The ASW Protocol Revisited: A Unified View [1]

Paul Hankes Drielsma [2] Sebastian Mödersheim [3]

Information Security, ETH Zurich, CH-8092 Zurich, Switzerland

Abstract

We revisit the analysis of the ASW contract signing protocol and use a unified view of the protocol as a whole as a basis to reason about the protocol and its objectives. This line of reasoning yields a simpler and clearer model of agents and protocol objectives which is within the scope of standard security analysis methods, as it does not require fairness constraints and uses only standard authentication and secrecy properties. We also analyse this model for finitely and infinitely many sessions of the protocol using the automated analysis tools OFMC and its extension OFMC-FP.

Key words: Contract Signing, Fair Exchange, Automated Protocol Analysis

1 Introduction

Contract signing protocols like the ASW protocol presented in [1] allow their users to digitally sign contracts without having to meet and sign a document or exchange it via standard mail, which can be very helpful in everyday communication in the business world.

When considering the formal analysis of such protocols, the difficulty arises that they are out of scope of many existing protocol analysis methods: although the act of signing and exchanging messages is standard for these methods, it is difficult to integrate the objectives which contract signing protocols aim to fulfil and the special assumptions upon which they rely.

We present a unified view of the ASW protocol in which the subprotocols are seen as a single protocol with different possible execution paths. While this view is implicit in the protocol models built, for instance, in [5,14], we

[1] This work was partially supported by the FET Open Project IST-2001-39252 and the BBW Project 02.0431, "AVISPA: Automated Validation of Internet Security Protocols and Applications".

[2] Email: drielsma@inf.ethz.ch

[3] Email: moedersheim@inf.ethz.ch

This is a preliminary version. The final version will be published in Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

explicitly describe and reason about the protocol and its objectives based on this high level view, which yields a simpler, more intuitive understanding of the ASW contract signing protocol.

In particular, our model is simpler than other approaches in two respects. First, as a consequence of our view, one does not have to distinguish between an intruder and dishonest and corrupt participants (or even different degrees of corruption), as it is necessary in several other models. Second, adopting this view allows us to reason about the objectives of such a protocol in a simple yet powerful way. We demonstrate, for instance, how several of the security objectives identified by the designers of the protocol can in fact be expressed as standard secrecy and authentication properties, thus "opening the door" to a variety of existing automated protocol analysis tools.

We then apply our unified view concretely, constructing a model of the ASW protocol and formally analysing it using the tools we have developed in our group, the On-the-Fly Model-Checker OFMC and its abstraction-based extension OFMC-FP. Both were developed in the context of the AVISPA project (http://www.avispa-project.org), which offers a toolset for the automated analysis of security protocols and applications. Using OFMC, we can verify the protocol for finitely many sessions (that is, executions of the protocol). Beyond this, we also perform an analysis with OFMC-FP, verifying the protocol for infinitely many sessions.

In the analysis, the tools report an attack on the ASW protocol which results from a subtlety in the specification of the objectives. Adapting these, we were able to verify that the protocol does ensure a slightly weaker objective that still implies the main fair exchange objective.

We observe that, even given the simpler understanding of the protocol that our approach affords, the design of a formal model for automatic analysis presents several challenges. After introducing these, we briefly discuss the results of our automated analysis for both finitely and infinitely many protocol sessions.

2 Background

The ASW protocol, presented by Asokan, Shoup, and Waidner in [1], is an optimistic fair exchange protocol for contract signing intended to enable two parties to commit themselves to a previously agreed upon contractual text. A trusted third party (T3P) is involved *only* if dispute resolution is required (hence the term *optimistic*, which differentiates this approach from others in which an online trusted party is involved in every exchange). In resolving disputes, the T3P issues either a *replacement contract* asserting that he recognises the contract in question as valid, or an *abort token* asserting that he has never issued, and will never issue, a replacement contract. An important requirement of the protocol is that the intruder cannot block messages between an honest agent and the T3P forever.

Exchange subprotocol:

1. $O \rightarrow R: \quad me_1 = Sig_O(V_O, V_R, T, text, h(N_O))$
2. $R \rightarrow O: \quad me_2 = Sig_R(me_1, h(N_R))$
3. $O \rightarrow R: \quad N_O$
4. $R \rightarrow O: \quad N_R$

Abort subprotocol:

1. $O \rightarrow T: \quad ma_1 = Sig_O(aborted, me_1)$
2. $T \rightarrow O: \quad ma_2 = \mathsf{if}\ resolved(me_1)\ \mathsf{then}\ Sig_T(me_1, me_2)$
   $\mathsf{else}\ Sig_T(aborted, ma_1)\ ;\ aborted(ma_1) = true$

Resolve subprotocol:

1. $O \rightarrow T: \quad mr_1 = me_1, me_2$
2. $T \rightarrow O: \quad mr_2 = \mathsf{if}\ aborted(me_1)\ \mathsf{then}\ Sig_T(aborted, me_1)$
   $\mathsf{else}\ Sig_T(me_1, me_2)\ ;\ resolved(me_1) = true$

Fig. 1. The Subprotocols of ASW

2.1 Protocol Objectives

The objectives that such a protocol is supposed to fulfil are manifold. We discuss here the security objectives identified by the designers and later refer to these informal descriptions in the discussion of our verification. Note that [1] refers to "fairness" in the sense of "fair exchange," but we adopt this latter term to avoid confusion with the notion of fairness constraints as understood by the model checking community, which we will use later.

Though [1] presents a framework for the fair exchange of arbitrary items, we consider only the application of this framework to contract signing. We therefore describe the objectives that follow in a manner specialised to our purposes.

1. First and perhaps foremost is the notion of *fair exchange*, which intuitively means that, at the end of a protocol execution, either both parties possess valid contracts, or neither does. In particular, we require that if one agent ends up with only an abort token, then the other cannot be in possession of a valid contract.
2. *Effectiveness* means that, if two honest agents P and Q have finished the protocol and never chose to abandon the current protocol run, then each indeed has a valid contract.
3. The protocol also provides guarantees of *timely completion*: more specifically, the originator and responder of a protocol run can be sure of completion within a finite amount of time.

4. The objective of *non-repudiability*, in the contract signing case, means that the contract contains an implicit proof of the agents' acceptance of the contractual text.
5. *Third party verifiability* dictates that, if the trusted third party should be corrupt and behave in such a way as to compromise fairness of the exchange for one of the protocol participants, then this corrupt behaviour can be proven to an external verifier.

The requirements for fair exchange are often stated in terms of liveness properties of the form "if one agent has a valid contract, then the other either has one as well or is in the position to eventually obtain one." In general, liveness properties are problematic for a variety of verification approaches, in particular those involving infinite state-spaces. In this case, one often approximates liveness properties via safety properties, i.e. if the protocol satisfies the safety property, then it also satisfies the liveness property that was approximated, as it is for instance done in [14]. In §3, we similarly identify appropriate safety properties to check; as we will show, however, from our unified view of the protocol we can directly obtain appropriate safety properties by a simple meta-reasoning.

2.2 Explanation of the protocol

The protocol, shown in Fig. 1, consists of three subprotocols: *exchange*, *abort*, and *resolve*. The former involves only the two protocol participants, the originator O and the responder R, while the latter two are only executed if the trusted third party T is called upon to resolve a dispute. Our notational conventions are as follows: $Sig_O(M)$ denotes the digital signature of message M by agent O, whose public key for signature verification is V_O. The contractual text we call *text*. During the protocol, each party generates a nonce, which we write N_O and N_R for the originator and responder, respectively. Finally, the function h is a cryptographic hash function which is assumed to be collision resistant. We note that the protocol defines two kinds of valid contracts: either the *standard contract* as it is obtained by the exchange subprotocol, or a *replacement contract* issued by the T3P, and both hold equal validity.

The Exchange Subprotocol: If both participants are honest and in the absence of network failures or intruder intervention, after execution of the exchange subprotocol, both will be in possession of a valid standard contract.

Both originator and responder generate nonces N_O and N_R which are called their respective *secret commitments* to the contract. Given these, they compute their so-called *public commitments* by hashing these values, yielding $h(N_O)$ and $h(N_R)$, respectively. The protocol then proceeds in two rounds: in the first, each party expresses his public commitment to the agreed-upon contract but does not disclose his secret commitment. In the second round, they then exchange their respective secret commitments. Each party can then hash this latter and thus verify that the purported secret commitment he receives

indeed corresponds to the public commitment from the first protocol stage. At the end of this exchange, each party is in possession of a valid standard contract of the form me_1, me_2, N_O, N_R.

The Abort Subprotocol: If O does not receive R's reply me_2 within an acceptable time frame (where the definition of "acceptable" is left entirely up to O), he may abort the protocol by invoking the trusted third party. He sends a signed abort request ma_1 indicating that he wishes to abort the exchange.

The T3P is assumed to maintain a permanent database of contracts for which he has been called upon to arbitrate. If he has already asserted the validity of the contract (indicated by $resolved(me_1)$), then he sends the originator a *replacement contract* of the form $Sig_T(me_1, me_2)$. Otherwise, he replies with a so-called *abort token*, signing the originator's abort request and adding an entry in his database of aborted contracts. Such a token does not render an existing contract invalid, but rather serves merely as a promise from the T3P that he has not previously resolved the contract in question and will not do so in the future.

The Resolve Subprotocol: The resolve subprotocol is analogous to the abort but can be invoked by either participant. The parties will request resolution of a contract from the T3P if they do not receive the secret commitment nonce of the other party within a reasonable amount of time. A resolution request includes both messages from the first stage of the exchange subprotocol, me_1 and me_2. If the T3P has already issued an abort token for the contract in question (indicated by $aborted(me_1)$), he replies in kind with an abort token. Otherwise, he issues a replacement contract and indicates in his database that he has resolved the contract.

2.3 The Intruder Model

We adopt the standard intruder model of Dolev and Yao [8] in which the intruder has complete control over the network but cannot break cryptography. In addition, the intruder can play as a normal protocol participant, acting as either the originator or the responder, but not as the T3P.

As we will discuss in more detail in §3, such an intruder model already subsumes the possibility of compromised or dishonest agents that collaborate with the intruder, and we want to show that the interests of honest agents are always ensured, even in protocol runs with the intruder.

3 The Unified View

The key idea behind this paper is to view, and reason about, ASW's subprotocols not in isolation, but rather as *one* protocol. This view is implicit in the construction of the protocol and accordingly also in the models of the protocol built by [5,14]. We explicitly exploit this view to reason about properties of the protocol. More specifically, we consider the abort and resolve subprotocols

$exchange_1.\ O \rightarrow R : me_1$

if *timeout* then	$abort_1.\ O \rightarrow T : ma_1$
	$abort_2.\ T \rightarrow O : ma_2$ (*abort token* or *replacement contract*)
else	

$exchange_2.\ R \rightarrow O : me_2$

$exchange_3.\ O \rightarrow R : N_O$

if *timeout* then	$resolve_1.\ O \rightarrow T : mr_1$
	$resolve_2.\ T \rightarrow O : mr_2$ (*abort token* or *replacement contract*)
else	

$exchange_4.\ R \rightarrow O : N_R$

Fig. 2. Originator role under our unified view of the protocol.

to be *part* of the main exchange protocol. The originator role, for instance, looks then as shown in Fig. 2 (the responder role is similar), where *timeout* represents the event that the agent playing O did not receive a reply to his last message within a reasonable amount of time. We avoid specifying the concrete amount of time after which the timeout shall occur: it may be just a few seconds or a full hour—important for the security of the protocol is only that there *is* such a timeout, so the agent will not wait for an answer forever. Fig. 3 illustrates the internal states of an agent playing the originator role: after sending his initial message, he is in the state in which he waits for a reply until the timeout. If the timeout occurs, then he tries to abort the protocol and thus waits for the answer of the trusted third party (which can be either a replacement contract or the signed abort token). Otherwise (if he receives a reply in time), he carries on with the regular protocol execution and sends his nonce, arriving in a state similar to the one he was in after sending the first message: either there is a reply within the allotted time or he contacts the trusted third party.

This model, though abstract, is thus a faithful representation of a real implementation of the protocol, as agents indeed protect themselves with such internal timeouts. Note that this is related to the possibility of abuse in this contract signing protocol: when the originator has made the first step, the responder has the freedom to either accept the contract by sending the second message, or to reject it by ignoring it. In particular, a dishonest responder could abuse the originator-signed part of the contract in negotiations with other agents (for instance, by soliciting more advantageous contracts from competitors). Note that, unlike for instance the similar GJM [9] protocol, the

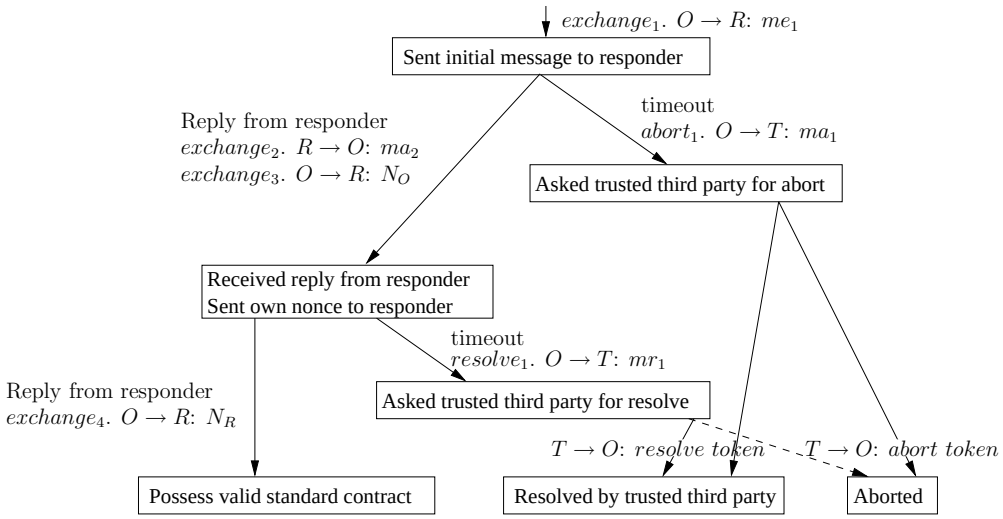

Fig. 3. A state transition view of the originator role. The dashed line represents a transition that can never occur if the trusted server is honest, as we will show below as part of our analysis.

ASW protocol has no means to prevent this abuse by special cryptographic primitives such as private contract signatures. Thus, the timeout is the only way to narrow the window of the originator's vulnerability to abuse attacks as argued in [6].

Although we assume that the intruder can control the entire network according to the Dolev-Yao model, the protocol requires that he cannot block messages between an honest agent and the T3P forever. One could intuitively imagine this situation as follows: all network connections could crash, but an honest agent can still transmit the necessary messages over other media (e.g. ordinary mail) to the T3P and this process cannot take forever.

One could say that we thus have two kinds of fairness assumptions: first, an honest agent will not wait forever for an answer from the other party, and second, the "emergency" protocols with the T3P will eventually succeed. Looking once again at Fig. 3, we can interpret this combination of fairness constraints as the guarantee that an honest agent playing the originator role (and a similar guarantee holds for the responder role) will not stay forever in any of the *intermediate states* (the states of the figure with an outgoing arrow), but will eventually reach one of the three *final states*, (i) where he received the responder's nonce [4] and thus now possesses a valid standard contract, (ii) where he received a valid replacement contract from the trusted third party, or (iii) where he received an abort token.

Thus, the two fairness constraints (timeout and guaranteed reply from the T3P) are sufficient to conclude that every honest agent playing either the

[4] If the responder sends a secret commitment that, when hashed, does not correspond with the public commitment, then this is treated as if he had not sent any message at all (which will probably result in a timeout).

originator or the responder role will eventually end up with either a valid (standard or replacement) contract or an abort token. Roughly speaking, if the agent receives a valid contract, then his interests are ensured, but if he receives an abort token, then it remains to show that nobody else can obtain a valid contract.

This gives us a fresh view on the protocol, as with a simple meta-argumentation we can now go from a model with fairness constraints to a state-reachability property in an infinite state transition system without fairness constraints.

The idea is essentially that we need only to check that if an honest agent reaches his final state of the protocol execution, then the guarantees he should obtain through the protocol are indeed satisfied. In other words, we do not need to consider the guarantees of agents in their intermediate states, since they will eventually reach their final state and we thus spare ourselves any considerations of the form "if the agent can eventually reach a certain state" in the properties we check.

The encoding of the objectives as safety properties is the basis for the deployment of automatic and semi-automatic methods for infinite-state analysis. Also in finite-state analysis, the restriction to safety properties is often essential, e.g. [14] use a similar argumentation that checking the protocol with fairness conditions can be reduced to checking properties of "terminal states" of agents.

4 Encoding of the Objectives

We now want to contrast two models: on the one hand, the model with the fairness constraints described above (i.e. that the agent will eventually get the timeout and the reply from the trusted third party), and on the other hand a model without fairness constraints.

In the model without fairness, the state transitions of the honest agents as shown in Fig. 3 are interpreted as follows: there is no timeout and no guaranteed replies, thus an agent can remain in any intermediate state forever. An agent's local state transition system is thus non-deterministic, as in the states where an honest agent waits for the reply of the other party, he can at any time (i.e. without timeout) begin the abort or resolve protocol, as appropriate.

It follows immediately that, if there is a violation of a safety property in the model with fairness, then there is also a violation in the model without fairness. This shows that our approach is sound in the sense that if we can prove properties in the model without fairness constraints, then they must also hold in the model with fairness constraints. The challenge is to find appropriate safety properties that indeed hold without the fairness constraints and that imply the safety and liveness properties of §2.1 that we wish to check.

We now review the objectives laid out in §2.1 under the new view of the

protocol and show how to encode those objectives that we wish to check as safety properties.

Let us begin with objective (3.), *timely completion*, which means that an honest agent will always eventually reach a valid standard or replacement contract or an abort token. This objective is a direct consequence of the model with fairness constraints, as discussed, and thus does not need to be explicitly checked.

Objective (1.), *fair exchange*, is the main objective of the protocol, namely that either both parties obtain a contract or neither does. We decompose this objective into the following two:

1a. If an honest agent receives an abort token, then nobody (except the trusted third party) can ever obtain a valid standard or replacement contract.

1b. If an agent A (who is not necessarily honest) has obtained a valid standard or replacement contract signed by an honest agent B, then B also possess a valid contract or can obtain one from the trusted third party.

Objective (1a.) is the main objective of our analysis, as it reflects the basic guarantee linked with the abort token. The nice aspect of this objective is that it refers to the final state of an honest agent (which will not subsequently change), not to an intermediate state. It is thus possible to check in the model without fairness that in all states where an agent has reached a final state with an abort token, nobody except the T3P can generate a valid standard or replacement contract matching that abort token. The inability to generate these messages can be expressed in standard protocol analysis approaches by secrecy properties. We can thus reduce the main objective of the protocol to a standard property in protocol analysis (though there is a technical difficulty in the direct application of tools as we will discuss below).

Objective (1b.) is a consequence of objectives (1a.) and (3.): if an agent A possesses a valid contract signed by an honest agent B, then by (3.) B will also eventually reach either an abort token or a valid contract, and by (1a.), if he gets an abort token, then A cannot possess a valid contract, which is a contradiction. Thus B will eventually obtain a valid contract.

Note that the circumstance in which the intruder or a dishonest agent playing the role of the originator can obtain both a valid contract and an abort token (by performing a normal run with an honest agent and asking the T3P for an abort) is not a violation of the objectives above: the abort token only guarantees that the T3P has never and will never resolve this contract but does not render an existing contract invalid.

We now turn to the objective (2.), *effectiveness*. In the view of our model, where every honest agent will eventually reach a final state, effectiveness means that when an honest agent A receives an abort token for a session with an honest agent B, then A or B must have chosen to abort the contract. Since the abort token from the T3P contains the signature of the agent A or B according

to the protocol, it contains the implicit proof that either A or B indeed wants to abort the protocol run. Effectiveness is thus an implicit guarantee due to the form of the abort token, and will therefore not be considered in the later analysis.

Objective (4.), *non-repudiability*, can also be seen as a consequence of the message formats, since in a valid standard or replacement contract, the signatures of both parties are contained and we can thus assume that they agree with the contract text. However, we are also interested in a further analysis, namely an analysis of the authentication properties (or agreement properties in [10]) of ASW. Our analysis will include checks for replay and for confusions of nonces. Such standard authentication properties do not rely on fairness and are thus straightforward to check.

Finally, objective (5.), *third-party verifiability*, is not relevant in our setting, as we assume that the T3P is always honest.

To summarise, we have showed that several objectives of the protocol are direct consequences of its structure, assumptions, and message formats. In essence, an honest agent will receive either an abort token or a valid contract. In the latter case, his interests are ensured, while it remains to show that in the case of an abort token, his interests are also ensured. This amounts to checking that, if an honest agent obtains an abort token, then the valid contract remains secret. Thus due to our view and the meta-reasoning about the protocol we have obtained a model that falls within the realm of standard automated protocol analysis approaches (which often support only secrecy and authentication properties), and we have avoided fairness issues completely. As we will see in the following section, the analysis even of this simplified model is challenging.

Let us conclude this section with a remark on the intruder model. Several approaches distinguish between the intruder and dishonest or corrupted agents (with various degrees of corruption). One of the reasons for this distinction is that the security properties of a protocol usually only hold for sessions between honest agents. In particular, the intruder can play, under his real name, the role of the initiator or the responder in a session with an honest agent; in such a session no security properties are ensured for this honest agent, while this session should not jeopardise the security of other sessions between honest agents.

Due to our simplified view, the main security property that we have to check for ASW, namely (1a.) that the secrecy of the valid contract once an honest agent has received an abort token, should also hold in the case that the other agent is dishonest. (But it is not necessarily ensured that this dishonest agent also has the same security guarantee.) This means that we need not distinguish between various kinds of corruption of dishonest agents.

5 Results

In the previous section, we have developed a unified view of the ASW protocol and a formulation of safety properties. We now apply these ideas concretely, formally specifying and then automatically analysing the protocol using two tools that we have developed in our group.

The first tool is the On-the-Fly Model-Checker (OFMC) which is based on a symbolic representation of the intruder, called the *lazy intruder* [4]. For termination, it requires a bound on the number of sessions that can be performed, but does not require other restrictions, e.g. on the complexity of messages. This is similar to the finite-state analysis of [14].

The second tool OFMC-FP is an extension of OFMC with an abstract fixed-point computation of the reachable states when there is no bound on the number of protocol runs that can be executed, however the complexity of messages is bounded in this method. OFMC-FP is still in a preliminary state at the time of writing: in particular, the user must manually design the employed abstractions.

Both OFMC and OFMC-FP were developed in the context of the AVISPA project and are based on the specification languages developed in this project. The user specifies protocols using the High-Level Protocol Specification Language (HLPSL [2]); these specifications are then automatically translated into the low-level Intermediate Format (IF [3]) which is the input language for automated analysis tools. The first task is thus to specify our view of the ASW protocol in HLPSL.

5.1 Specification

The construction of a formal model of ASW presents three major challenges:

Firstly, an aspect of the protocol that is difficult to model is the database of aborted and resolved contracts maintained by the T3P. Many existing protocol specification languages cannot express this, however HLPSL and IF include the necessary constructs (i.e. finite sets of messages) to model such a database. Moreover, the database cannot be integrated directly in infinite state verification approaches that use abstraction.

Secondly, when using OFMC, we bound the number of sessions of the honest agents to obtain a finite state-space. However, there is no a priori bound on the number of steps that the T3P can perform: in particular, the intruder can exchange an unbounded number of messages with the T3P. In the finite-session analysis with OFMC, we therefore also bound the number of requests from the intruder that the T3P can process.

Finally, although we have reduced the main problem of our analysis to a secrecy question, a further subtlety arises. When an abort token containing an initial message me_1 is issued, we must check for the secrecy of any valid contract that contains me_1. We thus do not state the secrecy of only *one* par-

$e_1.\ I \to R : me_1$	$e_1'.\ I \to R : me_1$
$e_2.\ R \to I : me_2$	$e_2'.\ R \to I : me_2'$
$e_3.\ I \to R : N_I$	Intruder stops communication
$e_4.\ R \to I : N_R$	
$a_1.\ I \to T : ma_1$	$r_1.\ R \to T : \{me_1, me_2'\}$
$a_2.\ T \to I : \textit{abort token}$	$r_2.\ T \to R : \textit{abort token}$

Fig. 4. An attack returned by OFMC: The intruder (denoted as I) aborts a contract that has already been exchanged. In a subsequent run with the same responder, the responder is then unable to resolve the protocol.

ticular message, but of a pattern of messages. Such a feature is currently not supported by HLPSL and IF (or most other protocol specification languages). As a simple way around the problem we specify an honest agent that acts as an observer and flags an error state appropriately.

5.2 Bounded-session Analysis with OFMC

Bounding the number of sessions and the number of requests from the intruder that the T3P processes, we can now directly check whether the transition of the referee ever fires, as well as check authentication properties.

OFMC discovers several authentication problems that were already identified in [14]. First, the protocol does not provide strong authentication (injective agreement as defined by Lowe in [10]), as it has no explicit protection against replay: if the intruder listens to a session of two honest agents, he can replay the exchange protocol with the responder any number of times and obtain valid contracts, each with a fresh responder nonce. However, we think one should assume an implicit replay-protection as part of the contract, e.g. transactions are usually identified by unique transaction numbers. In this scenario, we thus check that the protocol provides weak authentication (also called non-injective agreement, [10]). Weak authentication with respect to the contractual text and the nonces N_0 and N_R is also violated, and OFMC returns an attack trace resulting from the same authentication problems reported in [14]. Finally, weak authentication with respect to only the contractual text is verified by OFMC for various finite test scenarios.

When turning to the secrecy properties we have formulated, OFMC reported the attack displayed in Fig. 4. Assume the intruder I acting as the protocol originator and an honest responder R have completed a run of the exchange subprotocol without involving the T3P (steps e_1 through e_4). Each generated a secret commitment (N_I and N_R, respectively) and ends up with a valid contract in the standard form. Assume now that the intruder issues an abort request for this same contract to the T3P. This latter, having never

before resolved the contract in question, will respond with a valid abort token. The intruder now starts a second session with R, replaying the first message me_1 from the previous session. R generates a new nonce $N_R{}'$ and replies in good faith with the second message $me_2{}'$, including the hash of this new nonce. The intruder, however, does not reply with his nonce but rather ignores R. In turn, R will time out and request resolution of the contract from the T3P, who will respond with an abort token, since the originator message me_1 in question has already been aborted by I. Upon completion of the second protocol session, R therefore has an abort token, while I has a valid contract that corresponds to this abort token (in the sense that it contains the same initial message me_1). Of course, R himself also possesses this contract, having exchanged it with I in the first protocol session.

Formally this violates the objective (1a.): an honest agent has an abort token, while somebody else (the intruder) has a valid contract. It is not really a problem, since the honest agent itself also has this valid contract. In particular, the original objective (1.) is not violated, since both agents indeed possess valid contracts for the same contractual text. This is somewhat surprising, as we now see that there can be situations in which an honest agent indeed possesses both a valid contract and an abort token. Note that objective (1a.) is also considered by [14], who reported problems in the relation of nonces and contracts but did not detect that (1a.) can be violated. The same authors report in [13] an analogous attack on GJM, a similar contract signing protocol [9]. We note also that the improvement of the protocol that they suggest to address the authentication problems described above does not prevent this situation.

We have therefore relaxed objective (1a.) to the following weaker objective (1a′.): "if an honest agent has an abort token, then he also possesses a valid contract or nobody else can obtain one." Note that this property together with (1b.) still implies fair exchange (1.). For this weakened objective, OFMC detected no further attacks.

We also wish to note that an additional check on the responder side for replay of public commitments would prevent this attack.

5.3 Unbounded-session Analysis with OFMC-FP

We have also analysed the protocol using OFMC-FP, which employs a novel abstraction-based fixed-point computation. It was necessary to extend the existing OFMC-FP technique to allow for the integration of the server and its database of contracts. The technique is not completely automatic as the user must himself specify an appropriate abstraction.

OFMC-FP can also detect attacks, but due to the abstraction they may not be possible in our initial concrete model; however, if the security is proven for the abstract model, then this also holds for the concrete model. This is similar to other abstraction-based verification approaches like [7].

Using the OFMC-FP technique, we have established the verification results of OFMC for an unbounded number of participants, sessions, and transitions of the T3P; only the complexity of messages is bounded in this method. In particular, we have first shown that weak authentication on only the contractual text holds. Further, OFMC-FP reveals that a dishonest initiator can obtain a valid standard contract which has also been aborted by the server, which subsumes the violation of (1a.) already reported above. Also, we established that in all such situations, the other party also obtained a valid standard contract and thus verified the weakened property (1a′.). We note, however, that as part of the analysis, we have found that, for property (1a′.) to hold, an important prerequisite is the fact that an honest agent playing in the originator role can never obtain an abort token from the T3P as a reply to a resolve request, explaining the dashed line in Fig. 3.

6 Related Work and Conclusion

A substantial body of literature exists on the analysis of contract signing protocols, in particular ASW and the similar GJM protocol.

Shmatikov and Mitchell undertake an analysis (with a bounded number of sessions) of both the ASW and the GJM protocol using the finite-state model-checker Murφ [12,13,14]. Their approach is closest to ours, namely they also follow the principal idea to reduce the problem of fairness properties to safety properties. While they also implicitly employ the unified view of the protocol, they do not use it explicitly to perform meta-reasoning about the protocol. Another difference is that they also distinguish the intruder from dishonest agents (with varying degrees of corruption). Moreover, they check abuse-freeness for the GJM protocol (while ASW is not designed to ensure abuse-freeness).

Das and Dill [7] were the first to describe the automated analysis of a contract signing protocol, GJM, for an unbounded number of session using abstractions and the model-checker Murφ. Similar to our analysis, they focus on the property of fair exchange.

Kremer and Raskin [11] focus on abuse-freeness and argue that, under certain assumptions, even the ASW protocol is abuse free (while this was not one of the original objectives of the protocol designers). To appropriately model strategies of malicious agents and strategic advantages over other agents, they use a game theoretic method and alternating transition systems. They perform an automated analysis using the model-checker Mocha; note that they do not consider multiple runs of the protocol in parallel and adopt the strong typing assumption. [5]

There are several works (which do not focus on automated analysis) on

[5] The strong typing assumption is a stronger restriction than bounding the message size as it is done by OFMC-FP; no similar restriction is necessary for OFMC.

reasoning about such protocols and their guarantees, in particular optimism and fairness [5,6]. The employed models are considerably more detailed than ours in that they explicitly use time-outs and distinguish intruder and (different kinds of) dishonest agents. Also, here a similar view to ours is often taken, though to our knowledge not been used to reason about the protocol and its objectives.

We adopt this unified view and use it explicitly to reason about the protocol's objectives and thereby reduce several of them to standard authentication and secrecy properties which are easily digestible by many automated analysis tools for which protocols like ASW would previously have been out of scope.

Yet, as described in §5.1, even under the unified view, the specification and analysis with existing protocol analysis tools is challenging, in particular this holds for the modelling of the trusted third party that maintains a data-base of aborted and resolved contracts.

Our analysis demonstrated the same authentication failures discussed in [14] and also revealed that a guarantee that one might intuitively expect of the protocol, objective (1a.), is in fact violated by the attack we present. We can, however, show that, beyond these problems, the protocol is secure.

While we have focused on ASW in our work to date, we are optimistic that the benefits offered by the adoption of such a unified view will be applicable to similar protocols as well. In general, the meta-reasoning we perform regarding security objectives can help not only to better understand the objectives of a given protocol, but, as we have seen, can also identify potential ways in which seemingly complicated objectives can be reduced to more standard notions such as authentication and secrecy. In this way, we hope to extend the applicability of existing methods for the formal analysis of security protocols.

References

[1] N. Asokan, V. Shoup, and M. Waidner. Asynchronous protocols for optimistic fair exchange. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 86–99, 1998.

[2] AVISPA. Deliverable 2.1: The High-Level Protocol Specification Language. Available at `www.avispa-project.org/delivs/2.1/`, 2003.

[3] AVISPA. Deliverable 2.3: The Intermediate Format. Available at `www.avispa-project.org/delivs/2.3`, 2003.

[4] D. Basin, S. Mödersheim, and L. Viganò. An On-The-Fly Model-Checker for Security Protocol Analysis. In E. Snekkenes and D. Gollmann, editors, *Proceedings of ESORICS'03*, LNCS 2808, pages 253–270. Springer-Verlag, 2003. Available at `www.avispa-project.org`.

[5] R. Chadha, M. Kanovich, and A. Scedrov. Inductive methods and contract-signing protocols. In *P. Samarati, editor, Proceedings, 8th ACM Conference on*

Computer and Communications Security, pages 176–185, New York, November 2001. ACM Press.

[6] R. Chadha, J. C. Mitchell, A. Scedrov, and V. Shmatikov. Contract signing, optimism, and advantage. In *CONCUR: 14th International Conference on Concurrency Theory*. LNCS, Springer-Verlag, 2003.

[7] S. Das and D. L. Dill. Successive approximation of abstract transition relations. In *Proceedings of the 16th Annual IEEE Symposium on Logic in Computer Science (LICS-01)*, pages 51–60, Los Alamitos, CA, June 16–19 2001. IEEE Computer Society.

[8] D. Dolev and A. Yao. On the Security of Public-Key Protocols. *IEEE Transactions on Information Theory*, 2(29), 1983.

[9] J. A. Garay, M. Jakobsson, and P. MacKenzie. Abuse-free optimistic contract signing. In *Proc. 19th International Advances in Cryptology Conference – CRYPTO '99*, pages 449–466, 1999.

[10] G. Lowe. A hierarchy of authentication specifications. In *Proceedings of the 10th IEEE Computer Security Foundations Workshop (CSFW'97)*, pages 31–43. IEEE Computer Society Press, 1997.

[11] J. Raskin and S. Kremer. Game analysis of abuse-free contract signing. In *15th IEEE Computer Security Foundations Workshop*, pages 206–220. IEEE Computer Society Press, June 2002.

[12] V. Shmatikov and J. C. Mitchell. Analysis of a fair exchange protocol. In *Proceedings of the 1999 FLoC Workshop on Formal Methods and Security Protocols*, Trento, Italy, 1999.

[13] V. Shmatikov and J. C. Mitchell. Analysis of abuse-free contract signing. In *Proceedings of the 4th International Conference on Financial Cryptography (FinCrypto '00)*, 2001.

[14] V. Shmatikov and J. C. Mitchell. Finite-state analysis of two contract signing protocols. *Theoretical Computer Science*, 283(2):419–450, June 2002.