Unité Mixte de Recherche 5104 CNRS - INPG - UJF
Centre Equation
2, avenue de VIGNATE
F-38610 GIERES
tel : +33 456 52 03 40
fax : +33 456 52 03 50
http://www-verimag.imag.fr

# An NP Decision Procedure for Generic Dolev-Yao Constraints with Atomic Keys

*Laurent Mazaré*

**Report n$^o$ 21**

November 30, 2004

# An NP Decision Procedure for Generic Dolev-Yao Constraints with Atomic Keys

*Laurent Mazaré*

November 30, 2004

**Abstract**

When considering only a finite number of sessions, protocol insecurity is equivalent to satisfiability of a constraint involving an operator derived from Dolev and Yao $\vdash$. Considering only constraints satisfying some properties (these constraints are called well-formed constraints), their satisfiability has been proven NP-complete. As constraints coming from verification of protocol insecurity are well-formed, this result can be generalised to prove that protocol insecurity is a NP-complete problem (when considering a finite number of sessions). However, if we consider constraints with no restriction at all, this allows us to model more complex properties on protocols such as fairness or protocols composition. This paper extends the NP-completness result to general constraints, giving a sketch of the proof that is more complex than the proof for well-formed constraints. We then show how this could be used to verify other security properties.

**Keywords**: Security, Cryptographic Protocols, Dolev-Yao Constraints, Fairness, Protocol Composition.

**Reviewers**:

**Notes**:

**How to cite this report**:

@techreport { verimag-TR-2004-21,
title = { An NP Decision Procedure for Generic Dolev-Yao Constraints with Atomic Keys},
authors = { Laurent Mazaré},
institution = { Verimag Technical Report },
number = {21},
year = { 2004},
note = { }
}

# 1 Introduction

Almost ten years ago, Gavin Lowe found the well-known man in the middle attack (see [8] and [9]) against the Needham-Schroeder protocol that was first described in 1978 in [14]. This attack is the most commonly used example to show the value of formal methods in the verification of protocols. These methods allow to find a non-trivial but easily understandable flaw on a protocol on which no attack had been found in the first fifteen years. Since then, studies focused on verifying secrecy properties (for Dolev-Yao's model, see [6]) and produced two principal results: secrecy for a protocol with an infinite number of sessions is an undecidable problem whereas secrecy for a protocol with a finite number of sessions is a NP-complete problem (see amongst others [2] or [5] for a generalized result). To prove the second result, one possible way is to consider constraints based on Dolev-Yao's operator ( [13], [10]). The operator $\vdash$ when used in $E \vdash m$ means that the message $m$ is deducible from knowledge (i.e. a finite set of messages) $E$. Then, a Dolev-Yao constraint is a conjunction of atomic equations involving the $\vdash$ operator. The Dolev-Yao model specifies what an intruder could do to guess the secrets: the intruder can intercept any message, forge new messages and send them using an honest agent's identity. In this model, the secrecy problem (still with a finite number of sessions) is equivalent to satisfiability of a given constraint. The question of how to build constraints from protocol specifications has widely been discussed before and can be found in [13], [5] or in [2]. The idea is to test any possible interleavings of the different sessions (which are in finite number as the number of sessions is itself finite). For each interleaving, secrecy will be equivalent to a constraint. The main point is that constraints built from a protocol are always in a very specific class of constraints called well-formed constraints. Thus, satisfiability of well-formed constraints has quickly been proven NP-complete but there are no such results for general constraints.

In this paper, after giving a simple definition of constraints, we will prove that satisfiability remains NP-complete when considering any possible constraint. Then, we will extend constraints by authorising the use of the logical not operator. This extension will allow us to prove that problems more complex than secrecy are still decidable. When verifying a property on a protocol, three distinct constraints will appear: the first one specifies the initial state of the intruder (or of any agent), the second describes the unfolding of the protocol and the last one gives the property that we want to prove false. The second constraint is well-formed so previous studies omitted the first constraint and forced the last constraint to be a unique $E \vdash m$ constraint where $E$ was the final knowledge of the intruder. Our results allow us to use general constraints for the first and the third constraint. Thus, we can verify properties that we could not express using well-formed constraints only and this will allow verification of a larger class of properties. In this paper, we will give examples for fairness and protocol composition but we believe that there are lots of properties that can be described using our constraints. We will also give a sketch of the proof of our main results.

The remainder of this paper is organized as follows. In section 2, we will briefly recall the usual definitions needed for building constraints. Then, section 3 will introduce constraints and gives the two main theorems. Section 4 will give a sketch of the proof of the first main theorem (the same sketch is used to prove the second main theorem). After that, section 5 will give the arguments needed to prove that our problem (e.g. satisfiability of our constraints) is NP-complete. Section 6 will show two straightforward applications of the main results for protocol verification. Finally, section 7 will conclude this paper.

# 2 Basic Definitions

Let $A$ and $X$ be two infinite countable disjoint sets. $A$ is the set of atomic messages usually written $a$, $b$ and so on. $X$ is the set of message variables written $x$, $y$ and so on.

**Definition 1** *Let $\Sigma$ be the signature $A \cup \{pair, encrypt\}$ where pair and encrypt are two binary functions. The atomic messages are considered constant functions. Then, a* message *is a first order term over $\Sigma$ and the set of variables $X$, namely an element of $T(\Sigma, X)$. A message is said to be* closed *iff it is a closed term of $T(\Sigma, X)$, i.e. a term of $T(\Sigma)$.*

In the rest of this paper, we will use the well-known notations $\langle m_1, m_2 \rangle = pair(m_1, m_2)$ and $\{m_1\}_{m_2} = encrypt(m_1, m_2)$. The height the tree representing message $m$ can easily be defined recursively and will be noted $h(m)$. The set of variables used in a message $m$ is noted $var(m)$. The substitutions $\sigma$ from $X$ to $T(\Sigma, X)$ are defined as usual. Application of $\sigma$ to message $m$ will be noted $m\sigma$. If $\sigma$ is defined by $x\sigma = n$ and $y\sigma = y$ for any other variables $y$, then we can write $m[x\backslash n]$ instead of $m\sigma$. The domain of a substitution $\sigma$ is the set $dom(\sigma)$ of variables such that $x\sigma \neq x$.

Using the pair operator, it is possible to introduce n-tuples. Then, we will use the $t^n$ notation to denote tuples composed by $n$ times the $t$ message. Formally, $t^n$ is recursively defined by $t^1 = t$ and $t^{n+1} = \langle t, t^n \rangle$

We will use Dolev-Yao intruder's model [6]. The intruder controls the network: he can intercept any message, forge messages using his initial knowledge and intercepted messages and send these messages to other agents usurping an agent's identity. To forge new messages, the intruder uses Dolev-Yao's theory $\vdash$. If $E$ is a set of messages and $m$ is a message, $E \vdash m$ means that $m$ can be deduced from $E$ using the classical inferences (here presented in the case of symmetric cryptography).

$$\frac{m \in E}{E \vdash m} \quad \frac{E \vdash m \;\; E \vdash n}{E \vdash \langle m,n \rangle} \quad \frac{E \vdash \langle m,n \rangle}{E \vdash m} \quad \frac{E \vdash \langle m,n \rangle}{E \vdash n} \quad \frac{E \vdash m \;\; E \vdash n}{E \vdash \{m\}_n} \quad \frac{E \vdash \{m\}_n \;\; E \vdash n}{E \vdash m}$$

Given a message $m$, the following sets are defined using an obvious induction.

- $var(m)$ is the set of all variables appearing in $m$.
- $atoms(m)$ is the set of all atoms appearing in $m$.
- $keys(m)$ is the set of all messages appearing as a key in $m$.
- $st(m)$ is the set of all sub-messages of $m$.

The same set names will be used when considering sets of messages, protocols or constraints. As these definitions are obvious, they will not be given here. In this paper, the results will be restricted to the case of symmetric encoding. Extension to public key cryptography is pretty straightforward but will not be presented here to preserve simplicity. When considering the $\vdash$ operator, an immediate need is the related operator that allows to build equations. When writing $E \vdash m$, $E$ and $m$ have to be closed. The next section introduces a new operator $\Vdash$ (the intuitive meaning of which is an equation using $\vdash$). When $E$ and $m$ are not closed, $E \Vdash m$ represents the equation: does there exist a solution (variable assignement) $\sigma$ such that $E\sigma \vdash m\sigma$. Solutions of the equations will be called models of the constraint.

# 3 Main Results

To verify security of a protocol in a bounded number of session, a classical approach is to consider constraints using Dolev-Yao's operator. This has widely been discussed in the literature (see [13], [5] or [10] for example). Here, we will give a classical definition of constraint but we will prove its satisfiability (and NP-completeness of this problem) without making any assumptions on the form of the constraint.

**Definition 2 (Constraints)** *The set $DYC$ of Dolev-Yao constraints is defined by $C$ in the following grammar where $T$ is a finite set of messages and $m$ is a message.*

$$\begin{array}{rcl} C & ::= & C \wedge C | C \vee C | \top | \bot | C_A \\ C_A & ::= & T \Vdash m \end{array}$$

The previous $var$, $atoms$, $keys$ and $st$ notations are extended to constraints.

**Definition 3 (Atomic Keys Models)** *A substitution $\sigma$ is said to be a* model *for a constraint $C$ iff $C\sigma$ is closed and $\sigma \models C$ where $\models$ is defined using the usual inference rules extended by:*

$$\frac{T\sigma \vdash m\sigma}{\sigma \models T \Vdash m}$$

*If $\sigma$ is a model of $C$, then $\sigma$ is an* atomic keys model *of $C$ iff for each $x$ appearing in $C$, $x\sigma$ only use atoms as keys.*

To properly define atomic keys models, we shall define atomic keys messages by:

$$m ::= \langle m, m\rangle|\{m\}_a|\{m\}_x|a|x$$

Then a model $\sigma$ is an atomic keys model iff $x\sigma$ is an atomic keys message.

Another important definition is well-formed constraints. These constraints are well-known in the protocol analysis community because secrecy for a protocol with a bounded number of sessions is equivalent to satisfiability of a well-formed constraint. To introduce these constraints, two hypotheses need to be made whereas there are no assumption on constraints in $DYC$. More precisely, a constraint $C$ in $DYC$ is said to be *well formed* iff none of the environment is empty and for any conjunction $Co$ composing $C$ in disjunctive normal form, the two following conditions hold:

- If $T \Vdash m(U)$ and $T' \Vdash m'(U')$ are in $Co$, then $T \subseteq T'$ or $T' \subseteq T$ (Environment Inclusion).
- If $T \Vdash m(U) \in Co$ and $x \in var(T)$, then there exists $T' \Vdash m'(U') \in Co$ such that $x \in var(m')$ and $T' \subsetneq T$ (Variable Introduction).

In the following, we will only study the case of the atomic keys model. Thus, we will use the general models notation for atomic keys models and all the messages considered afterwards will be atomic keys messages. If constraint $C$ has at least an (atomic keys) model, we will say that $C$ is satisfiable. In the rest of this paper, we will prove that the satisfiability problem is decidable and NP-complete. However, this result is already well-known for well-formed constraints even in the case of composed keys, see for example [16]. Satisfiability of these constraints is exactly equivalent to the secrecy problem. The next theorem is a generalisation of this result to the whole set of constraints.

**Theorem 1** *Satisfiability for constraints in $DYC$ is decidable.*

The former theorem can still be extended. A rather logical (and useful as shown in section 6) extension is to add negation to our constraints.

**Definition 4 (Constraints)** *The set $eDYC$ of extended Dolev-Yao constraints is defined by $C$ in the following grammar where $T$ is a finite set of messages and $m$ is a message.*

$$\begin{array}{rcl} C & ::= & C \wedge C|C \vee C|\top|\bot|C_A \\ C_A & ::= & T \Vdash m|T \nVdash m \end{array}$$

To prove decidability, we will need to put a restriction on negative constraints: their right parts ($m$) must made of closed messages. In practice, this restriction is not a problem as right parts of positive constraints need to be opened (to allow an agent to build a new message), a negative constraint expresses that a message is not known by an agent. Most of the time, this message is a secret and is closed.

**Theorem 2** *Let $C$ be a constraint in $eDYC$ such that $T \nVdash m$ appears in $C$ implies that $m$ is closed. Then satisfiability of $C$ is decidable.*

# 4 Proving Decidability

The goal of this section is to give a sketch of the proof of the first theorem. To keep this readable, some technicals details will be omitted. Demonstration of theorem 2 uses globally the same technique. However, some lemmas are a little more technical in the second case and will not be described here.

**Property 1** *Let $C$ be a constraint in $DYC$ such that $atoms(C)$ is not empty and $C$ is satisfiable, then there exists a model $\sigma$ of $C$ such that:*

$$\bigcup_{x \in var(C)} atoms(x\sigma) \subseteq atoms(C)$$

*So,in particular,*

$$\bigcup_{x \in var(C)} keys(x\sigma) \subseteq atoms(C)$$

The condition that $atoms(C)$ must not be empty is not very restrictive as satisfiability of $C$ is equivalent to satisfiability of $C \wedge a \Vdash a$.

Moreover, checking satisfiability for constraints in $DYC$ is equivalent to checking satisfiability for constraints $C$ in $DYC$ such that $keys(C)$ does not contain any variable.

**Property 2** *Let $C$ be a constraint in $DYC$ such that $atoms(C)$ is not empty, then there exists a subset $\Gamma$ of $DYC$ such that:*

- *For all $C'$ in $\Gamma$, $keys(C') \subseteq A$ and $||C'|| = ||C||$.*
- *$card(\Gamma) \leq (card(Atoms(C)))^n$ where $n = card(keys(C) \cap X$.*
- *$C$ is satisfiable iff there exists $C'$ in $\Gamma$ such that $C'$ is satisfiable.*

Moreover, given a constraint $C$, the set $\Gamma$ can be computed in a time that is polynomial in $||C||$. So without loss of generality we can consider only constraints $C$ such that $keys(C) \subseteq A$.

To check satisfiability, our first step will be to test all possible unifications. Then we will test all the possible orders in which the keys can be compromised. For that purpose, the notation $T \vdash m[U]$ meaning that $m$ is deducible from $T$ using keys in $U$ is introduced.

**Definition 5** *Let $T$ be a finite set of closed messages, $m$ be a closed message, and $U$ be a finite set of atoms. Then $T \vdash m[U]$ is defined using classical Dolev-Yao inferences where the decode rule is replaced by:*

$$\frac{T \vdash \{m\}_u[U] \quad u \in U}{T \vdash m[U]}$$

*The set $DYC$ is trivially extended by adding the atomic constrain $T \Vdash m[U]$ where $T$ and $m$ are potentially not closed but $U$ is still a set of atoms.*

The quantification upon the order in which keys are compromised is expressed in the following property.

**Property 3** *Let $T \Vdash m$ be an atomic constraint, then we have that the next predicate is a tautology.*

$$T \Vdash m \Leftrightarrow \bigcup_{\{a_1,...,a_n\} \subseteq K} T \Vdash a_1[] \wedge T \vdash a_2[a_1] \wedge ... \wedge T \vdash a_n[a_1, ..., a_{n-1}] \wedge T \vdash m[a_1, ..., a_n]$$

*Where $K$ is the set of all keys in $T$ and $m$, namely $keys(T) \cup keys(m)$.*

Thus, using the former property, satisfiability of a constraint in $DYC$ is equivalent to satisfiability of a constraint using only atomic predicates of the form $T \Vdash m[U]$. Such constraints will be called quantified constraints.

**Definition 6** *The rewriting system $\hookrightarrow$ is defined over quantified constraints by:*

$$\begin{array}{rl} T \vdash \langle m, n\rangle[U] & \hookrightarrow T \vdash m[U] \wedge T \vdash n[U] \\ T \vdash \{m\}_n[U] & \hookrightarrow T \vdash m[U] \text{ if } n \in U \end{array}$$

Normal forms for the former rewriting system use only atomic constraints like $T \Vdash a[U]$ where $a$ is an atom, $T \Vdash x[U]$ where $x$ is a variable or $T \Vdash \{m\}_n[U]$ where $n$ is an atom that does not appear in $U$.

**Property 4** *The rewriting system $\hookrightarrow$ over quantified constraints is correct, complete and terminates. So every constraint is equivalent to a constraint in normal form.*

After rewriting the constraint to its normal form, each atomic constraint will be decomposed by looking at which atoms, variables and encoding can be obtained given an environment $T$ and a set of keys $U$ and by looking at which are needed given a message $m$ and a set of keys $U$.

**Definition 7 (Split)** *The $split$ function recursively defined over closed messages by the following lines where $U$ is a finite set of atoms.*

$$\begin{array}{ll} split(a,U) & = \{a\} \\ split(\langle m,n\rangle,U) & = split(m,U) \cup split(n,U) \\ split(\{m\}_u,U) & = split(m,U) \text{ if } u \in U \\ split(\{m\}_v,U) & = \{\{m\}_v\} \text{ if } v \notin U \end{array}$$

Then, the following equivalences are tautologies. And thus, constraints can be transformed to constraints involving $\in$, $\subseteq$ and $split$.

$$T \Vdash a[U] \Leftrightarrow \Big(a \in C \vee \bigvee_{x \in split(T,U)} a \in split(x,U)\Big)$$

$$T \Vdash \{m\}_n[U] \Leftrightarrow \Big(\{m\}_n \in C \vee \bigvee_{x \in split(T,U)} \{m\}_n \in split(x,U)\Big)$$

$$T \Vdash x[U] \Leftrightarrow \Big(split(x,U) \subseteq C \bigcup_{y \in split(T,U)} split(y,U)\Big)$$

Where $C$ contains all the atoms and messages obtained in $split(T,U)$ (not the variables). It is now easy to prove a restricted version of the first theorem.

**Property 5** *Let $C$ be a constraint in $DYC$. If for all atomic constraint $T \Vdash m$ occuring in $C$, $m$ is a closed message, then satisfiability of $C$ is decidable.*

In the rest of this section, we will try to prove the first theorem. This proof will use a finite model argument. First, let us introduce the norm which will allow us to bound the size of the minimal model. Let us call $C$ the constraint we study and $C'$ be the equivalent constraint using the $split$ notation.

**Definition 8** *Let $\{u\}_v$ be a message, $\sigma$ be a substitution, then $\sigma[\{u\}_v\backslash u]$ is the substitution defined by:*

$$x(\sigma[\{u\}_v\backslash u]) = (x\sigma)[\{u\}_v\backslash u]$$

Note that, in general, the following property is false.

$$m(\sigma[\{u\}_v\backslash u]) = (m\sigma)[\{u\}_v\backslash u]$$

This can be noticed by taking $m = \{u\}_v$ and $\sigma = Id$. However, when considering only messages satisfying a specific property, this becomes true. This is expressed in the two following properties.

**Property 6** *Let $m$, $n$ and $\{u\}_v$ be three messages, $U$ be a finite set of atoms such that $m \in split(n,U)$, then:*

$$m[\{u\}_v\backslash u] \in split(n[\{u\}_v\backslash u],U)$$

**Property 7** *Let $m$, $u$ and $u'$ be three messages, $U$ be a finite set of atoms and $\sigma$ be a substitution such that there does not exist any $n$ in $st(m)$ satisfying $n\sigma = u$. Then, the $[]$ operator is associative, i.e.*

$$(m\sigma)[u\backslash u'] = m\big(\sigma[u\backslash u']\big)$$

This will allows us to remove the unused encoding in a model. For that purpose, we will introduce a norm over models that will be minimal when considering models that have only the necessary encodings.

**Definition 9** *The split norm $||_s$ is defined over messages and extended to substitutions by the two equalities for any message $m$ and subsitution $\sigma$.*

$$|m|_s = \textstyle\sum_{n \in split(m,\emptyset)} h(n) \qquad |\sigma|_s = \textstyle\sum_{x \in dom(\sigma)} |x\sigma|_s$$

We can now express our main property. It tells that a minimal (for the former norm) solution is composed of sub-terms of the initial constraint. Knowing that, the main theorem is easy to deduce by trying to pair any sub-terms and checking wether the constraint is satisfied or not (this is of course decidable). As we only have conditions upon split (for the "smallest" variable), we only have to try with one copy of each sub-terms and this can be achieved in a finite time.

**Property 8** *Let $\sigma$ be a model of $C'$ such that $|\sigma|_s$ is minimal. Then for any variable $x$ of $C'$ and for any message $\{u\}_v$ in $split(x\sigma, \emptyset)$, there exists a message $b$ in $st(C')$ such that $\{u\}_v = b\sigma$.*

The proof of the second theorem is quite close to this one. After applying the former rules to the positive ($\Vdash$) part of constraint, we could have to add some encoding layers such that the negative ($\nVdash$) part is satisfied. As the number of needed layers can be bounded using a polynomial in the size of the negative part, satisfiability of our extended constraints remains decidable. This will be described more precisely in the full version of this paper that will appear as a technical report.

# 5 NP-completeness

We will now discuss the complexity of our approach. First, satisfiability of well-formed constraints is NP-hard (see for example [16] or [15]). This is the case for well-formed constraints that only involve $\Vdash$. And as all of these constraints are in the set of constraints that we are studying, our satisfiability problem is NP-hard. To show that this satisfiability problem is in NP and thus NP-complete, we will rely on the results presented in [15]. The authors of this paper proved that given a satisfiable well-formed constraint, there exists a model whose size is polynomial in the size of the constraint. The size used here is the number of different sub-terms. The same result holds for our method. In the former part, we proved that minimal models are made by pairing sub-terms of the initial constraint. Thus, the number of distinct sub-terms for any $x\sigma$ is clearly bounded. Adding the $nVdash$ constraints can add some complexity but only in a limited way (this will be presented in the full version) and the size of any $x\sigma$ remains bounded. That is why, the DAG size (number of distinct sub-terms) of our model remains polynomial in the size of the initial problem. The last thing to notice is that checking whether a closed constraint is closed or not is PTIME. This well-known property comes from MacAllester's theorem (see [12]) and satisfiability of Horn clauses. And so, we can conclude that satisfiability of our constraints is NP-complete.

# 6 Applications and Examples

Proving secrecy when considering a finite number of session only needs a restricted version of our first theorem. However, this theorem allows us two new things. In general, constraints representing a certain protocol property are composed of two subconstraints: the first one represents the unfolding of the protocol (lots of references describe how to compute this constraint, some are given in the second example), the second one represents the property we want to verify. In the secrecy case, this second constraint is of the form $T \Vdash s$ where $s$ is the secret and $T$ the final knowledge of the intruder given by the first constraint. We can now proceed to new verifications, by extending either the *unfolding constraint* or the *property constraint*. Thus, we will give an example of extension for both cases.

- Describe multiple concurrent knowledges. This will be used in example one to define fairness: a protocol will be said *fair* if at the end it is only possible to infer an advantage from $A$'s knowledge if an equivalent advantage is deducible from $B$'s knowledge.

- Describe the intruder's initial knowledge as a variable with some restrictions expressed using $\nVdash$. In example two, we will formalize secrecy of $s$ as the property: if $C$ does not initially know message $s$, he will not know it at the end of the protocol. This kind of property is interesting as it allows to compose protocols, if $P_1$ and $P_2$ are secure for an adapted version of this property, then applying $P_2$ after $P_1$ is still secure.

### 6.1 Fairness Verification

In this section, we describe how our results can be used to check fairness for a contract signing protocol. This extends the property constraint. A contract signing protocol will be said to be fair if after a session between $A$ and $B$, either $A$ has (is able to deduce) the contract signed by $B$ and $B$ has the contract signed by $A$, either none of them has the contract signed by the other one. This is formalized through constraint $C_f$.

$$C_f = (T_A \Vdash sign_B \wedge T_B \nVdash sign_A) \vee (T_A \nVdash sign_A \wedge T_B \Vdash sign_B)$$

$T_A$ and $T_B$ are the respective knowledge of agents $A$ and $B$ after the session. $sign_A$ and $sign_B$ can be for example $\{A, B, c\}_{priv(A)}$ and $\{B, A, c\}_{priv(B)}$ where $c$ is the contract. As soon as this constraint (and the two other constraints: initial situation and protocol run) is not satisfiable, the protocol will be said fair. To show what the whole constraint looks like, let us consider a simple example of contract signing protocol. First, $s_{ABc}$ will be a macro for message $\{A, B, c\}_{priv(A)}$. This protocol is the simple optimistic vote protocol. It has two phases: the principal one is made of exchange of commitments for signature (two first messages) followed by exchange of signatures (two last messages). $0$ is a particular atomic message used separate commitments from signatures.

$$\begin{array}{lcl} A \to B & : & m_1 = \{A, B, c, 0\}_{priv(A)} \\ B \to A & : & m_2 = \{A, B, c, 0\}_{priv(B)} \\ A \to B & : & m_3 = s_{ABc} \\ B \to A & : & m_4 = s_{BAc} \end{array}$$

If after a timeout, $A$ does not receive $S_{BAc}$, he enters the recovery phase in which he ask a trusted third party for $s_{BAc}$ but he has to provide $B$'s commitment ($\{A, B, c, 0\}_{priv(B)}$). Note that this phase cannot be perturbed by any intruder (else it would never be possible to achieve a fair contract signing). Thus, this can be encoded in the final constraint $C'_f$.

$$\begin{array}{rcl} C'_f & = & \big(T_A \Vdash s_{BAc} \Leftrightarrow T_B \Vdash s_{ABc}\big) \\ & \vee & \big(T_A \Vdash \{A, B, c, 0\}_{priv(B)} \Leftrightarrow T_B \Vdash s_{ABc}\big) \\ & \vee & \big(T_A \Vdash s_{BAc} \Leftrightarrow T_B \Vdash \{B, A, c, 0\}_{priv(A)}\big) \\ & \vee & \big(T_A \Vdash \{A, B, c, 0\}_{priv(B)} \Leftrightarrow T_B \Vdash \{B, A, c, 0\}_{priv(A)}\big) \end{array}$$

Then, the constraint we want to verify is $C_u^n \wedge \neg C'_f$ where $C_u^n$ is the unfolding constraint allowing $n$ parallel sessions. We will say that this protocol is fair iff the constraint is not satisfiable.

### 6.2 Initial Intruder Knowledge and Composition

In this example, we will study Lowe's fixed version (presented along with the classical attack on the original protocol in [8]) of Needham-Schroeder protocol (original protocol introduced in [14]). This protocol aims at creating a shared secret between agents $A$ and $B$. It is described in the following rules where $pub(X)$ is the public key of actor $X$ and $priv(X)$ is its private key.

$$\begin{array}{lcl} A \to B & : & \{N_A, A\}_{pub(B)} \\ B \to A & : & \{A, N_A, N_B\}_{pub(A)} \\ A \to B & : & \{N_B\}_{pub(B)} \end{array}$$

Then, the secrecy property will be extended by letting any freedom on the initial knowledge of $C$. This initial knowledge will be noted $X$. Then, secrecy's definition will be that $N_B$ is deducible from $C$ at the end of the protocol only if it was deducible at the beginning. Namely, if we call $\Gamma$ the constraint describing the evolution of $C$'s knowledge during a bounded number of sessions with initial knowledge $X$ and final knowledge $X, T$, then the secrecy will be verified iff the following constraint is not satisfiable.

$$X \nVdash N_B \wedge \Gamma \wedge X, T \Vdash N_B$$

Note that $X$ can contain message $N_B$ encoded with an unknown key. This is not a problem as during this protocol, $C$ does not learn any new key's inverse. However, we may want to express that $N_B$ does not appear at all in $X$. This could have been done by replacing $X \nVdash N_B$ in the former constraint with

$$X, priv(A), priv(B), priv(C) \nVdash N_B$$

For more information on how to compute constraint $\Gamma$, the interested reader should consult [13], [7] or [5]. This kind of constraint can be useful to check protocol composition. Namely, the verified property will be:

- For protocol 1, if the intruder does not initially know any message of $S_1$, he will not know at the end any message of $S'_1$.
- For protocol 2, if the intruder does not initially know any message of $S_2$, he will not know at the end any message of $S'_2$.

Then, if $S_2 \subseteq S'_1$, we can deduce that if initially, the intruder does not know any message in $S_1$, then he will not know any message in $S'_2$ after running protocol 1 then protocol 2 (we consider that only one session for each protocol is authorized).

# 7 Conclusions and Future Work

In this paper, we generalised the classical result of decidability of satisfiability for well-formed constraints to the case of general constraints. This problem is far more complex to solve but remains NP-complete. Then, we showed how this result can be used to verify complex properties over protocols (still when considering a finite number of sessions). Namely, we gave examples for fairness and protocol composition. However, we believe that there are other properties that can be studied using general constraints ([17] gives a good starting point on properties that can express safety of a system). For example, we could verify protocol security when considering that agents can make some checks based on Dolev-Yao theory. Agents could want to check that a given nonce is not deducible from their knowledge. This can be done by simply adding an atomic constraint on this nonce. An other example could be to check that there are no anonymous attack on a protocol: after the attack, it is sure that an honest agent can deduce the real name of the intruder (or any information on the intruder such as its private key). Our decision procedure uses a finite model property. Thus, as this problem is NP-complete, we cannot use directly in a prototype, even on the most simple examples, but we believe that, by refining our technique, it should be possible to find an algorithm that would complete in a reasonable time.

There are lots of possibilites for further work as general constraints are almost unexplored:

- First, there seem to be more properties that can be checked using a general constraint.
- As the result seems quite intuitive, perhaps is it possible to find a more simple decision procedure.
- A very useful extension would be to consider constraints involving a form of strong secrecy such as opacity [11] (does not allow equational theories) or static equivalence introduced in [1] (probably more complex but includes equational theories).
- Eventually, one could try to apply one of the known abstraction methods (see for example [4] or [3]) to generalize this result to the case of unbounded protocols (i.e. verifying property without assuming a finite number of sessions).

## Acknowledgments

This work has been partly supported by the RNTL project PROUVE-03V360.

# References


[1] M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. In *31th International Colloquium on Automata, Languages and Programming (ICALP'04)*, 2004. 7

[2] R. M. Amadio and D. Lugiez. On the reachability problem in cryptographic protocols. In *International Conference on Concurrency Theory*, volume 1877 of *LNCS*, pages 380–394, 2000. 1

[3] B. Blanchet. Abstracting cryptographic protocols by prolog rules. In *International Static Analysis Symposium*, volume 2126 of *LNCS*, pages 433–436, 2001. 7

[4] L. Bozga, Y. Lakhnech, and M. Périn. Abstract interpretation for secrecy using patterns. In *TACAS'03*, volume 2619 of *LNCS*, 2003. 7

[5] Hubert Comon-Lundh and Vitaly Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *Proceedings of the eighteenth annual IEEE symposium on Logic In Computer Science*. IEEE Computer Society Press, June 2003. 1, 3, 6.2

[6] D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983. 1, 2

[7] Florent Jacquemard, Michaël Rusinowitch, and Laurent Vigneron. Compiling and verifying security protocols. In *Logic Programming and Automated Reasoning*, pages 131–160, 2000. 6.2

[8] G. Lowe. An attack on the Needham-Schroeder public-key authentification protocol. *Information Processing Letters*, 56(3):131–133, 1995. 1, 6.2

[9] G. Lowe. Breaking and fixing the Needham-Schroeder Public-Key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 1055 of *LNCS*, pages 147–166, 1996. 1

[10] L. Mazaré. Satisfiability of Dolev-Yao constraints. In *Proc. of ARSPA*. To appear, 2004. 1, 3

[11] L. Mazaré. Using unification for opacity properties. In *Proc. of the Workshop on Issues in the Theory of Security (WITS'04)*. To appear, 2004. 7

[12] David A. McAllester. Automatic recognition of tractability in inference relations. *Journal of the ACM*, 40(2):284–303, 1993. 5

[13] J. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *ACM Conference on Computer and Communications Security*, pages 166–175, 2001. 1, 3, 6.2

[14] R.M. Needham and M.D. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978. 1, 6.2

[15] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *IEEE Computer Security Foundations Workshop*, 2001. 5

[16] Michaël Rusinowitch and Mathieu Turuani. Protocol insecurity with a finite number of sessions and composed keys is np-complete. *Theor. Comput. Sci.*, 299(1-3):451–475, 2003. 3, 5

[17] Peter Y.A. Ryan. Mathematical models of computer security. In Riccardo Focardi and Roberto Gorrieri, editors, *Foundations of Security Analysis and Design: Tutorial Lectures*, volume 2171 of *LNCS*. Springer-Verlag, 2001. 7