\cdot)$, $\mathsf{pcs}(\cdot,\cdot,\cdot,\cdot,\cdot)$, $\mathsf{pcsver}(\cdot,\cdot,\cdot,\cdot,\cdot)$, $\mathsf{sconvert}(\cdot,\cdot,\cdot)$, $\mathsf{tpconvert}(\cdot,\cdot,\cdot)$, $\mathsf{sver}(\cdot,\cdot,\cdot,\cdot)$, $\mathsf{tpver}(\cdot,\cdot,\cdot,\cdot)$, A, B, T, text, $\mathsf{initiator}$, $\mathsf{responder}$, ok, pcsok, sok, tpok, and $\mathsf{aborted}$. In addition, it includes infinite sets $\mathcal{C}$ and $\mathcal{R}$ of constants that stand for nonces and random coins used by the parties.

The equational theory $\mathcal{H}_{GJM}$ that we consider contains (among some obvious identities) the following identities to model private contract signatures (PCS):

$$\mathsf{pcsver}(w,\mathsf{pk}(x),\mathsf{pk}(y),\mathsf{pk}(z),\mathsf{pcs}(u,\mathsf{sk}(x),w,\mathsf{pk}(y),\mathsf{pk}(z))) = \mathsf{pcsok}, \quad (1)$$

$$\mathsf{pcsver}(w,\mathsf{pk}(x),\mathsf{pk}(y),\mathsf{pk}(z),\mathsf{fake}(u,\mathsf{sk}(y),w,\mathsf{pk}(x),\mathsf{pk}(z))) = \mathsf{pcsok}, \quad (2)$$

$$\mathsf{sver}(w,\mathsf{pk}(x),\mathsf{pk}(z),\mathsf{sconvert}(u,\mathsf{sk}(x),\mathsf{pcs}(v,\mathsf{sk}(x),w,\mathsf{pk}(y),\mathsf{pk}(z)))) = \mathsf{sok}, \quad (3)$$

$$\mathsf{tpver}(w,\mathsf{pk}(x),\mathsf{pk}(z),\mathsf{tpconvert}(u,\mathsf{sk}(z),\mathsf{pcs}(v,\mathsf{sk}(x),w,\mathsf{pk}(y),\mathsf{pk}(z)))) = \mathsf{tpok}. \quad (4)$$

A term of the form $\mathsf{pcs}(u,\mathsf{sk}(x),w,\mathsf{pk}(y),\mathsf{pk}(z))$ stands for a PCS computed by x (with $\mathsf{sk}(x)$) involving the text w, the party y, and the TTP z while u models the random coins used to compute the PCS. Everybody can verify the PCS with the public keys involved (identity (1)), but cannot determine whether the PCS was computed by x or y (identity (2)): instead of x computing the "real" PCS, y could have computed a "fake" PCS which would also pass the verification with pcsver. Using $\mathsf{sconvert}$ and $\mathsf{tpconvert}$, see (3) and (4), a "real" PCS can be converted by x and the TTP z, respectively, into a universally verifiable signature (verifiable by everyone who possesses $\mathsf{pk}(x)$ and $\mathsf{pk}(z)$).

With the Dolev-Yao systems $\mathcal{G}_{GJM}$, $\mathcal{G}'_{GJM}$ and the balance specifiers β_{GJM} and β'_{GJM} defined as in the case of the ASW protocol but with the messages adapted to the GJM protocol, we obtain:

Theorem 3 (GJM is unbalanced). *The Dolev-Yao systems $\mathcal{G}_{GJM}$ and $\mathcal{G}'_{GJM}$ are β_{GJM}- and β'_{GJM}-unbalanced, respectively, i.e., GJM is unbalanced for the initiator if in the TTP resolve takes priority over abort. Conversely, GJM is unbalanced for the responder if in the TTP abort takes priority over resolve.*

With the external view $(\mathcal{G}^{e}_{GJM}, \beta^{e}_{GJM})$ defined analogously to the ASW protocol (Alice may play the role of the initiator or responder and the TTP gives priority to resolve), again with the messages adapted to the GJM protocol, and $\mathcal{X}$ defined as above, we obtain:

Theorem 4 (GJM is abuse-free). *The external view $(\mathcal{G}^{e}_{GJM}, \beta^{e}_{GJM})$ is $\mathcal{X}$-abuse free. The same is true if the TTP gives priority over abort.*

We prove for all tests Charlie can perform: If a message m that satisfies the test can be derived at some unbalanced state, then another message m' could have been derived already in a previous state which is not a descendent of an unbalanced state and which also satisfies the test; m' is essentially obtained by replacing every occurrence of a real PCS by a fake one. Interestingly, the proof requires to model random coins. Without such coins, one could deduce that given two different messages both passing the same test `pcsver`, one of the messages must be the real PCS while the other one is the fake one.

7 Conclusion

We have proposed a new definition of abuse freeness which involves as key features (i) a specifically designed notion of test performed by the outside party and (ii) a formalization of the assumptions of the outside party by the notion of external view. We have applied our definition to the ASW and GJM protocol, where for the latter protocol we have developed an equational theory to describe the semantics of private contract signatures.

In view of the results in [1, 7, 10], we are currently investigating decidability of abuse freeness as defined here. Also, we study whether balance can be achieved in a real concurrent communication model, given that both the ASW and the GJM protocol are unbalanced no matter what priority the TTP gives to abort and resolve requests received at the same time.

References

1. M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. In *ICALP 2004*, volume 3142 of *LNCS*, pages 46–58. Springer, 2004.
2. M. Abadi and C. Fournet. Mobile Values, New Names, and Secure Communication. In *POPL 2001*, pages 104–115. ACM Press, 2001.
3. N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *IEEE Symposium on Research in Security and Privacy*, pages 86–99, 1998.
4. R. Chadha, M.I. Kanovich, and A.Scedrov. Inductive methods and contract-signing protocols. In *CCS 2001*, pages 176–185. ACM Press, 2001.
5. R. Chadha, S. Kremer, and A. Scedrov. Formal analysis of multi-party contract signing. In *CSFW 2004*, pages 266–279. IEEE Computer Society Press, 2004.
6. R. Chadha, J.C. Mitchell, A. Scedrov, and V. Shmatikov:. Contract Signing, Optimism, and Advantage. In *CONCUR 2003*, volume 2761 of *LNCS*, pages 361–377. Springer, 2003.
7. Y. Chevalier and M. Rusinowitch. Combining Intruder Theories. In *ICALP 2005*, volume 3580 of *LNCS*, pages 639–651. Springer, 2005.
8. D. Dolev and A.C. Yao. On the Security of Public-Key Protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
9. 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.
10. D. Kähler, R. Küsters, and Th. Wilke. Deciding Properties of Contract-Signing Protocols. In *STACS 2005*, volume 3404 of *LNCS*, pages 158–169. Springer-Verlag, 2005.
11. D. Kähler, R. Küsters, and Th. Wilke. A Dolev-Yao-based Definition of Abus-free Protocols. Technical report, IFI 0607, CAU Kiel, Germany, 2006. Available from http://www.informatik.uni-kiel.de/reports/2006/0607.html
12. S. Kremer and J.-F. Raskin. Game analysis of abuse-free contract signing. In *CSFW 2002*, pages 206–220. IEEE Computer Society, 2002.
13. V. Shmatikov and J.C. Mitchell. Finite-state analysis of two contract signing protocols. *Theoretical Computer Science (TCS), special issue on Theoretical Foundations of Security Analysis and Design*, 283(2):419–450, 2002.