An Optimized Intruder Model for SAT-based Model-Checking of Security Protocols

Alessandro Armando and Luca Compagna

AI-Lab, DIST – Università degli Studi di Genova,
Viale Causa 13, 16145 Genova, Italy.
Tel: +39 0103536545, Fax: +39 0103532948
{armando,compa}@dist.unige.it

Abstract

In previous work we showed that automatic SAT-based model-checking techniques based on a reduction of protocol (in)security problems to a sequence of propositional satisfiability problems can be used to effectively find attacks on protocols. In this paper we present an optimized intruder model that may lead in many cases to shorter attacks which can be detected in our framework by generating smaller propositional formulae. The key idea is to assume that some of the abilities of the intruder have instantaneous effect, whereas in the previously adopted approach all the abilities of the intruder were modeled as state transitions. This required non trivial extensions to the SAT-reduction techniques which are formally described in the paper. Experimental results indicate the advantages of the proposed optimization.

1 Introduction

In the last decade we have witnessed a dramatic speed-up of SAT solvers: problems with thousands of variables are now solved routinely in milliseconds by state-of-the-art SAT solvers. This has led to breakthroughs in important areas such as planning and model-checking. Motivated by these results, in [2,3,4] we proposed reductions of protocol (in)security problems to satisfiability problems in propositional logic that can be used to effectively find attacks on protocols. We have developed a model-checker, SATMC, based on these ideas and experimental results obtained by running SATMC against security protocols drawn from the Clark-Jacob's library [9] confirm the effectiveness of the approach.

In this paper we propose an optimized intruder model that leads in many cases to shorter attacks which can be detected in our framework by generating smaller propositional formulae. The key idea is to assume that some of the abilities of the intruder have instantaneous effect as opposed to the previously

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

adopted approach in which all the abilities were modeled as state transitions. This required non trivial extensions to the SAT-reduction techniques which are formally described in the paper. We have implemented the proposed optimization within SATMC. Experimental results on protocols drawn from the Clark-Jacob's library clearly indicate the advantages of the proposed optimization.

Outline of the paper. We start in Section 2 by presenting, via a well-known authentication protocol, our model with particular emphasis to an optimized intruder model based on the concept of axiom. In Section 3 we define the notion of protocol insecurity problem with axioms. Section 4 is devoted to the formal description of the automatic compilation of protocol insecurity problems with axioms into a set of propositional satisfiability problems. Experimental results and some implementation detail are discussed in Section 5. We conclude in Section 6 with some final remarks and a discussion of the future work.

2 Modeling Security Protocols

Although the general verification problem—to prove that the security protocol satisfies the security guarantees for which it has been designed in a scenario with an arbitrary number of parallel sessions—is undecidable [12,13], for many protocols, verification can be reduced to verification of a bounded number of sessions. Moreover, even for those protocols that should be checked under a unbounded number of concurrent protocol executions, violations in their security requirements often exploit only a small number of parallel sessions. For these reasons, in many case of interest it is sufficient to consider a finite and small number of parallel sessions. For instance, all known attacks on the protocols in the Clark-Jacob's library involve at most two concurrent sessions.

We model the concurrent execution of a security protocol by means of a state transition system specified in the IF [1] declarative language based on multi-set rewriting which is particularly amenable to formal analysis [7,15]. States are represented by sets of atomic formulae of a sorted first order language called *facts* and transitions by means of *labeled rewrite rules* over sets of facts. In our setting, we assume that the network is controlled by means of the very general Dolev-Yao intruder [11]. [1] In this model the intruder has the ability to eavesdrop and divert messages as well as that to compose, decompose, encrypt, and decrypt messages (provided the encryption keys are known). Finally, he can send those messages to other participants with a false identity. Besides this, we make the standard assumptions of *perfect cryptography* i.e. an encrypted message can be decrypted only by using the appropriate decryption key and of *strong typing* i.e. agents only accept type-correct messages and

[1] It is worth pointing out that the model is not bound to the Dolev-Yao intruder, but, on the contrary, it is expressive enough to get generic intruder models suited for a variety of communication networks such as secure channels, wireless channels, etc.

therefore type confusion is not allowed. [2]

In order to clarify our presentation, let us consider the example of the Needham-Schroeder Public-Key (NSPK) authentication protocol [16]. In the common Alice&Bob notation, the NSPK protocol looks like follow:

$$\begin{array}{llcll} (1) & A & \rightarrow & B & : \quad \{A, Na\}_{Kb} \\ (2) & B & \rightarrow & A & : \quad \{Na, Nb\}_{Ka} \\ (3) & A & \rightarrow & B & : \quad \{Nb\}_{Kb} \end{array}$$

where A and B are the roles involved in the protocol; Ka and Kb are the public keys of A and B, respectively; and Na and Nb are nonces [3] generated, respectively, by A and B. Notice that, the above high level protocol specification describes a kind of template $NSPK(A, B, Ka, Kb, Na, Nb)$ parametrized by some free variables [4] appearing in it. A ground instance of the security protocol template represents a session of the protocol. Successful execution of the NSPK protocol should convince both A and B that they have been talking to each other. The rationale is that only B and A could have formed the appropriate response to the message issued in (1) and in (2), respectively. In fact, a malicious agent I can deceit *bob* (an instance of B) into believing that he is talking with *alice* (instance of A) whereas he is talking with I. This is achieved by executing concurrently two sessions $NSPK(alice, I, ka, ki, na, ni)$ and $NSPK(alice, bob, ka, kb, na2, nb)$ of the protocol and using messages from one session to form messages in the other as illustrated by the following protocol trace:

$$\begin{array}{llcll} (1.1) & alice & \rightarrow & I & : \quad \{alice, na\}_{ki} \\ (2.1) & I(alice) & \rightarrow & bob & : \quad \{alice, na\}_{kb} \\ (2.2) & bob & \rightarrow & I(alice) & : \quad \{na, nb\}_{ka} \\ (1.2) & I & \rightarrow & alice & : \quad \{na, nb\}_{ka} \\ (1.3) & alice & \rightarrow & I & : \quad \{nb\}_{ki} \\ (2.3) & I(alice) & \rightarrow & bob & : \quad \{nb\}_{kb} \end{array}$$

where $I(alice)$ indicates the intruder pretending to be *alice*. At the end of the above trace *bob* believes he has been talking with *alice*, but this is obviously not the case.

[2] As pointed out in [14], in many case of interest, type-flaw attacks can be prevented by tagging the fields of a message with information indicating its intended type.

[3] *Nonces* are numbers randomly generated by principals and they are intended to be used *only once*.

[4] Free variables are indicated by means of capital letters excepted I that is used to indicate the malicious agent.

Let us describe in our setting the state transition system and the security requirement of the above example.

Facts. The facts useful to describe the NSPK protocol are the following:

- $ik(T)$, meaning that the intruder knows the message T;
- $fresh(N)$, meaning that the nonce N has not been used yet;
- $m(J, S, R, T)$, meaning that principal S has (supposedly) [5] sent message T to principal R at protocol step J; and
- $w(J, S, R, [T_1, \ldots, T_k], C)$, representing the state of execution of principal R at step J of session C; in particular it means that R knows the messages $T_1, \ldots, T_k$ at step J of session C, and—if $J \neq 0$—also that a message from S to R is awaited for step J of session C to be executed.

Initial State. The initial state of the system representing two concurrent sessions of the NSPK is: [6]

$$w(0, a, a, [a, i, ka, ka^{-1}, ki], 1) \tag{1}$$
$$. w(0, a, a, [a, b, ka, ka^{-1}, kb], 2) . w(1, b, a, [b, a, kb, kb^{-1}, ka], 2) \tag{2}$$
$$. fresh(nc(n1, 1)) \tag{3}$$
$$. fresh(nc(n1, 2)) . fresh(nc(n2, 2)) \tag{4}$$
$$. ik(i) . ik(a) . ik(b) . ik(ki) . ik(ki^{-1}) . ik(ka) . ik(kb) \tag{5}$$

The fact (1) represents the initial state of the honest agent a that plays the role of initiator in session 1 and knows at the beginning her identity, the identity of intruder (the agent she would like to talk with), her public and private keys, [7] and the intruder public key. Facts (2) represent the initial state of the honest agents a and b involved as initiator and responder, respectively, in session 2. Facts (3) and (4) state the initial freshness of the nonces $nc(n1, 1)$, $nc(n1, 2)$, and $nc(n2, 2)$. Notice that, since we bound the number of parallel sessions also the number of fresh terms to be taken into account in the protocol analyses can be computed and bounded in advance. Facts (5) represent the information initially known by the intruder (commonly, in a asymmetric cryptosystem, its identity, its public and private keys as well as the identities of honest agents and their public keys). Notice that, according to the standard close world assumption all the facts that do not occur in the set representing the initial state are considered false.

[5] As we will see, since the intruder may fake other principals' identity, the message might have been sent by the intruder.

[6] To improve readability we use the "." operator as set constructor. For instance, we write "$x . y . z$" to denote the set $\{x, y, z\}$.

[7] Notice that with ka^{-1} we indicate the inverse of the public key ka.

Goal Predicates. A security protocol is intended to enjoy specific security properties. In our example this property is the ability of authenticating the responder with the initiator and vice versa. A security property can be specified by providing goal predicates representing a set of "bad" states. For instance, it is immediate to see that any state in which b believes to have completed a session with a, while a has not started this session with him, represents a violation of the expected authentication property between b and a. This set of bad states can be easily represented by the goal predicate $w(1, a, b, [b, a, kb, kb^{-1}, ka], s(2)) \wedge w(0, a, a, [a, b, ka, ka^{-1}, kb], 2)$. [8]

Labeled Rewrite Rules. As mentioned above, labeled rewrite rules over sets of facts are used to specify how the transition system evolves. In particular, we distinguish between rules modeling the behavior of honest agents (so called protocol rules) and rules representing the Dolev-Yao intruder (so called intruder rules). These rules will be described in the next sections. Concerning the notation used, let $M1$ and $M2$ be messages, then the terms $\{M1\}_{M2}$ and $\langle M1, M2\rangle$ represent the asymmetric encryption of $M1$ using $M2$ as asymmetric key and the pairing of the $M1$ and $M2$, [9] respectively.

2.1 Modeling the behavior of Honest Participants

Honest participants strictly behave according to the protocol. The following rewrite rule models the activity of sending the first message of the NSPK protocol:

$$fresh(nc(n1, S)) \centerdot w(0, A, A, [A, B, Ka, Ka^{-1}, Kb], S) \xrightarrow{step_0(A,B,Ka,Kb,S)} w(2, B, A, [nc(n1, S), A, B, Ka, Ka^{-1}, Kb], S) \centerdot m(1, A, B, \{A, nc(n1, S)\}_{Kb})$$

Notice that, the nonce $nc(n1, S)$ is added to the knowledge of A for subsequent use. The receipt of the message and the reply of the responder is modeled by:

$$fresh(nc(n2, S)) \centerdot m(1, A, B, \{A, Na\}_{Kb}) \centerdot w(1, A, B, [B, A, Kb, Kb^{-1}, Ka], S) \xrightarrow{step_1(A,B,Ka,Kb,Na,S)} w(3, A, B, [Na, nc(n2, S), B, A, Kb, Kb^{-1}, Ka], S) \centerdot m(2, B, A, \{Na, nc(n2, S)\}_{Ka})$$

The state of the responder is updated by increasing the protocol step and by extending the knowledge with the acquired informations namely the nonce

[8] Notice that with $s(C)$ we indicate the next execution of session C.

[9] To improve readability we write $M1, M2$ in place of $\langle M1, M2\rangle$ where the pairing message can be easily evinced from the context.

sent by the initiator and the nonce freshly generated by the responder itself. The third step of the protocol is modeled by:

$$m(2, B, A, \{Na, Nb\}_{Ka}) \\ . w(2, B, A, [Na, A, B, Ka, Ka^{-1}, Kb], S) \xrightarrow{step_2(A,B,Ka,Kb,Na,Nb,S)} \\ w(0, A, A, [A, B, Ka, Ka^{-1}, Kb], s(S)) \\ . m(3, A, B, \{Nb\}_{Kb})$$

Notice that, after the application of this rule, A completes her part in the session S and she is eventually ready to start an other session $s(S)$ of the protocol with B. The final step of the protocol is modeled by:

$$m(3, A, B, \{Na\}_{Kb}) \\ . w(3, A, B, [Na, Nb, B, A, Kb, Kb^{-1}, Ka], S) \xrightarrow{step_3(A,B,Ka,Kb,Na,Nb,S)} \\ w(1, A, B, [B, A, Kb, Kb^{-1}, Ka], s(S))$$

After the application of this rule also B finishes his part in the session S and he is eventually ready to communicate with A in an other session $s(S)$.

2.2 Modeling the behavior of the Intruder

Contrary to honest agents that execute faithfully each statement specified by the protocol, the Dolev-Yao intruder has many degrees of freedom. By observing all the traffic in the network, it extends its knowledge and from such a knowledge it can compose and send fraudulent messages to honest participants. However, many of these messages can be of no interest for the analysis of the protocol. In fact, as previously mentioned, we focus on reachability problems with a finite number of parallel sessions in which honest agents only react to a message if, and only if, this message has the type and the pattern expected by the receiver. The set of such "interesting" messages is finite. For instance, the following partially instantiated rule modeling the ability of the Dolev-Yao intruder of pairing messages

$$ik(a) . ik(M) \xrightarrow{pairing(a,M)} ik(a) . ik(M) . ik(\langle a, M\rangle))$$

can be applied in the initial state of the NSPK by substituting M with a and in every successor state extending the knowledge of the intruder with messages of the form $\langle a, \langle a, \ldots\rangle\rangle$. However, such messages are not "interesting" for the analysis of our example and, therefore, also the rules applied for composing them are useless. As a matter of fact, *diverting*, *impersonating* and *decomposing* rules are sufficient to model the Dolev-Yao intruder and it is easy to see that this optimization is correct as it preserves the existing attacks and does not introduce new ones.

Diverting Rules. The following rule models the ability of the intruder of diverting the information exchanged by the honest participants:

$$m(I, A, B, M) \xrightarrow{divert(A,B,I,M)} ik(M)$$

It states that, if a message has been sent on the communication channel, then the intruder can read its content and remove it from the channel.

Impersonating Rules. The observation that most of the messages generated by a strictly compliant Dolev-Yao intruder are rejected by the receiver as non-expected or ill-formed suggests to restrict these rules so that the intruder sends only messages matching the patterns expected by the receiver. For each protocol rule of the form

$$\ldots \;.\; m(I, A, B, M) \;.\; w(I, A, B, Knw, S) \xrightarrow{step_I(\ldots)} \ldots$$

and for each possible set of messages $\{M_{1,k}, \ldots, M_{j_k,k}\}$ (let m be the number of such sets, then $k = 1, \ldots, m$ and $j_k > 0$) from which the Dolev-Yao intruder would be able to build a message M' that matches (and complies with the type of) M, we add a new rule of the form

$$\begin{aligned} \ldots . w(I, A, B, Knw, S) . i(A) . i(B) . i(M_{1,k}) . \ldots . i(M_{j_k,k}) \\ \xrightarrow{impersonate_{I,k}(\ldots)} \ldots . m(I, A, B, M') . w(I, A, B, Knw, S) \\ . i(A) . i(B) . i(M_{1,k}) . \ldots . i(M_{j_k,k}) . i(M') \end{aligned}$$

This rule states that if agent B is waiting for a message M from A and the intruder is able to compose a message M' matching (and complying with the type of) M, then the intruder can impersonate A and send M'. This optimization, first introduced in [15], often reduces the number of rule instances in a dramatic way (for more details see [2]).

It is worth pointing out that, by applying the *step compression* optimization proposed in [5], it is possible to merge the above intruder rules with protocol rules reducing dramatically the dimension of the search space to be explored. See [2] for more details on the implementation of this optimization in our setting.

Decomposing Rules. In order to be able to apply either the impersonating rules or the step compressed rules, the intruder must be provided with the smallest set of rules modeling its ability of decomposing every sub-message of

messages exchanged in the parallel sessions of the protocol. For instance, the rules useful to decompose the first message of the NSPK are the following:

$$ik(\{M\}_K) \centerdot ik(K^{-1}) \xrightarrow{decrypt(K,M)} ik(\{M\}_K) \centerdot ik(K^{-1}) \centerdot ik(M)$$
$$ik(\langle M1, M2\rangle) \xrightarrow{decompose(M1,M2)} ik(M1) \centerdot ik(M2)$$

The first rule states that if the intruder knows both a message encrypted with the key K and the decryption key K^{-1}, then the transition system can move in a state where the knowledge of the intruder is extended with the content M of the message. Similarly, the second rule states that if the intruder knows the pairing message $\langle M1, M2\rangle$, then the rule can be applied to reach a state in which the knowledge of the intruder is extended with the messages $M1$ and $M2$.

Especially for industrial-scale security protocols in which messages can have a complex structure, the number of decomposing rule instances to be applied in order to find out an attack can be significant. As we will see in Section 4, the size—in terms of atoms and clauses—of the generated propositional formulae increases linearly with the length of the attack. Since the complexity of the SAT problem grows exponentially according to the increasing of the atoms number, it is critical to reduce such a length. The optimization proposed in this paper is based on the concept of *axiom*. An axiom is a formula that states a relation between the facts of the transition system and that holds in each state of the transition system.[10] It turns out that axioms are particularly suited to represent relations between intruder knowledge facts. For instance, the axiom

$$ik(\{M\}_K) \wedge ik(K^{-1}) \supset ik(M)$$

states that, every time the intruder knows both a message encrypted with the key K and the decryption key K^{-1}, then it knows *instantaneously* also the content M of the message. The main difference between an axiom and a rule is in the fact that the former must hold in every state of the transition system, while the latter is not forced to be executed even if all its preconditions are satisfied and it spends a transition for being executed. As a consequence by replacing decomposing rules with appropriate decomposing axioms we obtain a twofold benefit: the size of the propositional formulae generated can decrease as well as the number of transition steps to be applied for finding attacks. Notice that, given the set of intruder decomposing rules $\mathcal{DR} \subset \mathcal{R}$, it is straightforward to build the set of decomposing axioms $\mathcal{DA}$:

$$\mathcal{DA} = \{p_1 \wedge \cdots \wedge p_j \supset c \mid (p_1 \centerdot \ldots \centerdot p_j \xrightarrow{\ell} C) \in \mathcal{DR}, c \in C\}$$

It can be proved both that this optimization is correct as it preserves the

[10] Notice that, if an axiom contains variables, then they are intended universally quantified.

existing attacks and does not introduce new ones, and that it leads to equal or shorter attacks.

3 Protocol Insecurity Problems with Axioms

The concepts presented in Section 2 can be recast into the concept of protocol insecurity problem with axioms. A *protocol insecurity problem with axioms* is a tuple $\Xi = \langle \mathcal{F}, \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{I}, \mathcal{G} \rangle$ where $\mathcal{F}$ is a set of atomic formulae of a sorted first-order language called *facts*, $\mathcal{L}$ is a set of function symbols called *rule labels*, $\mathcal{A}$ is a set of axioms of the form $p_1 \wedge \ldots \wedge p_j \supset c$, where $p_1, \ldots, p_j, c$ are in $\mathcal{F}$, and $\mathcal{R}$ is a set of rewrite rules of the form $L \xrightarrow{\ell} R$, where L and R are finite subsets of $\mathcal{F}$ such that the variables occurring in R occur also in L, and ℓ is an expression of the form $l(\underline{x})$ where $l \in \mathcal{L}$ and $\underline{x}$ is the vector of variables obtained by ordering lexicographically the variables occurring in L. The components $\mathcal{I}$ and $\mathcal{G}$ of a protocol insecurity problem with axioms are the initial states and a boolean formula representing the bad states of the protocol, respectively. In this setting, a state is represented by the set of facts $S \subseteq \mathcal{F}$ that are true in it. As a consequence, all the facts that are not in S are considered false (close world assumption). Moreover, all the states of Ξ must satisfy the set of axioms $\mathcal{A}$. Formally, $states(\Xi) = \{S \mid S \subseteq \mathcal{F}, S \models \mathcal{A}\}$.[11] Let S be a state and $(L \xrightarrow{\ell} R) \in \mathcal{R}$, if σ is a substitution such that $L\sigma \subseteq S$ and $S' = (S \setminus L\sigma) \cup R\sigma$ is such that $S' \models \mathcal{A}$, then one possible next state of S is S' and we indicate this with $S \overset{\ell\sigma}{\rightsquigarrow} S'$. We assume the rewrite rules are *deterministic* i.e. if $S \overset{\ell\sigma}{\rightsquigarrow} S'$ and $S \overset{\ell\sigma}{\rightsquigarrow} S''$, then $S' = S''$. A *solution to a protocol insecurity problem with axioms* Ξ, called *attack*, is a sequence of rules $\ell_1\sigma_1, \ldots, \ell_n\sigma_n$ such that $S_i \overset{\ell_i\sigma_i}{\rightsquigarrow} S_{i+1}$ for $i = 1, \ldots, n$ with $S_1 \subseteq \mathcal{I}$ and $S_n \models \mathcal{G}$. The *length* of an attack is the number of rules occurring in it.

It is convenient to relax the definition of the transition relation associated to a protocol insecurity problem with axioms by allowing parallel execution of rules, though preserving the interleaving semantic of the model. This means that every trace in which some rules are executed simultaneously can be linearized into a longer trace in which all the rules are executed sequentially. In order to guarantee the interleaving semantic of the model we define $\oplus$ to be the (commutative) relation of mutual exclusion (*mutex* for short) between rules. With the introduction of axioms in the concept of protocol insecurity problem, the mutex relation requires the construction of a directed graph $G_{\mathcal{A}} = \langle \mathcal{F}, E \rangle$ representing the dependencies between facts wrt the set of axioms. The basic idea is that for each substitution σ and $(p_1 \wedge \cdots \wedge p_j \supset c) \in \mathcal{A}$, the pairs $\langle c\sigma, p_1\sigma \rangle, \ldots, \langle c\sigma, p_j\sigma \rangle$ are in the set of edges E. A fact c is *dependent* from a fact p, denoted with $c \hookrightarrow_{\mathcal{A}} p$, iff there is in $G_{\mathcal{A}}$ a path from the node c to the node p. Let $dep_{\mathcal{A}}(c) = \{p \mid c \hookrightarrow_{\mathcal{A}} p\} \cup \{c\}$ be the set of facts from

[11] Notice that, $S \models \mathcal{A}$ iff for each substitution σ and $(p_1 \wedge \cdots \wedge p_k \supset c)$ in $\mathcal{A}$, it holds that $S \models (p_1\sigma \wedge \cdots \wedge p_k\sigma) \supset c\sigma$.

which c depends wrt $\mathcal{A}$ and, similarly, let $dep_{\mathcal{A}}(C) = \bigcup_{c \in C} dep_{\mathcal{A}}(c)$ be the set of facts from which the facts in C depend wrt $\mathcal{A}$. The mutex relation $\oplus$ is thus defined as follow: for each $L_1 \xrightarrow{\ell_1} R_1$, $L_2 \xrightarrow{\ell_2} R_2$ in $\mathcal{R}$ and for each pair of substitutions σ_1 and σ_2 such that $\ell_1\sigma_1 \neq \ell_2\sigma_2$, then $\ell_1\sigma_1 \oplus \ell_2\sigma_2$ iff $L_1\sigma_1 \cap dep_{\mathcal{A}}((L_2\sigma_2 \setminus R_2\sigma_2)) \neq \emptyset$ or $L_2\sigma_2 \cap dep_{\mathcal{A}}((L_1\sigma_1 \setminus R_1\sigma_1)) \neq \emptyset$. If $\ell_1 \oplus \ell_2$ we say that ℓ_1 and ℓ_2 are *conflicting* rule instances. The parallel execution of rules is thus allowed by the following relaxed definition of transition. Let S be a state, if there exist a set of rules $\{L_1 \xrightarrow{\ell_1} R_1, \ldots, L_m \xrightarrow{\ell_m} R_m\} \subseteq \mathcal{R}$ and a set of substitutions $\{\sigma_1, \ldots, \sigma_m\}$ such that, by defining $L = (\bigcup_{i=1}^{m} L_i\sigma_i)$ and $R = (\bigcup_{i=1}^{m} R_i\sigma_i)$, (i) for each $i, j = 1, \ldots, m$ with $i \neq j$, then $\ell_i\sigma_i \oplus \ell_j\sigma_j$ does not hold (non-conflicting rule instances), (ii) $L \cap R = \emptyset$, (iii) $L \subseteq S$, and (iv) $((S \setminus L) \cup R) \models \mathcal{A}$, then one possible next state of S is $S' = ((S \setminus L) \cup R)$. We indicate this with $S \overset{\Lambda}{\underset{\mathrm{P}}{\rightsquigarrow}} S'$ where $\Lambda = \{\ell_1\sigma_1, \ldots, \ell_m\sigma_m\}$. Similarly, the definition of solution to a protocol insecurity problem can be recast into the concept of *partial-order attack*. A partial-order attack to a protocol insecurity problem Ξ is a sequence of sets of rewrite rule instances $\Lambda_1, \ldots, \Lambda_n$ such that $S_i \overset{\Lambda_i}{\underset{\mathrm{P}}{\rightsquigarrow}} S_{i+1}$ for $i = 1, \ldots, n$ with $S_1 \subseteq \mathcal{I}$ and $S_n \models \mathcal{G}$. The length of a partial-order attack corresponds to the number of sets in the sequence. It can be proved that a partial-order attack to Ξ corresponds to a set of attacks to Ξ. Moreover, let $reach(S, \rightsquigarrow)$ and $reach(S, \underset{\mathrm{P}}{\rightsquigarrow})$ the set of states that can be reached from the state S by means of the transition relations $\rightsquigarrow$ and $\underset{\mathrm{P}}{\rightsquigarrow}$, respectively. It can be proved that $reach(S, \rightsquigarrow) = reach(S, \underset{\mathrm{P}}{\rightsquigarrow})$.

4 Automatic SAT-Compilation of Protocol Insecurity Problems with Axioms

Let $\Xi = \langle \mathcal{F}, \mathcal{L}, \mathcal{R}, \mathcal{A}, \mathcal{I}, \mathcal{G} \rangle$ be a protocol insecurity problem with axioms such that (i) the sets of facts, axioms, and rules are finite, (ii) the set of axioms $\mathcal{A}$ does not entail any propositional equivalence between facts (i.e. for all the facts $p_1, \ldots, p_j, c$ in $\mathcal{F}$, it holds that $\not\models_{\mathcal{A}} (p_1 \wedge \cdots \wedge p_j) \equiv c$), and let k be a positive integer, then it is possible to build a propositional formula Φ^k_Ξ such that any model of Φ^k_Ξ corresponds to a partial-order attack of length k solution of Ξ. It is worth pointing out that the axioms modelling the ability of the intruder of decomposing messages satisfy the above requirements (i) and (ii).

In previous work we described how a protocol insecurity problems without axioms is compiled into a set of SAT formulae using two encoding techniques: the first belongs to the family of so-called *linear encodings* [2,3], the second is the more sophisticated *graphplan-based encoding* [4]. Let us see how to extend our approach to be able to deal with the axioms described in Sections 2 and 3. The basic idea does not change and consists into adding an additional time-index to the rules and facts to indicate the state at which the rule begins or the fact holds. Facts are thus indexed by 0 through k and rules by 0 through

$k-1$. If p is a fact or an rule and i is an index in the appropriate range, then p^i is the corresponding time-indexed propositional variable. However, with the introduction of the axioms into the concept of protocol insecurity problem, significant alterations must to be done in the encodings schemes. In the rest of this section we will formally describe how to compile a protocol insecurity problem with axioms into SAT by using the linear encoding technique. Similar concepts apply also to the graphplan-based encoding technique. Notice also that the encoding technique we are going to present can be applied also to compile protocol insecurity problems without axioms into SAT simply by setting $\mathcal{A} = \emptyset$.

Similarly to the classic bounded model-checking approach [6], the propositional formula Φ^k_Ξ is defined by

$$\Phi^k_\Xi = I(\underline{f}^0) \wedge \bigwedge_{i=0}^{k-1} T_i(\underline{f}^i, \underline{\ell}^i, \underline{f}^{i+1}) \wedge G(\underline{f}^k)$$

where $\underline{f}$ and $\underline{\ell}$ are vectors of facts and rules respectively [12] and

- $I(\underline{f}^0)$ is a formula encoding the initial states $\mathcal{I}$;
- $G(\underline{f}^k)$ is an k-indexed formula encoding the goal states represented by $\mathcal{G}$;
- $T_i(\underline{f}^i, \underline{\ell}^i, \underline{f}^{i+1})$ is a formula encoding the transition relation between states reachable in i steps and states reachable in $i+1$ steps.

The main difference between our encoding techniques and those employed in other bounded model-checkers (e.g. NuSMV [8]) is in the way the formula that encodes the transition relation, i.e. $T_i(\underline{f}^i, \underline{\ell}^i, \underline{f}^{i+1})$, is generated.

By using the linear encoding technique, each component of Φ^k_Ξ is defined as follow:

- $I(\underline{f}^0)$ is a conjunction of the formulae f^0 if $f \in \mathcal{I}$ and $\neg f^0$ if $f \notin \mathcal{I}$;
- $G(\underline{f}^n)$ is obtained from $\mathcal{G}$ by replacing each fact f with f^n;
- $T_i(\underline{f}^i, \underline{\ell}^i, \underline{f}^{i+1})$ is equivalent to $T(\underline{f}^i, \underline{\ell}^i, \underline{f}^{i+1})$ (i.e. the formula encoding the transition relation is independent from the time step) and is a conjunction of the *Universal Formulae*, *Explanatory Frame Formulae*, *Axioms Formulae*, and *Conflict Exclusion Formulae*.

In order to improve readability, let us define $\tilde{\mathcal{R}}$ and $\tilde{\mathcal{A}}$ to be the sets of rule instances and axiom instances, respectively. Clearly, $\tilde{\mathcal{R}} = \{L\sigma \xrightarrow{\ell\sigma} R\sigma \mid L \xrightarrow{\ell} R \in \mathcal{R}, \sigma \text{ is a substitution}\}$ and $\tilde{\mathcal{A}} = \{p_1\sigma \wedge \cdots \wedge p_j\sigma \supset c\sigma \mid p_1 \wedge \cdots \wedge p_j \supset c \in \mathcal{A}, \sigma \text{ is a substitution}\}$.

Universal Formulae. They express how the transition relation evolves and

[12] Let $\underline{p}$ be a vector of facts or rules and i be an index in the appropriate range, then $\underline{p}^i$ is the corresponding time-indexed vector of propositional variable.

they are defined as the conjunction of the following:

$$\ell^i \supset \bigwedge\{f^i \mid f \in L\}$$
$$\ell^i \supset \bigwedge\{f^{i+1} \mid f \in (R \setminus L)\}$$
$$\ell^i \supset \bigwedge\{\neg f^{i+1} \mid f \in (L \setminus R)\}$$

for each $L \xrightarrow{\ell} R \in \tilde{\mathcal{R}}$.

Explanatory Frame Formulae. They express the inertia of the transition system. By introducing the axioms, the definition of such formulae require to define the set $\hat{\mathcal{A}}$ of the *contraposed* axioms of $\mathcal{A}$. For instance, $\neg b \supset \neg a$ is the contraposed axiom of $a \supset b$. Given the set of axiom instances $\tilde{\mathcal{A}}$, the set of contraposed axiom instances $\hat{\mathcal{A}}$ is $\{\neg c \wedge p_1 \wedge \cdots \wedge p_{h-1} \wedge p_{h+1} \wedge \cdots \wedge p_j \supset \neg p_h \mid (p_1 \wedge \cdots \wedge p_j \supset c) \in \tilde{\mathcal{A}}, h = 1, \ldots, j\}$. The explanatory frame formulae are thus defined as the conjunction of the following:

$$(f^i \wedge \neg f^{i+1}) \supset \Bigg(\bigvee \Big\{ \ell^i \mid (L \xrightarrow{\ell} R) \in \tilde{\mathcal{R}}, f \in (L \setminus R) \Big\} \vee \bigvee \{ \neg p_1^{i+1} \wedge p_2^{i+1} \wedge \cdots \wedge p_j^{i+1} \mid (\neg p_1 \wedge p_2 \wedge \cdots \wedge p_j \supset \neg f) \in \hat{\mathcal{A}} \} \Bigg)$$

$$(\neg f^i \wedge f^{i+1}) \supset \Bigg(\bigvee \Big\{ \ell^i \mid (L \xrightarrow{\ell} R) \in \tilde{\mathcal{R}}, f \in (R \setminus L) \Big\} \vee \bigvee \Big\{ p_1^{i+1} \wedge \cdots \wedge p_j^{i+1} \mid (p_1 \wedge \cdots \wedge p_j \supset f) \in \tilde{\mathcal{A}} \Big\} \Bigg)$$

for all facts $f \in \mathcal{F}$. In order to translate the above formulae into CNF without incurring in a combinatorial explosion in the number of clauses it suffices to replace each conjunction $p_1 \wedge \cdots \wedge p_j$ with a new variable v and to add the formula $v^i \equiv (p_1^i \wedge \cdots \wedge p_j^i)$. It is immediate to see that in the worst case the number of these new variables is in $O(|\tilde{\mathcal{A}}|)$.

Axioms Formulae. They express properties between the facts of the transition system. They are defined as the conjunction of $(p_1^i \wedge \cdots \wedge p_j^i) \supset c^i$ for each $(p_1 \wedge \cdots \wedge p_j \supset c) \in \tilde{\mathcal{A}}$.

Conflict Exclusion Formulae. They state what pair of rule instances cannot be executed in parallel for guaranteeing the interleaving semantic of the model. They are defined as the conjunction of $\neg(\ell_1^i \wedge \ell_2^i)$ for all $\ell_1 \neq \ell_2$ such that $\ell_1 \oplus \ell_2$.

It is immediate to see that the number of atoms in Φ^k_Ξ is in $O(k|\mathcal{F}| + k|\tilde{\mathcal{R}}| + k|\tilde{\mathcal{A}}|)$. Moreover the number of clauses generated by the Universal Formulae is in $O(kP_0|\tilde{\mathcal{R}}|)$ where P_0 is the maximal number of facts mentioned in an rule instance (usually a small number); the number of clauses generated by the Explanatory Frame Formulae is in $O(k|\mathcal{F}| + kR_0|\tilde{\mathcal{A}}|)$ where R_0 is the maximal number of facts mentioned in a precondition of an axiom instance (usually a small number); finally, the number of clauses generated by the Conflict Exclusion Formulae is in $O(k|\tilde{\mathcal{R}}|^2)$.

5 Implementation and Experimental Results

We have implemented the above ideas in SATMC, a SAT-based Model-Checker for security protocol analysis. Given a protocol insecurity problem with axioms Ξ, SATMC compiles it into a SAT formula Φ^k_Ξ using one of its encoding techniques for increasing values of k and the propositional formula generated at each step is fed to a state-of-the-art SAT solver (Chaff, SIM, and SATO are currently supported). As soon as a satisfiable formula is found, the corresponding model is translated back into a partial order attack which is reported to the user. Two encoding techniques are currently implemented in SATMC and both have been extended for supporting axioms: the first belongs to the family of so-called linear encodings, the second is the more sophisticated graphplan-based encoding.

We have run our tool against a selection of (flawed) security protocols drawn from the Clark/Jacob library [9]. For each protocol we have automatically generated, by means of a translator from IF [1] into the SATMC internal language, two corresponding protocol insecurity problems modeling a scenario with a bounded number of sessions in which the involved principals exchange messages on a channel controlled, respectively, by the Dolev-Yao intruder and by the optimized Dolev-Yao intruder. As explained in Section 2 the optimization consists in modeling the intruder ability of decomposing messages by means of axioms instead of rules.

Tables 1 and 2 report the results of our experiments obtained by applying the linear encoding and the graphplan-based encoding, respectively, on the protocol insecurity problem without the optimized intruder (**DY**) and with it (**Optimized DY**). Experiments have been carried out on a PC with a 1.4 GHz CPU and 1 GB of RAM. For each protocol and for each protocol insecurity problem with and without the optimized intruder we give the smallest value of k at which the attack is found (**K**), and the number of propositional variables (**Atoms**) and clauses (**Clauses**) in the SAT formula. [13] It is immediate to see that for both the applied encoding techniques, SATMC is able to find out up to

[13] Notice that, if the length of the attack on the protocol insecurity problems with and without the optimized intruder is the same, the results obtained are so comparable that we avoid to show them.

Table 1
Experimental data using the linear encoding

	DY			Optimized DY		
Protocol	**K**	**Atoms**	**Clauses**	**K**	**Atoms**	**Clauses**
KaoChow 2	9	530,726	1,804,005	7	414,536	1,489,121
KaoChow 3	9	995,323	5,736,662	7	776,805	4,590,268
NSCK	9	114,530	334,086	8	88,343	298,491
NSPK	7	6,612	33,326	4	3,714	19,242
NSPK-server	8	9,157	53,741	5	5,600	33,835
Woo-Lam M	6	481,394	2,498,382	5	409,114	2,133,265

Table 2
Experimental data using the Graphplan-based encoding

	DY			Optimized DY		
Protocol	**K**	**Atoms**	**Clauses**	**K**	**Atoms**	**Clauses**
KaoChow 2	9	726	3,065	7	458	1,784
KaoChow 3	9	990	5,019	7	587	2,606
NSCK	9	435	1,392	8	348	1,105
NSPK	7	411	1,249	4	199	549
NSPK-server	8	847	2,688	5	380	1,177
Woo-Lam M	6	481	1,518	5	358	1,137

40% shorter attacks [14] by generating up to 50% smaller propositional formulae. These results confirm the effectiveness of the proposed optimized intruder model for the SAT-based model-checking approach and pave the way to the application of SATMC to protocols of industrial complexity where attacks can eventually require a considerable number of intruder knowledge inference steps.

6 Conclusions and Perspectives

We have proposed an optimized intruder model for SAT-based model-checking of security protocols. We have shown that this optimization, based on the idea of modeling the decomposition of the intruder knowledge by means of axioms instead of rules, leads to the discovering of shorter attacks by generating smaller propositional formulae. We have enhanced our SAT-based

[14] Notice that for each security protocol analyzed, the attacks found on the corresponding protocol insecurity problem with and without the optimized intruder are identical except for the fact that in the case of the former we save all those rules executed for decomposing the knowledge of the intruder.

model-checker (SATMC) to be able to analyse protocol insecurity problems extended with a set of axioms (without equivalence cycles). Experimental results obtained by running SATMC against a selection of security protocols drawn from the Clark-Jacob's library confirm the effectiveness of the proposed optimized intruder model and pave the way to its application to large protocols which arise in practical applications. As a matter of fact, messages exchanged in industrial-scale security protocols can have a complex structure and, therefore, attacks on such protocols can require a considerable number of intruder knowledge manipulations. Since the proposed optimization allows for decomposing the whole intruder knowledge instantaneously, we conjecture that it will be particularly successful on such protocols and experiments will be done in this direction.

Another interesting and challenging future direction that we would like to investigate concerns the extension of our SAT-reduction techniques for supporting the encoding of generic set of axioms also specifying equivalences between facts. This would be particularly useful to express algebraic equations that specify, for instance, special properties of cryptographic operators such as exponentiation in the Diffie-Hellman protocol [10]. It is worth pointing out that on such a protocol we have already obtained some preliminary results in our current setting. In fact, by partially specifying exponentiation by means of a set of axioms, we have been able to discover the well-known attack on the Diffie-Hellman protocol.

Acknowledgments

We are grateful to Cristina Frà for her contribution to the implementation of the encodings for supporting such an optimized intruder. Moreover we wish to thank the anonymous reviewers for their useful comments. This work was partially funded by EU Calculemus Training Network (HPRN-CT-2000-00102) and by FET Open EC Project "AVISPA: Automated Validation of Internet Security Protocols and Applications" (IST-2001-39252).

References

[1] 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 *Proceedings of CAV'02*, LNCS 2404, pages 349–354. Springer-Verlag, 2002.

[2] A. Armando and L. Compagna. Automatic SAT-Compilation of Protocol Insecurity Problems via Reduction to Planning. In *Proceedings of FORTE 2002*, LNCS 2529, pages 210–225. Springer-Verlag, 2002.

[3] A. Armando and L. Compagna. Abstraction-driven SAT-based Analysis of Security Protocols. In *Proceedings of SAT 2003*, LNCS 2919. Springer-Verlag, 2003. Available at http://www.avispa-project.org.

[4] A. Armando, L. Compagna, and P. Ganty. SAT-based Model-Checking of Security Protocols using Planning Graph Analysis. In *Proceedings of FME'2003*, LNCS 2805. Springer-Verlag, 2003.

[5] 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 http://www.avispa-project.org.

[6] A. Biere, A. Cimatti, E. Clarke, and Y. Zhu. Symbolic Model Checking without BDDs. In W. R. Cleaveland, editor, *Proceedings of TACAS'99*, LNCS 1579, pages 193–207, Berlin, 1999. Springer-Verlag.

[7] I. Cervesato, N. A. Durgin, P. Lincoln, J. C. Mitchell, and A. Scedrov. A meta-notation for protocol analysis. In *CSFW*, pages 55–69, 1999.

[8] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri. NuSMV: A New Symbolic Model Verifier. *LNCS 1633*, 1999.

[9] J. Clark and J. Jacob. A Survey of Authentication Protocol Literature: Version 1.0, 17 November 1997. Available at http://www.cs.york.ac.uk/~jac/papers/drareview.ps.gz.

[10] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

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

[12] N. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov. Undecidability of Bounded Security Protocols. In *Proceedings of the FLOC'99 Workshop on Formal Methods and Security Protocols (FMSP'99)*, 1999.

[13] S. Even and O. Goldreich. On the security of multi-party ping pong protocols. In *Proceedings of 24th IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society, 1983.

[14] J. Heather, G. Lowe, and S. Schneider. How to prevent type flaw attacks on security protocols. In *Proceedings of The 13th Computer Security Foundations Workshop (CSFW'00)*. IEEE Computer Society Press, 2000.

[15] F. Jacquemard, M. Rusinowitch, and L. Vigneron. Compiling and Verifying Security Protocols. In M. Parigot and A. Voronkov, editors, *Proceedings of LPAR 2000*, LNCS 1955, pages 131–160. Springer-Verlag, 2000.

[16] R. M. Needham and M. D. Schroeder. Using Encryption for Authentication in Large Networks of Computers. Technical Report CSL-78-4, Xerox Palo Alto Research Center, Palo Alto, CA, USA, 1978. Reprinted June 1982.