A Decision Procedure for the Verification of Security Protocols with Explicit Destructors*

Extended Abstract †

Stéphanie Delaune
France Télécom R&D
LSV, CNRS UMR 8643, ENS de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex, France
stephanie.delaune@lsv.ens-cachan.fr

Florent Jacquemard
INRIA, Research Unit Futurs, project SECSI
LSV, CNRS UMR 8643, ENS de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex, France
florent.jacquemard@inria.fr

ABSTRACT

We present a non-deterministic polynomial time procedure to decide the problem of insecurity, in the presence of a bounded number of sessions, for cryptographic protocols containing explicit destructor symbols, like decryption and projection. These operators are axiomatized by an arbitrary convergent rewrite system satisfying some syntactic restrictions. This approach, with parameterized semantics, allows us to weaken the security hypotheses for verification, *i.e.* to address a larger class of attacks than for models based on free algebra. Our procedure is defined by an inference system based on basic narrowing techniques for deciding satisfiability of combinations of first-order equations and intruder deduction constraints.

Categories and Subject Descriptors: C.2.2 Network Protocols: Protocol verification

General Terms: Security, Theory, Verification.

Keyword: Security Protocols, Formal Methods, Constraint Solving.

1. INTRODUCTION

Security protocols are paramount in today's secure transactions through public channels. It is therefore essential to obtain through formal proofs as much confidence as possible in their correctness. Many works have been devoted to the use of formal methods in order to automate the proof of existence of logical attacks on such protocols.

*This work has been partly supported by the RNTL project PROUVÉ 03V360 and the ACI-SI Rossignol.

†A full version of this paper is available as the LSV Research Report LSV-04-8 at `http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/rr-lsv-2004-8.rr.ps`

CCS'04, October 25-29, 2004, Washington, DC, USA.

This problem is undecidable in general, and the undecidability results from several factors: the ability of agents to generate fresh random data (nonces), the unlimited size of terms, the unboundedness of the number of sessions. Removing the last condition is however sufficient for decidability (while removing the others is not, see [15, 8, 1]), and several decision procedures (at least NP-complete) have been proposed (under this condition) for different models of attackers [4, 23, 11, 9, 6, 7, 24, 27, 12, 26]. In these approaches, the cryptographic operations like encryption, signature, application of one-way functions *etc* are abstracted into function symbols and the messages are represented by logical terms rather than bit-strings. Logical attacks can be characterized by sequences of abstract messages exchanged by honest agents executing the protocol and a malicious agent (called the intruder), and searching for such attacks can be reduced to solving systems of symbolic constraints [4, 23, 9]. Most of the former decision procedures are based on a symbolic constraint reduction system (*i.e.* a set of inference rules) which strongly depends on the capabilities of the intruder to analyze messages, and are therefore restricted to some particular intruder model.

Moreover, in all the approaches cited above the messages cannot contain symbols representing destructors like decryption or projection, for which we would have simplification rules of the form $d(\{x\}_y, y) \to x$. It means that in these approaches, decryption of a message with a key succeeds only if the message was encrypted with the corresponding key. From a computational point of view, a decryption procedure satisfying such an assumption needs some kind of integrity checking [5]. From a formal point of view, it has been also noticed in [22] that the absence of a decryption operator masks some possible attacks. Note also that some analysis tools, such as the NRL analyzer described in [21], are capable of analyzing protocols that employ decryption explicitly.

In this paper, we present a non-deterministic polynomial time procedure to decide the problem of insecurity in presence of a bounded number of sessions, for cryptographic protocols containing explicit destructor symbols. We propose a reduction of this problem to solving sets of equations and other symbolic constraints called *intruder constraints*. We provide a generic narrowing based inference system for the resolution of such sets of constraints modulo a convergent rewriting system which defines the semantics of destructor symbols (including in particular cryptographic primitives for encryption and decryption).

The advantages of this approach are twofold. On one hand, we have a generic decision procedure which can be applied to any model which can be axiomatized by rewriting systems in our class. Modeling the properties of cryptographic operators (and hence the capabilities of an intruder to analyze messages) with equational systems was already the approach of [16] which is often cited as the pioneer paper in the domain of formal verification of cryptographic protocols. It is also the base of more recent languages for formal protocol description, like [3]. This approach has also been investigated later in e.g. [25, 17] for procedures of abstract interpretation based on tree automata techniques – hence, not for decision. The class of rewriting systems which are in the scope of our results contains some relevant theories such as the standard theory of [16] extended by the fact that decryption and projections are explicit and the theory of involution which is mentioned in [26]. Moreover, the usage of our constraint solving procedure, is not limited to the verification of cryptographic protocols, though the restrictions were tailored for this application.

On the other hand, our framework allows us to specify protocols in a language which improves most of those used in the procedures cited above, both in readability and expressiveness. First, since we are able to deal with first-order equations, we can add some equations in protocol specifications, like in [11], in order to specify explicitly some tests performed by the participants at some stage of the protocol. Second, some destruction operators such as decryption or projections can be defined by the rewriting system, and these operators may be used in the protocol specifications, in order to specify unambiguously the actions taken by the agents in protocol execution. For instance, if a protocol specifies that an agent A who knows a symmetric key K shall receive a ciphertext $\{N\}_K$ (number N encrypted with K), and answer N, it is often implicitly assumed that A must check whether this message is indeed a ciphertext and that it is really encrypted with K before trying to decipher it and posting the result. In our settings, we can specify such a protocol in a more general way: A, upon receiving some message X, replies with $d(X, K)$. If X has the form $\{N\}_K$, then A's reply will be simplified (to N) in the network thanks to the rewrite rule $d(\{x\}_y, y) \to x$ for the definition of the decryption operator d. This relaxes the above implicit hypotheses concerning the verifications of X by A, and hence enables more attacks.

Related Works

Modeling the behavior of a cryptosystem in terms of rewrite rules is more expressive than the standard approach which consist in modeling cryptosystems in terms of free algebras. Some recent works [22, 19] compare the both approaches, for the case of the decryption operator, and give syntactic conditions on protocols under which security in the free algebra implies security in the rewrite rules model. The main condition, called *EV-freeness*, expresses that a principal should not apply an encryption operator to a term unless it has been able to verify that that term has some kind of structure. It is shown in [22, 19] that for protocols which verifying the conditions, using of an explicit decryption operator does not enable any new attacks. Hence, in this case, formal cryptographic protocol analysis can be made in the convenient free algebra model. However, although the *EV-freeness* condition is generally considered as a good practice to design cryptographic protocols, it cannot be assumed in any cases, as we shall see below with the analysis of the Denning-Sacco key distribution protocol presented in Section 2. Note also that *EV-freeness* is related to the problem of integrity checking mentioned above. In this paper, we show that the verification of protocol insecurity in models with rewrite rules for explicit destructors has the same theoretical complexity as in free algebras models.

In [10], the authors prove the decidability of the deducibility by intruder for a class of equational theories. However this class is incomparable with ours. Indeed, for example they allow the homomorphism property but not the idempotence property. In [2], it is shown that the problems of deducibility and indistinguishability (static equivalence) are both decidable in PTIME in a model with explicit destructors and equational theories slightly more general than those considered here. Note that these two works are limited to a passive attacker, who can only listen to messages (*i.e.* the procedures only deals with constraints without variables), whereas we treat both cases of passive (PTIME decision procedure) and active attackers, who can send messages to the network (we solve systems of constraints with variables).

Chevalier et *al.* present a framework which is quite general in the sense that different intruder's deduction capabilities can be captured by the concept of the so-called *oracle rules*. Oracle rules are deduction rules that satisfy certain high-level conditions which allow to bound the length of derivations and substitutions of an attack and also allow to replace a subterm composed by the intruder by a smaller message. We retrieve some similar conditions in the proof of completeness of our decision procedure. However, we do not know whether the deduction relation studied in this paper could be defined with oracle rules.

After some motivating examples (Section 2) and preliminary definitions of our framework (Section 3), we show first how to convert the insecurity problem into a constraint solving problem (Section 4). Then, we investigate in Section 5 the verification of ground constraints with a locality lemma from which it follows that this problem can be decided in polynomial time. Finally, we introduce our inference system for constraint solving (Section 6) and prove its correctness, completeness and termination, and show that it provides a non-deterministic polynomial algorithm for the decision of constraint satisfiability.

2. MOTIVATIONS

Consider the following protocol for a symmetric key exchange in an asymmetric cryptosystem. This is a simplification of the Denning-Sacco key distribution protocol [13], omitting certificates and timestamps.

$$\begin{array}{llll} 0. & A & \to & B: & \langle A, \{\{K_{ab}\}^a_{pub(A)^{-1}}\}^a_{pub(B)} \rangle \\ 1. & B & \to & A: & \{secret\}^s_{K_{ab}} \end{array}$$

In the first message, the agent A sends to B a freshly chosen symmetric key K_{ab} for further secure communications. This key is encrypted using an asymmetric encryption function (denoted by $\{\_\}^a_\_$) and the secret key of A, $pub(A)^{-1}$. The result of this encryption is later encrypted with B's public key $pub(B)$ so that only B shall be able to learn K_{ab}. Moreover, A appends its name at the beginning of the message (using the pairing function denoted $\langle \_, \_ \rangle$) so that the receiver B knows which public key to use in order to obtain K_{ab}. Then, B can extract the symmetric key K_{ab} and use it to encrypt (with a symmetric encryption function denoted $\{\_\}^s_\_$) a secret code *secret* he wants to communicate to A (message 1).

It is well-known that the above common syntax used to describe cryptographic protocols is ambiguous, and the procedures for formal verification of protocols are usually rather based on specifications as sequences of programs, one for each agent. In our running example, the program of B can be specified as follows:

$$B\text{'s role:} \quad \mathsf{recv}\big(\langle x_A, \{\{x_{K_{ab}}\}^a_{pub(x_A)^{-1}}\}^a_{pub(x_B)}\rangle\big); \quad \mathsf{send}\big(\{secret\}^s_{K_{ab}}\big) \tag{1}$$

This version of the Denning-Sacco protocol is flawed: there exists an attack involving two sessions of the protocol and an intruder. In the first session, an honest and naive agent a playing A's role initiates voluntarily a communication with the intruder (without knowing he is an intruder). The intruder thus learns $a, \{\{K_{ab}\}^a_{pub(a)^{-1}}\}^a_{pub(I)}$, where $pub(I)$ is the intruder's public key. Hence, the intruder is able to extract the signed key $\{K_{ab}\}^a_{pub(a)^{-1}}$ and the key K_{ab} itself (we assume that he knows the public key of a). Thereafter, the intruder can fool an honest agent b playing B's role (in another session) by sending him $a, \{\{K_{ab}\}^a_{pub(a)^{-1}}\}^a_{pub(b)}$, which makes b believe that he has received a symmetric key K_{ab} from a. The secret in b's answer is thus not secure, because the intruder knows K_{ab}.

As noted in the introduction, in the above program (1), we implicitly assume that the agent B checks that the second component of the received message is a ciphertext, with an encryption with the private key of x_A (the first component of the received tuple) and an encryption with his public key (the value of the variable x_B is the name of the agent B in the above program). We may want to specify a more lax agent B which is not capable of such a check, and blindly applies the decryption algorithm twice to any received message. Such an agent B can be specified by the following program, which makes use of an asymmetric decryption function (denoted $ad(\_, \_)$) and left- and right-projection operators (resp. $\pi_1(\_)$ and $\pi_2(\_)$):

$$B\text{'s role:} \quad \mathsf{recv}(x); \quad \mathsf{send}\big(\{secret\}^s_{ad(ad(\pi_2(x), pub(x_B)^{-1}), pub(\pi_1(x)))}\big) \tag{2}$$

The answer of B in the above program shall be simplified by rewrite rules defining ad and π_1, π_2 presented later in Section 3.2. There are no ambiguities or implicit checks in program (2) and its verification is performed under security properties which are strictly more general (weaker) than for program (1). Indeed, there exists an attack of program (2) involving only one session, where the intruder does not need to wait for an honest agent to initiate a communication with him.

Moreover, we can also use equations in programs to express explicitly some checks performed by the agent B. Consider for instance a patched version of the above Denning-Sacco protocol:

$$\begin{array}{llll} 0. & A \to B: & A, \{\{\langle\langle A, B\rangle, K_{ab}\rangle\}^a_{pub(A)^{-1}}\}^a_{pub(B)} \\ 1. & B \to A: & \{secret\}^s_{K_{ab}} \end{array}$$

Some redundancy has been added on purpose in the first message in order to prevent the above first attack. In our setting, the program for B's role can be specified as follows:

$$\begin{array}{l} \mathsf{recv}(x); \\ x_B = \pi_2(\pi_1(ad(ad(\pi_2(x), pub(x_B)^{-1}), pub(\pi_1(x))))); \\ \pi_1(x) = \pi_1(\pi_1(ad(ad(\pi_2(x), pub(x_B)^{-1}), pub(\pi_1(x))))); \\ \mathsf{send}\big(\{secret\}_{ad(ad(\pi_2(x), pub(x_B)^{-1}), pub(\pi_1(x)))}\big) \end{array}$$

With the first equation, B verifies whether he finds his name x_B at the second position of the ciphertext, and with the second equation he checks whether both occurrences of the name of agent A (before and inside the ciphertext) are the same.

The use of explicit destructors and equations allows also to address a broader class of protocols than the ones described in the standard role's model. For instance, the following protocol (see [28]) can not be expressed in the standard role's model.

$$\begin{array}{llll} 0. & A \to B: & \{\langle M, B\rangle\}^s_K \\ 1. & B \to A: & B \\ 2. & A \to B: & K \\ 3. & B \to A: & M \end{array}$$

The message $\{\langle M, B\rangle\}_K$ is seen as a variable x by the agent B who does not know the decryption key K, and one can not express that x must be decomposed after the reception of K in message 2 without the explicit use of a function symbol for symmetric decryption sd. In our approach B's role can be specified as follows:

$$B\text{'s role} \quad \begin{array}{l} \mathsf{recv}(x); \mathsf{send}(x_B); \\ \mathsf{recv}(y); \pi_2(sd(x, y)) = x_B; \mathsf{send}(\pi_1(sd(x, y))) \end{array}$$

3. PRELIMINARIES

We now introduce some notations and basic definitions for terms and term rewriting systems (the reader may refer to [14] for a comprehensive survey on term rewriting systems), and then proceed with the definition of the so-called intruder constraints.

3.1 Terms, Substitutions

We assume given a signature $\mathcal{F}$ and an infinite set of variables $\mathcal{X}$. The set $\mathcal{F}$ is partitioned into a subset $\mathcal{PF}$ of *private* functions symbols, and a subset $\mathcal{VF}$ of *visible* or *public* functions symbols. The set of terms built with $\mathcal{F}$ and $\mathcal{X}$ is denoted $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and its subset of *ground* terms (terms without variables) $\mathcal{T}(\mathcal{F})$. We note $vars(t)$ the set of variables occurring in a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, and $head(t)$ the root symbol of t. The *positions* in a term t are represented as sequence of positive integers (Λ denotes the empty sequence) and are denoted by $Pos(t)$. If $p \in Pos(t)$, the subterm of t at position p, denoted $t|_p$, is defined recursively by: $t|_\Lambda = t$ and $f(t_1, \ldots, t_n)|_{ip} = t_i|_p$ if $1 \leq i \leq n$ and ip is the concatenation of i at the beginning of the sequence p. The term obtained by replacing $t|_p$ by the term s is denoted $t[s]_p$. We note $st(t)$ the set of subterms of t and $sst(t) = st(t) \setminus \{t\}$ the set of strict subterms of t. These notations are extended as expected to sets of terms and term rewriting systems.

A *substitution* is the term morphism extension of a finite mapping $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ where $x_1, \ldots, x_n \in \mathcal{X}$ and

$t_1, \ldots t_n \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. If $t_1, \ldots t_n \in \mathcal{T}(\mathcal{F})$, the substitution is called *ground*. As usual, the application of a substitution σ to a term t and the composition of substitutions σ_1 by σ_2 are written in postfix notation, respectively $t\sigma$ and $\sigma_1\sigma_2$. A substitution σ is *grounding for* t if $t\sigma \in \mathcal{T}(\mathcal{F})$. Given two terms u and v the *replacement* of u by v, denoted by $[u \mapsto v]$, maps every term t to the term $t[u \mapsto v]$ which is obtained by replacing all occurrences of u in t by v. Note that the result of such replacement is uniquely determined.

In the paper, $|S|$ denotes the cardinal of the set S. The *size* $\|t\|$ of a term t is the number of positions in t. For convenience we extend this notation to a set of terms T as the sum of the size of each term in T. The *dag-size* $\|T\|_d$ of a set of terms T is the number of distinct subterms of T (*i.e.* it is the number of nodes in a representation of T as a dag with maximal sharing). More details about the dag representations of terms can be found in [26].

3.2 Term Rewriting Systems

A *term rewriting system* (TRS) is a finite set of *rewrite rules* $l \to r$ where $l \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and $r \in \mathcal{T}(\mathcal{F}, vars(l))$. A term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ rewrites to s by a TRS $\mathcal{R}$, denoted $t \to_{\mathcal{R}} s$ if there is a rewrite rule $l \to r$ in $\mathcal{R}$, a position p of t and a substitution σ such that $t|_p = l\sigma$ and $s = t[r\sigma]_p$. If $p = \Lambda$, we write $t \xrightarrow{\Lambda}_{\mathcal{R}} s$. We write $\xrightarrow{*}_{\mathcal{R}}$ for the reflexive and transitive closure of $\to_{\mathcal{R}}$ and $\xleftrightarrow{*}_{\mathcal{R}}$ for its reflexive, transitive and symmetric closure. A $\mathcal{R}$-*unifier* of two terms $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ (also called $\mathcal{R}$-*solution* of the *equation* $s = t$) is a substitution σ such that $s\sigma \xleftrightarrow{*}_{\mathcal{R}} t\sigma$. If $\mathcal{R} = \emptyset$, we simply call σ a unifier. It is well-known that unifiable terms have a *most general unifier* (*mgu*), *i.e.* a substitution σ such that $\sigma \leq \tau$ (there exists ρ such that $\sigma\rho = \tau$) for every other unifier τ of s and t.

A TRS $\mathcal{R}$ is *terminating* if there are no infinite chains $t_1 \to_{\mathcal{R}} t_2 \to_{\mathcal{R}} \ldots$, *confluent* if for all t_0, t_1, t_2 such that $t_1 \xleftarrow{*}_{\mathcal{R}} t_0 \xrightarrow{*}_{\mathcal{R}} t_2$, there exists t_3 such that $t_1 \xrightarrow{*}_{\mathcal{R}} t_3 \xleftarrow{*}_{\mathcal{R}} t_2$, and *convergent* if it is both terminating and confluent. A term t is in $\mathcal{R}$-*normal form* if there is no term s with $t \to_{\mathcal{R}} s$ and the set of $\mathcal{R}$-normal forms is denoted $NF_{\mathcal{R}}$. If $t \xrightarrow{*}_{\mathcal{R}} s$ and $s \in NF_{\mathcal{R}}$ then we say that s is a $\mathcal{R}$-normal form of t, and write $s = t\downarrow_{\mathcal{R}}$. The application of the operator $\downarrow_{\mathcal{R}}$ is extended to set of terms as expected. A substitution σ is called $\mathcal{R}$-normal if for every variable $x \in dom(\sigma)$, $x\sigma \in NF_{\mathcal{R}}$.

Definition 1. *A TRS $\mathcal{R}$ is called* public-collapsing *if every rule $\ell \to r \in \mathcal{R}$ satisfies the two following conditions:*

1. *$r \in vars(\ell)$ or $r \in NF_{\mathcal{R}} \cap \mathcal{T}(\mathcal{VF})$ and $r \neq \ell$,*
2. *if $\ell = f(l_1, \ldots, l_n)$ with $f \in \mathcal{VF}$, then for all strict subterms of ℓ of the form $g(t_1, \ldots, t_m)$ with $g \in \mathcal{VF}$, either $g(t_1, \ldots, t_m) \in NF_{\mathcal{R}} \cap \mathcal{T}(\mathcal{VF})$, or there exists $j \leq m$ such that $t_j = r$.*

Now, we are going to illustrate this definition by giving several equational theories, relevant to cryptographic protocols verification, which fall in the class of convergent public-collapsing TRS (see also [3]). Let $\mathcal{VF} = \{\{\_\}^s_\_, sd(\_,\_), \{\_\}^a_\_, ad(\_,\_), \langle\_,\_\rangle, \pi_1(\_), \pi_2(\_), pub(\_)\}$, and $\mathcal{PF} = \{\_^{-1}\}$. The meaning of these functions is described in Section 2.

Dolev-Yao theory. The following TRS corresponds to the theory of [16] for public key encryption. This theory has been studied in many works but, as noted in Section 2, the use of explicit decryption and projections symbols and equations in protocol specifications permits to generalize other approaches.

$$sd(\{x\}^s_y, y) \to x, \qquad ad(\{x\}^a_y, y^{-1}) \to x,$$
$$x^{-1^{-1}} \to x, \qquad ad(\{x\}^a_{y^{-1}}, y) \to x,$$
$$\pi_i(\langle x_1, x_2\rangle) \to x_i \ (i = 1, 2)$$

Inverse-key theory. The three following rules extend the Dolev-Yao theory:
$\{sd(x,y)\}^a_y \to x$, $\{ad(x,y)\}^a_{y^{-1}} \to x$, $\{ad(x, y^{-1})\}^a_y \to x$. They are useful when we assume that decryption is just an encryption with the inverse key like for the cryptosystem RSA.

Theory of involution. It is mentioned in [26] and can also be encoded by a convergent public-collapsing TRS by adding the following rules to the standard theory: $\{\{x\}^a_y\}^a_{y^{-1}} \to x$, $\{\{x\}^a_{y^{-1}}\}^a_y \to x$. This approach improves the model of [26] since we consider cases where the rules are applied everywhere in terms and not only at the top of messages.

Probabilistic encryption. We can consider function symbols $pe(\_,\_,\_)$ and $pd(\_,\_)$ for probabilistic encryption and decryption [18], and rules such as: $pd(pe(m,k,r),k) \to m$, where the function pe takes a message m, an encryption key k and a random input r.

The following trivial lemma shall be used later while reasoning on public-collapsing systems.

Lemma 1. *Let $\mathcal{R}$ be a public-collapsing TRS and let $s, s_1, \ldots, s_n \in \mathcal{T}(\mathcal{F})$ be in $\mathcal{R}$-normal form. We have $s = f(s_1, \ldots, s_n)\downarrow_R$ iff $s = f(s_1, \ldots, s_n)$ or $f(s_1, \ldots, s_n) \xrightarrow{\Lambda}_{\mathcal{R}} s$.*

3.3 Intruder Deductions and Constraints

We assume from now on given a convergent public-collapsing TRS $\mathcal{R}$. We assume given a linear well-founded ordering $\prec$ on $\mathcal{T}(\mathcal{F})$ and a special term denoted by 0 such that $0 \in NF_{\mathcal{R}}$ and is minimal w.r.t. $\prec$. We shall use a linear extension $\ll$ of $\prec$ to multisets of ground terms. We are studying below the saturation of sets of ground terms under the application of visible function symbols of $\mathcal{VF}$ and rewrite rules of $\mathcal{R}$ ($\mathcal{R}$ is supposed to define the semantics of the symbols of $\mathcal{F}$).

Given a set of ground terms $T \subseteq \mathcal{T}(\mathcal{F})$, the *intruder set* $\mathcal{I}_{\mathcal{R}}(T)$ is the smallest, w.r.t. inclusion, subset of $\mathcal{T}(\mathcal{F})$ containing T, closed under $\xleftrightarrow{*}_{\mathcal{R}}$, and such that for all $t_1, \ldots, t_n \in \mathcal{I}_{\mathcal{R}}(T)$ and all $f \in \mathcal{VF}$ of arity n, $f(t_1, \ldots, t_n) \in \mathcal{I}_{\mathcal{R}}(T)$. This aims, in the context of protocol verification, at modeling an intruder who is able to deduce messages from the ones collected on the insecure network.

An *intruder constraint* is a tuple written $t_1, \ldots, t_n \Vdash r$ where $t_1, \ldots, t_n, r \in \mathcal{T}(\mathcal{F}, \mathcal{X})$. The terms $t_1, \ldots, t_n$ are called the *hypotheses* of the intruder constraint and r is called its *target*. An intruder constraint is said to be *basic* when $r \in \mathcal{X}$. Since the order of the hypotheses does not matter, we shall sometimes write an intruder constraint $T \Vdash r$ where T is the finite set $\{t_1, \ldots, t_n\}$. A $\mathcal{R}$-*solution* of an intruder constraint $T \Vdash r$ is a grounding substitution σ such that $r\sigma \in \mathcal{I}_{\mathcal{R}}(T\sigma)$.

Remark. An intruder constraint $t_1, \ldots, t_n \Vdash r$ may be understood as a restricted kind of second-order equation $x(t_1, \ldots, t_n) = t$ where $t_1, \ldots, t_n, t$ are first order terms and x is a second order variable which can take its values in contexts made of public operators of $\mathcal{VF}$.

DEFINITION 2. *A finite set of intruder constraints $\mathcal{C}$ is well-formed if its elements can be ordered as $T_0 \Vdash r_0, \ldots, T_l \Vdash r_l$ such that the following conditions hold:*

1. $0 \in T_0$ *and* $st(\mathcal{R}) \cap NF_{\mathcal{R}} \cap \mathcal{T}(\mathcal{VF}) \subseteq T_0$,
2. *for all* $i < l$, $T_i \subseteq T_{i+1}$,
3. *for all* $i \leq l$, *for all* $x \in vars(T_i)$, *there exists* $j < i$ *such that* $x \in vars(r_j)$.

The definitions of constraints and solutions and the above restrictions have been validated by the application to the verification of protocols presented in Section 4. Intuitively, $T \Vdash r$ is true if, knowing all the terms in T, an intruder is able to construct r. The condition *1* imposes that some terms are in the hypotheses of all the intruder constraints. However it is not really a restriction since these terms, built with public symbols, can always be constructed by the intruder. Condition *2* captures the fact that the intruder never forgets information (every message read by the intruder is added to its knowledge) and Condition *3* says that every variable of $\mathcal{C}$ appears for the first time in the target of a constraint. Indeed, in our application in Section 4, every variable of $\mathcal{C}$ corresponds to a message received by an agent following the protocol, and the intruder must be able to send such a message.

The conditions of Definition 2 are invariant (under some conditions) under the application of a substitution and normalization with $\mathcal{R}$:

LEMMA 2. *Given a finite well-formed set of intruder constraints $\mathcal{C} = \{T_0 \Vdash r_0, \ldots, T_l \Vdash r_l\}$ and a substitution σ, $\mathcal{C}\sigma$ is well-formed and if moreover for each $i \leq l$, $r_i\sigma \in NF_{\mathcal{R}}$, then $\mathcal{C}\sigma\downarrow_{\mathcal{R}}$ is well-formed.*

Note that the hypothesis $r_i\sigma \in NF_{\mathcal{R}}$ is crucial. Indeed, let us consider for instance the well-formed $\mathcal{C} = \{T \Vdash sd(\{a\}^s_x, y); T, x \Vdash b\}$, and the substitution $\sigma = \{x \mapsto y\}$. The system $\mathcal{C}\sigma\downarrow_{\mathcal{R}} = \{T \Vdash a; T, x \Vdash b\}$ does not fulfill Condition *3* of Definition 2 but $sd(\{a\}^s_x, y)\sigma = sd(\{a\}^s_y, y) \notin NF_{\mathcal{R}}$.

3.4 Proof Trees

We find convenient for the proofs of the next sections to represent the intruder deductions leading to a term of $\mathcal{I}_{\mathcal{R}}(T)$ by a proof tree describing the deduction steps.

DEFINITION 3. *Given a finite set $T \subseteq \mathcal{T}(\mathcal{F})$ and $u \in \mathcal{T}(\mathcal{F})$,* a proof *$P$ of $T \vdash_{\mathcal{R}} u$ is a tree labeled by terms of $\mathcal{T}(\mathcal{F})$ such that:*

- *every leaf of P is labeled with $v\downarrow_{\mathcal{R}}$ for some $v \in T$,*
- *every internal node of P with n sons $P_1, \ldots, P_n$ whose roots are respectively labeled with $v_1, \ldots, v_n$ is labeled by $f(v_1, \ldots, v_n)\downarrow_{\mathcal{R}}$ for some $f \in \mathcal{VF}$,*
- *the root of P is labeled with $u\downarrow_{\mathcal{R}}$, this label is denoted $root(P)$.*

The size of a proof P is the number of its nodes.

Note that with this definition, every label of a proof is in $NF_{\mathcal{R}}$. A proof P of $T \vdash_{\mathcal{R}} u$ (not reduced to a leaf) is called a *composition proof* if its direct subtrees $P_1, \ldots, P_n$ are such that $root(P) = f(root(P_1), \ldots, root(P_n))$ for some $f \in \mathcal{VF}$. Otherwise, it is called a *decomposition proof* and, by Lemma 1, it means that there exists $f \in \mathcal{VF}$ such that $f(root(P_1), \ldots, root(P_n)) \xrightarrow{\Lambda}_{\mathcal{R}} root(P)$.

EXAMPLE 1. $T = \{\{m_1\}_k, k, m_2\}$, $\mathcal{R} = \{d(\{x\}_y, y) \to x\}$.

$$\frac{T \vdash_{\mathcal{R}} \{m_1\}_k \quad T \vdash_{\mathcal{R}} k}{T \vdash_{\mathcal{R}} d(\{m_1\}_k, k)\downarrow_{\mathcal{R}} = m_1}\,(d \in \mathcal{VF}) \quad \textit{is a decomposition proof.}$$

$$\frac{T \vdash_{\mathcal{R}} m_2 \quad T \vdash_{\mathcal{R}} k}{T \vdash_{\mathcal{R}} d(m_2, k)}\,(d \in \mathcal{VF}) \quad \textit{is a composition proof.}$$

LEMMA 3. *Given a finite set $T \subseteq \mathcal{T}(\mathcal{F})$ and $u \in \mathcal{T}(\mathcal{F})$, $u \in \mathcal{I}_{\mathcal{R}}(T)$ iff there exists a proof of $T \vdash_{\mathcal{R}} u$.*

4. VERIFICATION OF CRYPTOGRAPHIC PROTOCOLS

In this section, we show how the problem of insecurity of cryptographic protocols, assuming a bounded number of sessions, can be reduced to solving systems of intruder constraints and equations. An effective solving procedure is presented in the next sections. We shall describe first our model for cryptographic protocols and their execution (Section 4.1), second the security properties that we shall consider (Section 4.2), and then the construction of a constraint system given a protocol and a security problem (Section 4.3).

4.1 Protocol Semantics

We consider a simple representation of cryptographic protocols and their execution by agents which should fit with most of the formalisms in use.

A *protocol* is a finite set of programs, each program being a finite sequence of *instructions* of the form $\mathsf{recv}(x); \mathcal{E}; \mathsf{send}(s)$ with $x \in \mathcal{X}$, $s \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ and $\mathcal{E}$ is a set (possibly empty) of equations on terms of $\mathcal{T}(\mathcal{F}, \mathcal{X})$.

EXAMPLE 2. *The first version of the Denning-Sacco protocol described in Section 2 is made of two programs:*

$$\begin{array}{ll} A\text{'s role}: & \mathsf{recv}(x^0_0); x^0_0 = 0; \\ & \mathsf{send}\big(\langle x^0_A, \{\{x^0_{Kab}\}^a_{pub(x^0_A)^{-1}}\}^a_{pub(x^0_B)}\rangle\big); \\ & \mathsf{recv}(x^0_1); \mathsf{send}(0) \end{array}$$

$$\begin{array}{ll} B\text{'s role}: & \mathsf{recv}(x^1_0); \\ & \mathsf{send}\big(\{x^1_S\}^s_{ad(ad(\pi_2(x^1_0), pub(x^1_B)^{-1}), pub(\pi_1(x^1_0)))}\big) \end{array}$$

The symbols $x^0_0, x^0_1, x^0_A \ldots$ are all distinct variables of $\mathcal{X}$. The second instruction of program A implements only the reception of the last message by A.

Given a protocol $\mathcal{P}$, an agent executing a program p of $\mathcal{P}$ is represented by a *process* (p, σ) where σ is a ground substitution. A *configuration* is a pair (S, N) where S is a finite set of processes whose programs have disjoint sets of variables, and N is a set of ground terms representing the network controlled by an intruder. We define small step semantics for the execution of processes. Each step changes the running configuration $\big(\{(p, \sigma)\} \cup S, N\big)$ to $\big(\{(p', \sigma')\} \cup S, N'\big)$ by the execution of the instruction $instr := \mathsf{recv}(x); \mathcal{E}; \mathsf{send}(s)$ if $p = instr; p'$ and there exists a $\mathcal{R}$-solution θ of the equations in $\mathcal{E}\sigma$ such that $x\sigma\theta \in \mathcal{I}_{\mathcal{R}}(N)$, $\sigma' = \sigma\theta$ (execution of $\mathsf{recv}(x)$, control of the conditions in $\mathcal{E}$, and update of σ) and $N' = N \cup \{s\sigma'\}$ (execution of $\mathsf{send}(s)$).

We shall assume that for every execution step as above, the term sent $s\sigma'$ is ground. It means that the agent is able to construct the term s to be sent with the substitution in its initial process (its initial knowledge) or with the messages received from other agents. This is ensured by the following condition: we call an initial configuration $(\{(p_0, \sigma_0), \dots, (p_m, \sigma_m)\}, N_0)$ of a protocol $\mathcal{P}$ *runnable* iff for each $i \le m$, such that the program p_i is a sequence $(\mathsf{recv}(x_{i,j}); \mathcal{E}_{i,j}; \mathsf{send}(s_{i,j}))_{j \le n}$, for each $j \le n$, for each $x \in vars(s_{i,j})$, x is in the domain of σ_i or there exists $k \le j$ such that $x = x_{i,k}$.

EXAMPLE 3. *Any initial configuration with set of processes $\{(p_0, \sigma_0), (p_1, \sigma_1)\}$, where p_0 and p_1 are respectively the programs A's role and B's role of Example 2 and σ_0 and σ_1 are described below, is runnable for the protocol of Example 2 (a, b, k, s are constants):*

$$\sigma_0 = \{x^0_A \mapsto a, x^0_B \mapsto b, x^0_{Kab} \mapsto k\} \; \sigma_1 = \{x^1_B \mapsto b, x^1_S \mapsto s\}$$

4.2 Security Properties

Let $S_0 = \{(p_0, \sigma_0), \dots, (p_m, \sigma_m)\}$. An *interleaving* of S_0 is a finite sequence I, without repetition, of pairs of integers (i, j) where $0 \le i \le m$ (i is the index of a process of $\mathcal{S}_0$) and $0 \le j < |p_i|$ (j is the index of an instruction of p_i), which satisfies the following *ordering condition*: for each i with $0 \le i \le m$, the subsequence of I of pairs with first component i has the form $(i, 0), \dots, (i, n)$, with $n < |p_i|$. This condition expresses that I describes a partial linear execution of the respective programs of the processes, up to some point.

We say that a configuration (S, N) is reached from (S_0, N_0) via an interleaving I, denoted $(S_0, N_0) \to_I (S, N)$ if there is a finite sequence of configurations $(S_0, N_0), \dots, (S_{|I|}, N_{|I|}) = (S, N)$ such that for each $k < |I|$, (S_k, N_k) changes to (S_{k+1}, N_{k+1}) by execution of the jth instruction of the ith process of S_0, where (i, j) is the kth element of the sequence I. We are interested here in the following problem:

> **Protocol Insecurity (PI):** given a protocol $\mathcal{P}$, a runnable initial configuration (S_0, N_0) of $\mathcal{P}$, an interleaving I of S_0, and a ground term s, does there exist (S, N) such that $(S_0, N_0) \to_I (S, N)$ and $s \in N$?

We can express several trace properties of protocols as instances of PI. This typically the case of authentication failure (where one process p completes the protocol presumably with an interlocutor process p' whereas p' did not even start to run, and therefore p has been fooled in communicating only with the intruder), or of secrecy violation with some interleaving. Concerning the later problem, we shall also remark that the following problem of secrecy for any interleaving is reducible to PI since the number of interleaving is finite:

> **Weak Secrecy (WS):** given a protocol $\mathcal{P}$, a runnable initial configuration (S_0, N_0) of $\mathcal{P}$, and a ground term s, does there exists an interleaving I of S_0 and (S, N) such that $(S_0, N_0) \to_I (S, N)$ and $s \in N$?

4.3 Verification via Constraint Solving

Given some input (S_0, N_0) (with S_0 as above), I and s of PI, let us construct the set $\mathcal{C}$ containing, for each $k \le |I|$ such that (i, j) is the k-th element of the sequence I, and $\mathsf{recv}(x_{i,j}); \mathcal{E}_{i,j}; \mathsf{send}(s_{i,j})$ is the j-th instruction of the program p_i:

- the basic intruder constraint $T_k \Vdash x_{i,j}$, where $T_1 = N_0$ and $T_{k+1} := T_k \cup \{s_{i,j}\sigma_i\}$
- the equations of $\mathcal{E}_{i,j}\sigma_i$.

Moreover, $\mathcal{C}$ contains the additional basic intruder constraint $T_{|I|} \Vdash x$ (x is a fresh variable) and the equation $x = s$, which means that the secret is revealed.

Note that the subset of intruder constraints of $\mathcal{C}$ is well-formed. We can show that the $\mathcal{R}$-solvability of $\mathcal{C}$ is equivalent to PI. We shall present here the construction of $\mathcal{C}$ on our running example.

EXAMPLE 4. *As announced in Section 2, there is an attack on the protocol of Example 2, starting with the initial configuration (S_0, N_0) with S_0 given in Example 3, and $N_0 = \{0, a, b, pub(a), pub(b)\}$, when $\mathcal{R}$ is the standard Dolev-Yao theory of Section 3.2. In this attack, an intruder, claiming to be a (process p_0) sends to b (process p_1) the "message" $\langle a, \{0\}^a_{pub(b)}\rangle$. The answer of b is then:* $\{s\}_{ad(ad(\pi_2(\langle a,\{0\}^a_{pub(b)}\rangle),pub(b)^{-1}),pub(\pi_1(\langle a,\{0\}^a_{pub(b)}\rangle)))} \xrightarrow{*}_{\mathcal{R}} \{s\}^s_{ad(0,pub(a))}$ *and s is revealed since the encryption key $ad(0, pub(a))$ belongs to $\mathcal{I}_{\mathcal{R}}(N_0)$. The interleaving describing the trace of the attack is the sequence of length one $((1, 0))$ (it consists in a single instruction 0 of process p_1), and the (well-formed) set of basic intruder constraints and equations $\mathcal{C}$ associated to this interleaving is:*

$$\{N_0 \Vdash x^1_0; N_0, \{s\}_{ad(ad(\pi_2(x^1_0),pub(b)^{-1}),pub(\pi_1(x^1_0)))} \Vdash x; x = s\}$$

The first intruder constraint expresses that the process p_0 is able to receive the expected message x^1_0, i.e. that the intruder can construct it from its initial knowledge N_0 ($x^1_0 \in \mathcal{I}_{\mathcal{R}}(N_0)$). The second intruder constraint expresses that from p_0's answer and N_0, the intruder is able to deduce x. Finally, the last equation expresses that x is the secret. We can check that $\sigma = \sigma_1 \cup \{x^1_0 \mapsto \langle a, \{0\}^a_{pub(b)}\rangle, x \mapsto s\}$ is a $\mathcal{R}$-solution.

We shall give in the next two sections a resolution procedure for the problem of the satisfiability of a well-formed set of constraints. This will allow us to prove the main result of this paper.

THEOREM 1. *PI is decidable in non-deterministic polynomial time.*

COROLLARY 1. *WS is decidable in non-deterministic polynomial time.*

PROOF. The maximal length of an interleaving of S_0 is polynomial (in the size of S_0). $\square$

5. CHECKING GROUND CONSTRAINTS

In this section, we show how to solve intruder constraints without variables, *i.e.* how to decide, given a finite set $T \subseteq \mathcal{T}(\mathcal{F})$ such that $st(\mathcal{R}) \cap NF_R \cap \mathcal{T}(\mathcal{VF}) \subseteq T$, $0 \in T$ and given a term $u \in \mathcal{T}(\mathcal{F})$, whether $u \in \mathcal{I}_{\mathcal{R}}(T)$ holds or not. Following the approach of [9], we show first that $u \in \mathcal{I}_{\mathcal{R}}(T)$ ensures the existence of a *local* proof, *i.e.* a proof which only involves terms in $st(T\downarrow_{\mathcal{R}} \cup \{u\downarrow_{\mathcal{R}}\})$. Then, we show that using this result, we can determine in polynomial time in the size of T and u, whether $u \in \mathcal{I}_{\mathcal{R}}(T)$.

LEMMA 4 (LOCALITY). *Let T be a finite subset of $\mathcal{T}(\mathcal{F})$ such that $st(\mathcal{R}) \cap NF_R \cap \mathcal{T}(\mathcal{VF}) \subseteq T$ and $0 \in T$, and let a* *term $u \in \mathcal{T}(\mathcal{F})$. Every minimal size proof P of $T \vdash_{\mathcal{R}} u$ is labeled by terms in $st(T\downarrow_{\mathcal{R}} \cup \{u\downarrow_{\mathcal{R}}\})$ and if moreover P is a decomposition proof then it is labeled by terms in $st(T\downarrow_{\mathcal{R}})$.*

PROOF. *(sketch, see full version).* We prove the two results simultaneously by induction on the proof P. The only difficult case is when we have to take into account a rewriting step after the application of a visible function symbol, *i.e* when P is a decomposition proof. Clearly, $root(P)$ is a subterm of one of the direct subproof of P, however it remains to show that the root of the direct subproofs of P are labeled with subterms of T. It is treated by case analysis on the condition verified by the rewrite rule involved in the reduction. $\square$

Now, using this locality Lemma 4, we show that we can decide in polynomial time whether $u \in \mathcal{I}_{\mathcal{R}}(T)$.

PROPOSITION 1. *Given a finite set $T \subseteq \mathcal{T}(\mathcal{F})$ such that $st(\mathcal{R}) \cap NF_R \cap \mathcal{T}(\mathcal{VF}) \subseteq T$ and $0 \in T$, and a term $u \in \mathcal{T}(\mathcal{F})$, whether $u \in \mathcal{I}_{\mathcal{R}}(T)$ can be decided in polynomial time in $\|T \cup \{u\}\|_d$.*

PROOF. By Lemmas 3 and 4, if $u \in \mathcal{I}_{\mathcal{R}}(T)$ then there exists a proof P of $T \vdash_{\mathcal{R}} u$ labeled only with terms in $st(T\downarrow_{\mathcal{R}} \cup \{u\downarrow_{\mathcal{R}}\})$. To decide the existence of such a proof tree, we construct (following [20]) the set $\mathcal{S}$ of ground Horn clauses containing:

- every $\Rightarrow P(s)$ s.t. $s \in T\downarrow_{\mathcal{R}}$,
- every $P(s_1), \ldots, P(s_n) \Rightarrow P(f(s_1, \ldots, s_n)\downarrow_{\mathcal{R}})$ s.t. $s_1, \ldots, s_n, f(s_1, \ldots, s_n)\downarrow_{\mathcal{R}} \in st(T\downarrow_{\mathcal{R}} \cup \{u\downarrow_{\mathcal{R}}\})$ and $f \in \mathcal{VF}$
- $P(u\downarrow_{\mathcal{R}}) \Rightarrow$.

The clauses of $\mathcal{S}$ of two first kind above implement a marking of every ground subterm $t \in st(T\downarrow_{\mathcal{R}} \cup \{u\downarrow_{\mathcal{R}}\})$ such that there exists a proof of $T \vdash_{\mathcal{R}} t$. Therefore, the existence of a proof of $T \vdash_{\mathcal{R}} u$ is equivalent to the HORN-SAT problem for $\mathcal{S}$, which size is polynomial in $\|T \cup \{u\}\|_d$ (the degree of the polynomial is the maximum of the arities of symbols of $\mathcal{VF}$). Hence, $u \in \mathcal{I}_{\mathcal{R}}(T)$ can be decided in polynomial time. $\square$

6. SATISFIABILITY OF WELL-FORMED SETS OF INTRUDER CONSTRAINTS

We shall lift the decision result of Section 5 with a non-deterministic polynomial time procedure to decide the satisfiability w.r.t. $\mathcal{R}$ of well-formed sets of basic intruder constraints (with variables) and equations.

6.1 Constraints Transformation Rules

We present in Figure 1 a set of transformation rules which operate on tuples of the form $(\mathcal{P}, \mathcal{C}, \mathcal{S})$, called *constraints systems* where:

- $\mathcal{P}$ is a set of equations and basic intruder constraints,
- $\mathcal{C}$ is a set of intruder constraints,
- $\mathcal{S}$ is a set of equations in solved form representing bindings in the solution, *i.e.* $\mathcal{S} = \{x_1 = t_1, ..., x_n = t_n\}$ where each $x_i \in \mathcal{X}$ and has only one occurrence in $\mathcal{S}$.

We may associate a substitution $\{x_1 \mapsto t_1, ..., x_n \mapsto t_n\}$ to the third component $\mathcal{S}$ of a system. Below, we shall make no distinction between $\mathcal{S}$ and its associated substitution.

DEFINITION 4. *A $\mathcal{R}$-solution of a system $(\mathcal{P}; \mathcal{C}; \mathcal{S})$ is a grounding substitution σ such that σ is a $\mathcal{R}$-solution of each intruder constraint in $\mathcal{P} \cup \mathcal{C}$, σ is a $\mathcal{R}$-solution of each equation in $\mathcal{P}$, and σ is a unifier of each equation in $\mathcal{S}$.*

Given an initial system of the form $(\mathcal{B} \cup \mathcal{E}, \emptyset, \emptyset)$, where $\mathcal{B}$ is a finite well-formed set of basic intruder constraints and $\mathcal{E}$ is a finite set of equations, the repeated non-deterministic application of the rules of Figure 1 shall terminate (Section 6.2) and produce (in at least one derivation branch) a system in solved form $(\emptyset, \emptyset, \mathcal{S})$ (such systems always have a $\mathcal{R}$-solution) iff $(\mathcal{B} \cup \mathcal{E}, \emptyset, \emptyset)$ has a $\mathcal{R}$-solution (Sections 6.3 and 6.4). This gives a non-deterministic polynomial time procedure for the decision of the satisfiability which is shown NP-hard in Section 7.

From now on, we shall note $\Longrightarrow_{\mathsf{N}}$, $\Longrightarrow_{\mathsf{U}}$,... the binary relation defined by the application respectively of the above rule (N), (U)..., $\Longrightarrow$ denotes the union of all these relations and $\Longrightarrow^+$ and $\Longrightarrow^*$ are the respective transitive and reflexive-transitive closures of $\Longrightarrow$.

6.2 Termination

PROPOSITION 2 (TERMINATION). *The relation $\Longrightarrow$ is strongly terminating. Moreover, given a system $(\mathcal{P}_0; \emptyset; \emptyset)$, for every transformation sequence $(\mathcal{P}_0; \emptyset; \emptyset) \Longrightarrow (\mathcal{P}_1; \mathcal{C}_1; \mathcal{S}_1) \Longrightarrow \ldots \Longrightarrow (\mathcal{P}_n; \mathcal{C}_n; \mathcal{S}_n)$, the length n, the number of successors of every $(\mathcal{P}_i, \mathcal{C}_i, \mathcal{S}_i)$ with $\Longrightarrow$ and the value $\|\mathcal{P}_i\| + \|\mathcal{C}_i \cup \mathcal{S}_i\|_d$ are polynomial in $\|\mathcal{P}_0\|$ and $\|\mathcal{R}\|$.*

PROOF. *(sketch, see full version for details).* Let the complexity of system $(\mathcal{P}; \mathcal{C}; \mathcal{S})$ be the tuple $(|\mathcal{P}|, nb(\mathcal{P}), nbv(\mathcal{C}), |\mathcal{C}|)$ ordered lexicographically where: $nb(\mathcal{P})$ is the number of terms in $st(\mathcal{P})$ which are unifiable with a left member of a rule of $\mathcal{R}$, and $nbv(\mathcal{C})$ is the number of distinct variables in $\mathcal{C}$. We can show that each rule of Figure 1 reduces the complexity, hence that $\Longrightarrow$ terminates. $\square$

6.3 Correctness

The following proposition shows that the constraint solving system defined in Figure 1 is correct.

PROPOSITION 3 (CORRECTNESS). *For every system $(\mathcal{P}; \emptyset; \emptyset)$, if $(\mathcal{P}; \emptyset; \emptyset) \Longrightarrow^* (\emptyset; \emptyset; \mathcal{S})$ then $(\mathcal{P}; \emptyset; \emptyset)$ has a $\mathcal{R}$-solution.*

PROOF. *(sketch, see full version for details).* By induction on the length of the derivation, we show for every rule (R), that if $(\mathcal{P}_1; \mathcal{C}_1; \mathcal{S}_1) \Longrightarrow_{\mathsf{R}} (\mathcal{P}_2; \mathcal{C}_2; \mathcal{S}_2)$ and the second system $(\mathcal{P}_2; \mathcal{C}_2; \mathcal{S}_2)$ has a $\mathcal{R}$-solution σ, then σ is also a $\mathcal{R}$-solution of $(\mathcal{P}_1; \mathcal{C}_1; \mathcal{S}_1)$. $\square$

6.4 Completeness

We show now the completeness of the constraint solving system of Figure 1 (Proposition 4). We shall first prove three technical lemmas: Lemma 5 and Lemma 6 are used in the proof of Lemma 7, which is the key in the proof of the Proposition 4 – it establishes the completeness of the rule (VE).

LEMMA 5. *Let $T \subseteq NF_{\mathcal{R}}$ be such that $st(\mathcal{R}) \cap NF_{\mathcal{R}} \cap \mathcal{T}(\mathcal{VF}) \subseteq T$, let $u \in NF_{\mathcal{R}}$, $v \in st(u)$ such that $v \notin st(T)$, and let P be a proof of $T \vdash_{\mathcal{R}} u$. There exists a composition proof of $T \vdash_{\mathcal{R}} v$.*

$$\frac{\mathcal{P} \cup \{e[u]\};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P} \cup \{e[r]\};\ \mathcal{C}\eta;\ \mathcal{S}\eta \cup \eta}\ (\mathsf{N})$$

Narrowing
e is an equation or an intruder constraint, $u \notin \mathcal{X}$, $l \to r$ is a fresh variant (a copy in which all the variables have been renamed) of a rule of $\mathcal{R}$, $\eta = mgu(l\mathcal{S}, u\mathcal{S})$, $root(l) = root(u)$.

$$\frac{\mathcal{P} \cup \{t_1 = t_2\};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C}\eta;\ \mathcal{S}\eta \cup \eta}\ (\mathsf{U})$$

Syntactic Unification
$\eta = mgu(t_1\mathcal{S}, t_2\mathcal{S})$.

$$\frac{\mathcal{P} \cup \{c\};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C} \cup \{c\mathcal{S}\};\ \mathcal{S}}\ (\mathsf{B})$$

Blocking
c is an intruder constraint.

$$\frac{\mathcal{P};\ \mathcal{C};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C}\{x \mapsto t\};\ \mathcal{S}\{x \mapsto t\} \cup \{x \mapsto t\}}\ (\mathsf{VE})$$

Variable Elimination
$x \in vars(\mathcal{C}), t \in st(\mathcal{C}) \setminus vars(\mathcal{C})$, there is no occurrence of x in t.

$$\frac{\mathcal{P};\ \mathcal{C} \cup \{T \Vdash u\};\ \mathcal{S}}{\mathcal{P};\ \mathcal{C};\ \mathcal{S}}\ (\mathsf{G})$$

Ground
if all the terms in T and u are ground and $u \in \mathcal{I}_{\mathcal{R}}(T)$.

Figure 1: Constraint Transformation Rules

Proof. We assume that P is a minimal proof of $T \vdash_{\mathcal{R}} u$. If P contains a node labeled with v, then it is the root of a proof of $T \vdash_{\mathcal{R}} v$ as expected. This proof is indeed a composition proof: otherwise, by Lemma 4, we would have $v \in st(T)$, which contradicts the hypotheses.

We show now that P necessarily contains one node labeled with v, by contradiction. Assume that P contains no such node. We shall construct recursively a path in P, from the root up to one leaf, every node of which is labeled with a term t such that $v \in sst(t)$, and we shall show in parallel that the existence of such a path leads to a contradiction. By hypothesis, v is a subterm of the label u of the root of P. Assume that this condition is also true for all the nodes of a path $(s_0, \ldots, s_k)$ in P, labeled respectively with $t_0 = u$, $t_1, \ldots, t_k$ (hence $v \in sst(t_0), \ldots, v \in sst(t_k)$) and let us consider the sons of the last node s_k of the path.

If s_k is a leaf, then $t_k \in T$ by Definition 3 and $v \in sst(t_k)$. It contradicts the hypothesis that $v \notin st(T)$.

If s_k has n sons $P_1, \ldots, P_n$, then we have (by Definition 3) $t_k = f(root(P_1), \ldots, root(P_n)) \downarrow_{\mathcal{R}}$ for some $f \in \mathcal{VF}$, and thanks to Lemma 1, there are only two cases to consider: 0 or 1 step of rewriting. In both case, we show (see full version) that we can set the next node s_{k+1} of the path as the root of one of the P_i, $i \leq n$. $\square$

The following technical lemma allows to apply replacements on proof tree labels.

Lemma 6. *Let $v = g(v_1, \ldots, v_k) \in NF_{\mathcal{R}} \setminus (st(\mathcal{R}) \cap NF_{\mathcal{R}} \cap \mathcal{T}(\mathcal{VF}))$, with $g \in \mathcal{VF}$, let δ be the replacement $\delta = \{v \mapsto 0\}$ and let $u_1, \ldots, u_n \in NF_{\mathcal{R}}$ and $u = f(u_1, \ldots, u_n) \downarrow_{\mathcal{R}}$ with $f \in \mathcal{VF}$. If $u \neq v, v_1, \ldots, v_k$, then $u\delta = f(u_1\delta, \ldots, u_n\delta) \downarrow_{\mathcal{R}}$.*

Proof. By Lemma 1 we have to consider only two cases:
1. $u = f(u_1, \ldots, u_n)$. In this case, we have: $u\delta = f(u_1, \ldots, u_n)\delta = f(u_1\delta, \ldots, u_n\delta)$ since $v \neq f(u_1, \ldots, u_n)$.
2. $f(u_1, \ldots, u_n) \xrightarrow{\Lambda}_{\mathcal{R}} u$, with a rewrite rule $f(l_1, ..., l_n) \to r \in \mathcal{R}$. We denotes by Pos_v the set of all occurrences of v in the term $f(u_1, \ldots, u_n)$. If $\Lambda \in Pos_v$, then $v \notin NF_{\mathcal{R}}$ which contradicts the hypothesis. If for all $p \in Pos_v$, there exists p' a prefix of p such that $f(l_1, ..., l_n)|p' \in \mathcal{X}$ then $f(u_1\delta, \ldots, u_n\delta) \xrightarrow{\Lambda}_{\mathcal{R}} u\delta$ with the same rule as before. Otherwise there exists $p \in Pos_v$ such that $f(l_1, ..., l_n)|p \notin \mathcal{X}$. We have $p \neq \Lambda$, let $p = i.p'$. The case where $l_i|p' \in \mathcal{X}$ is impossible due to the choice of p, $l_i|p' \in \mathcal{PF}$ is impossible since $head(v) \in \mathcal{VF}$. Lastly, if $l_i|p' \in \mathcal{VF}$ then either there exists j such that $l_i|p'.j = r$ and u is equal to v_j which contradicts the hypotheses, or $l_i|p' \in st(\mathcal{R}) \cap NF_R \cap \mathcal{T}(\mathcal{VF})$ which means that $v \in st(\mathcal{R}) \cap NF_R \cap \mathcal{T}(\mathcal{VF})$, and it is a contradiction. $\square$

Given two substitutions σ_1 and σ_2, we write $\sigma_1 \ll \sigma_2$ iff $\{x\sigma_1 |\ x \in dom(\sigma_1)\} \ll \{x\sigma_2 |\ x \in dom(\sigma_2)\}$.

Lemma 7. *Let σ be a minimal (w.r.t. $\ll$) $\mathcal{R}$-solution of a well-formed set $\mathcal{C}$ of intruder constraints such that all the terms in $\mathcal{C}\sigma$ are in $NF_{\mathcal{R}}$. For all $x \in vars(\mathcal{C})$, there exists $t \in st(\mathcal{C}) \setminus vars(\mathcal{C})$ such that $t\sigma = x\sigma$.*

Proof. *(sketch, see full version for technical details).* Let $(C_1, \ldots, C_\ell)$ be a sequence of the constraints of $\mathcal{C}$ ordered as in Definition 2, and for each $i \leq \ell$, let S_i and r_i be respectively the set of hypotheses and target of C_i, and $C_i\sigma$ be the (ground) constraint obtained from C_i by instantiating all the terms in its hypotheses and target with σ. We reason by contradiction. Assume that there exists $x \in vars(\mathcal{C})$ such that for all $t \in st(\mathcal{C}) \setminus vars(\mathcal{C})$, $t\sigma \neq x\sigma$, and let δ be the replacement $\{x\sigma \mapsto 0\}$. We show that $\sigma' = \sigma\delta$ is also a $\mathcal{R}$-*solution* of $\mathcal{C}$, and it contradicts the minimality of σ. Note first that for each $i \leq \ell$, and each $s \in S_i \cup \{r_i\}$, $s(\sigma\delta) = (s\sigma)\delta$.

Let $m = \min\{i \mid x\sigma \in st(C_i\sigma)\}$. By the above hypothesis, there exists $y \in vars(C_m)$ such that $x\sigma \in st(y\sigma)$. Otherwise, there would exist $t \in st(\mathcal{C}_m) \setminus vars(\mathcal{C}_m)$ such that $t\sigma = x\sigma$. Moreover, each such variable y occurs in r_m and not in the hypotheses S_m. Otherwise, since $\mathcal{C}$ is well-formed, there would exists $i < m$ such that $y \in vars(r_i)$, hence $x\sigma \in st(C_i\sigma)$.

For each $i < m$, $x\sigma \notin st(C_i\sigma)$, hence $C_i\sigma' = (C_i\sigma)\delta = C_i\sigma$ and σ' is a $\mathcal{R}$-*solution* of C_i.

Let $i \geq m$ and let P_i be a proof of $S_i\sigma \vdash_{\mathcal{R}} r_i\sigma$. By Lemma 3, there exists a proof P_m of $S_m\sigma \vdash_{\mathcal{R}} r_m\sigma$ on which we can apply Lemma 5 in order to obtain a composition proof P_x of $S_m\sigma \vdash_{\mathcal{R}} x\sigma$. We construct a proof P_i'' of $S_i\sigma' \vdash_{\mathcal{R}} r_i\sigma'$, by transformation on the proof tree P_i, using subproofs of P_x. Roughly, if we let $x\sigma = f(u_1, \ldots, u_n)$, we replace in P_i every subtree whose root is labeled by some u_j by the jth direct subtree of P_x (which is indeed a proof of $S_m\sigma \vdash_{\mathcal{R}} u_j$), and we replace in P_i every subtree whose root is labeled by $x\sigma$ with a leaf labeled by 0. Finally, we apply δ to all the other nodes of P_i and, using Lemma 6, we can show that the tree P_i'' obtained is indeed a proof of $S_i\sigma' \vdash_{\mathcal{R}} r_i\sigma'$, and hence that σ' is a $\mathcal{R}$-*solution* of C_i by Lemma 3. $\square$

PROPOSITION 4 (COMPLETENESS). *Let $\mathcal{B}$ be a finite well-formed set of basic intruder constraints, $\mathcal{E}$ a finite set of first-order equations. If $(\mathcal{B} \cup \mathcal{E}; \emptyset; \emptyset)$ has a $\mathcal{R}$-solution, then there exists a sequence of reductions of the form $(\mathcal{B} \cup \mathcal{E}, \emptyset, \emptyset) \Longrightarrow^* (\emptyset, \emptyset, \mathcal{S})$.*

PROOF. *(sketch, see full version).*We show, by induction on the complexity of systems, the more general result that if there exists a $\mathcal{R}$-normal solution σ of a system $(\mathcal{P}, \mathcal{C}, \mathcal{S}')$ such that the set of intruder constraints in $\mathcal{PS}' \downarrow_{\mathcal{R}} \cup \mathcal{C}$ is well-formed, and the terms in $\mathcal{C}\sigma$ are in $NF_{\mathcal{R}}$, then there exists a sequence of reductions of the form $(\mathcal{P}, \mathcal{C}, \mathcal{S}') \Longrightarrow^* (\emptyset, \emptyset, \mathcal{S})$.

The base case $(\emptyset, \emptyset, \mathcal{S}')$ is trivial. For the induction step, we assume that σ is a minimal (w.r.t. $\lll$) $\mathcal{R}$-normal solution as above, and we show that for each $(\mathcal{P}, \mathcal{C}, \mathcal{S}')$, we can apply one of the constraint transformation rules of Figure 1, and that the system obtained has a $\mathcal{R}$-normal solution σ' of the above form. The difficult case is the application of the rule (VE). It is treated by using Lemma 7. To conclude, we observe that the above result can be applied to $(\mathcal{B} \cup \mathcal{E}; \emptyset; \emptyset)$. $\square$

Using the above Propositions, we deduce a NP decision procedure for the decision of satisfiability.

THEOREM 2. *Given a convergent public-collapsing TRS $\mathcal{R}$, a finite well-formed set $\mathcal{B}$ of basic intruder constraints and a finite set $\mathcal{E}$ of equations, the existence of $\mathcal{R}$-solution of $\mathcal{B} \cup \mathcal{E}$ is decidable in non-deterministic polynomial time.*

PROOF. By Proposition 2, the repeated application of the rules of Figure 1 to the system $(\mathcal{B} \cup \mathcal{E}; \emptyset; \emptyset)$ gives a finite deduction tree whose nodes are labeled by constraints systems and every node is obtained from its parent node by application of a rule of Figure 1. According to Propositions 4 and 3, $\mathcal{B} \cup \mathcal{E}$ is satisfiable (w.r.t. $\mathcal{R}$) if and only if there exists a leaf with a label of the form $(\emptyset; \emptyset; \mathcal{S})$ in the deduction tree.

By Proposition 2, both the depth d and the maximal branching degree b of this deduction tree is polynomial in $\|\mathcal{B} \cup \mathcal{E}\|$ and $\|\mathcal{R}\|$. The non-deterministic polynomial time algorithm consists in choosing a path p in the deduction tree, *i.e.* a sequence of length at most d of natural numbers smaller or equal to b, and checking that the path p leads to a leaf labeled by $(\emptyset; \emptyset; \mathcal{S})$ in the deduction tree. It can be done in polynomial time by the iterative application of the instances of rules of Figure 1 corresponding to the components of p. The application of every instance of rule takes a polynomial time. It is obvious for (N), (U), (B) and (VE). For (G), it is a consequence of Proposition 1 and the bound of Proposition 2 for the system-dag-size of the labels of the deduction tree. $\square$

7. NP-HARDNESS

We show now that PI and WS are NP-hard for polynomial time reductions by reduction of 3-SAT. The proof is inspired from the one given by [26]. However, the protocol built from the given instance of 3-SAT is reduced to a minimum thanks to the flexibility of our formalism concerning the choice of a rewriting system.

Let $X_1, ..., X_n$ be propositional variables and let us consider the following instance of 3-SAT:

$$\bigwedge_{i=1}^{m} (X_{\alpha_{i,1}}^{\epsilon_{i,1}} \vee X_{\alpha_{i,2}}^{\epsilon_{i,2}} \vee X_{\alpha_{i,3}}^{\epsilon_{i,3}})$$

where $\alpha_{i,j} \in 1..n$ and $\epsilon_{i,j} \in \{0, 1\}$ and u^1 (resp. u^0) denotes u (resp. $\neg u$) for any term u.

We use a signature made of $\mathcal{VF} = \{0, 1, \pi_1(\_), \ldots, \pi_n(\_), \langle \_, \ldots, \_ \rangle\}$ and $\mathcal{PF} = \{\_ \wedge \_, \_ \vee \_, \neg \_, secret\}$, where $\langle\ \rangle$ has arity n. Let $\mathcal{R}$ be a convergent public collapsing TRS which defines the truth tables of $\wedge$, $\vee$, $\neg$ with $0 \wedge 0 \to 0$ *etc* (0 is false and 1 is true) and the projections with $\pi_i(\langle x_1, \ldots, x_n \rangle) \to x_i$ for $i = 1, \ldots, n$.

We construct a protocol $\mathcal{P}$ with only one program made up of one instruction:

$$\mathsf{recv}(x); f_1(x) \wedge \ldots \wedge f_m(x) = 1; \mathsf{send}(secret)$$

where, for all $i \leq n$, $f_i(x) = \pi_{\alpha_{i,1}}(x)^{\epsilon_{i,1}} \vee \pi_{\alpha_{i,2}}(x)^{\epsilon_{i,2}} \vee \pi_{\alpha_{i,3}}(x)^{\epsilon_{i,3}}$ (we omit the parenthesis in the expressions with $\wedge$ and $\vee$) . Finally, let S_0 contains one process (p_0, σ_0) with $\sigma_0 = \emptyset$ and $N_0 = \{0, 1\}$.

We can show that PI for $\mathcal{P}$, (S_0, N_0), *secret* and the interleaving $((0, 0))$ has a solution iff the above instance of 3-SAT has a solution represented by $x = \langle X_1, \ldots, X_n \rangle$ (each X_i is 0 or 1) and this term x is in $\mathcal{I}_{\mathcal{R}}(N_0)$. Note that the choice of the interleaving is limited to $((0, 0))$ (or the empty sequence), hence the reduction is also valid to show the NP-hardness of WS. By Theorem 1 and Corollary 1, we deduce that PI and WS are NP-complete and, with the construction of Section 4.3, it implies that the problem of solvability of well-formed sets of basic intruder constraints and equations (Theorem 2) is also NP-complete.

8. CONCLUSION

We have defined a complete inference system for solving equations and intruder constraints modulo convergent and public-collapsing TRS, and we have shown how it provides a generic non-deterministic polynomial time procedure for the verification of the security of cryptographic protocols in presence of a finite number of sessions, and with the addition of operators whose semantics are defined by a convergent public-collapsing TRS.

A natural extension to this work is the search of public-collapsing theories other than those described in Section 3.2, for the weakening of security hypotheses. For instance, one may want to consider dictionary attacks [12]. An exclusive or operator $+$ can be axiomatized by the rewrite rules $x + x \to 0$, $x + 0 \to x$, $x + x + y \to y$ with associativity and commutativity (AC) of $+$. The three first rules fulfill our public-collapsing condition. Hence, we should consider to extend our solving procedure to a procedure modulo AC in order to deal with xor, like [6, 9].

We could also study the generalization of the class of convergent TRS handled. An application could be for instance to model honest protocol transitions by rewrite rules, making the guess of an interleaving in the proof of Corollary 1 unnecessary.

At last, and this is a more difficult task, we could try to extend our result to the decision of static equivalence (following [2]). A solution could be to extend the class of constraints under consideration. As noted in Section 3.3, intruder constraints correspond to second order equations (modulo a convergent TRS) of the form $x(t_1, \ldots, t_n) = t$. Being able to deal with equations of the form $x(t_1, \ldots, t_n) = x(s_1, \ldots, s_n)$ could permit us to study properties related to observation equivalence, hence to consider some properties more general than the weak secrecy.

9. REFERENCES

[1] R.M. Amadio and W. Charatonik. On Name Generation and Set-Based Analysis in the Dolev-Yao Model. In *Proc. 13th Int. Conference on Concurrency Theory (CONCUR)*, 2002.

[2] M. Abadi and C. Cortier. Deciding Knowledge in Security Protocols under Equational Theories. In *Proc. 31st Int. Colloquium on Automata, Languages and Programming (ICALP)*, 2004.

[3] M. Abadi and C. Fournet. Mobile Values, New Names, and Secure Communication. In *Proc. 28th ACM Symposium on Principles of Programming Languages (POPL)*, 2001.

[4] R.M. Amadio and D. Lugiez. On the Reachability Problem in Cryptographic Protocols. In *Proc. 12th Int. Conference on Concurrency Theory (CONCUR)*, 2000.

[5] S. Bellovin. Problem Areas for the IP Security Protocols. In *Proc. 6th Usenix Unix Security Symposium*, 1996.

[6] Y. Chevalier, R. Kuester, M. Rusinowitch, and M. Turuani. An NP Decision Procedure for Protocol Insecurity with Xor. In *Proc. 18th Int. Symposium on Logic in Computer Science (LICS)*, 2003.

[7] Y. Chevalier, R. Kuester, M. Rusinowitch, and M. Turuani. Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Product in Exponents. In *Proc. 23rd Foundations of Software Technology and Theoretical Computer Science (FST&TCS)*, 2004.

[8] H. Comon and V. Cortier. Tree Automata with One Memory, Set Constraints and Cryptographic Protocols. *Theoretical Computer Science*, 2004. To Appear.

[9] H. Comon-Lundh and V. Shmatikov. Intruder Deductions, Constraint Solving and Insecurity Decision in Presence of Exclusive Or. In *Proc. 18th Int. Symposium on Logic in Computer Science (LICS)*, 2003.

[10] H. Comon-Lundh and R. Treinen. Easy Intruder Deductions. In *Verification: Theory & Practice, Essays Dedicated to Zohar Manna on the Occasion of His 64th Birthday*, 2003.

[11] R. Corin and S. Etalle. An Improved Constraint-Based System for the Verification of Security Protocols. In *Proc. 9th Int. Symposium on Static Analysis, (SAS)*, 2002.

[12] S. Delaune and F. Jacquemard. A Theory of Dictionary Attacks and its Complexity. In *Proc. 17th Computer Security Foundations Workshop (CSFW)*, 2004.

[13] D.E. Denning and G.M. Sacco. Timestamps in Key Distribution Protocols. In *Communications of the ACM*, 24(8), 1981.

[14] N. Dershowitz and J.-P. Jouannaud. *Rewrite Systems.* Elsevier and MIT Press, 1990.

[15] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov. Undecidability of Bounded Security Protocols. In *Proc. Workshop on Formal Methods and Security Protocols (FMSP)*, 1999.

[16] D. Dolev and A. Yao. On the Security of Public Key Protocols. In *Proc. Transactions on Information Theory*, 1983.

[17] T. Genet and F. Klay. Rewriting for Cryptographic Protocol Verification. In *Proc. 17th Int. Conference on Automated Deduction (CADE)*, 2000.

[18] S. Goldwasser and S. Micali. Probabilistic Encryption. In *Journal of Computer and System Science*, 28(2), 1984.

[19] C. Lynch and C. Meadows. On the Relative Soundness of the Free Algebra Model for Public Key Encryption. In *Proc. 4th Workshop on Issues in the Theory of Security (WITS)*, 2004.

[20] D. McAllester. Automatic Recognition of Tractability in Inference Relations. In *Journal of the ACM*, 40(2), 1993.

[21] C. Meadows. Applying Formal Methods to the Analysis of a Key Management Protocol. In *Journal of Computer Security*, 1(1), 1992.

[22] J. Millen. On the Freedom of Decryption. In *Information Processing Letters*, 86(6), 2003.

[23] J. Millen and V. Shmatikov. Constraint Solving for Bounded-Process Cryptographic Protocol Analysis. In *Proc. 8th Conference on Computer and Communications Security (CCS)*, 2001.

[24] J. Millen and V. Shmatikov. Symbolic Protocol Analysis with Products and Diffie-Hellman Exponentiation. In *Proc. 16th Computer Security Foundations Workshop (CSFW)*, 2003.

[25] D. Monniaux. Abstracting Cryptographic Protocols with Tree Automata. In *Proc. 6th Int. Static Analysis Symposium (SAS)*, 1999.

[26] M. Rusinowitch and M. Turuani. Protocol Insecurity with Finite Number of Sessions is NP-complete. In *Proc. 14th Computer Security Foundations Workshop (CSFW)*, 2001.

[27] V. Shmatikov. Decidable Analysis of Cryptographic Protocols with Products and Modular Exponentiation. In *Proc. 13th European Symposium on Programming (ESOP)*, 2004.

[28] M. Turuani. *Sécurité des Protocoles Cryptographiques: Décidabilité et Complexité*. PhD thesis, Université Henri Poincaré - Nancy 1, 2003.