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

Satisfiability of Dolev-Yao Constraints

Laurent Mazaré

Report n^o 22

October 27, 2004

Satisfiability of Dolev-Yao Constraints

Laurent Mazaré

October 27, 2004

Abstract

The secrecy problem, under certain hypothesis, is equivalent to satisfiability of constraints involving Dolev-Yao's operator $\vdash$. Making some restrictions over these constraints, their satisfiability has been proven to be NP-complete. However, to check opacity [11] or some modified versions of secrecy, there is a need to find similar results for a larger class of predicates. This paper starts to extend this decidability result to more general constraints allowing in particular inequalities and gives a simple decision procedure based on a rewriting system.

Keywords: Security, Formal Verification, Dolev-Yao Constraints, Rewriting Systems, Opacity.

Reviewers:

Notes:

How to cite this report:

@techreport { verimag-TR-2004-22,
title = { Satisfiability of Dolev-Yao Constraints},
authors = { Laurent Mazaré},
institution = { Verimag Technical Report },
number = {22},
year = { 2004},
note = { }
}

1 Introduction

During the last decade, verification of security protocols has been widely investigated. The majority of the studies focussed on demonstrating secrecy properties using formal methods (see for example [4], [6], [5] or [9]). These methods have lead to effective algorithms and so to concrete tools for verifying secrecy like those proposed by the EVA project [3] or the Avispa project [2]. The main result is that considering an active intruder and a bounded number of sessions in Dolev-Yao model (see [8]), secrecy is decidable and is an NP-complete problem. One possible way to demonstrate this result is to use constraints based on Dolev-Yao's operator $\vdash$. This has been done for example in [12] or in [14]. A Dolev-Yao constraint will be a conjunction of atomic constraints of the form $E \Vdash m$, where E is an environment (i.e. a finite set of messages), m is a message and both of them are potentially not closed. Thus, in this case, the secrecy problem is equivalent to satisfiability of a given constraint.

In this paper, after giving a formal definition of Dolev-Yao constraints, we will investigate their satisfiability. In a first part, we will restrict ourselves to "well-formed" constraints. These constraints will form an extension of the well-known set of constraints which satisfiability is exactly equivalent to secrecy in Dolev-Yao's model with a bounded number of sessions. This extension will add the possibility to use the $\neq$ operator. Thus, protocols which check that a received variable is not a given one (for example, a protocol verifying that a nonce has not already been used before) could be verified by checking satisfiability of this kind of constraint. Using a method based on term rewriting, satisfiability of such constraints will be proved to be decidable and we will provide a concrete decision algorithm. Moreover, one of the hypothesis over well-formed constraints could be removed. This will define quasi well-formed constraints which satisfiability will be proved decidable too. In addition to protocols checking inequalities, there are two main motivations when extending classical Dolev-Yao constraints. The first one, leading to the definition of well-formed, is to study opacity [11] instead of secrecy in the case of active intruders. Opacity in this case will be equivalent to satisfiability of a well-formed constraint. The criterion defining "quasi well-formed" constraints is useful when studying opacity too, so that we could model the unfolding of two parallel sessions. An other use of these constraints could be to model attacks performed by two distinct intruders that are not allowed to communicate. Application of these constraints to opacity will not be entirely detailed here and will be the object of a future paper.

The remainder of this paper is organized as follows. In section 2, we will briefly recall the usual definitions leading to Dolev-Yao constraints and proper definitions for well-formed and quasi well-formed will be given. Then, in section 3, we will use a rewrite system to prove that satisfiability of well-formed constraints is decidable. Section 4 will extend this result to the case of quasi well-formed constraint. In section 5, we will quickly explain why satisfiability of our constraints is NP-complete. Section 6 will show how these results could be applied to check opacity with an active intruder. And eventually, section 7 will conclude this paper.

2 Dolev-Yao Constraints

Let A, X and F be three 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 F is the set of functions (written f, g,...).

Definition 2.1 *Let Σ be the signature $A \cup \{pair, encrypt\} \cup F$ where pair and encrypt are two binary functions. The atomic messages are supposed constant functions and each function f has a fixed arity $ar(f)$. Then a* message *is a first order term over Σ 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 of a message m could be easily defined recursively and will be noted $|m|$. The set of variables used in a message m is noted $var(m)$, this is not the usual definition of var as we will consider that $f(...)$ is "atomic":

$$var(a) \quad = \emptyset$$

$$\begin{array}{ll} var(x) & = \{x\} \\ var(f(...)) & = \emptyset \\ var(\langle m, n \rangle) & = var(m) \cup var(n) \\ var(\{m\}_n) & = var(m) \cup var(n) \end{array}$$

The substitutions σ from X to $T(\Sigma, X)$ are defined as usual. Application of σ to message m will be noted $m\sigma$ (instead of $\sigma(m)$). If σ is defined by $x\sigma = n$ and $y\sigma = y$ for any other variables y, then we could write $m[x \backslash n]$ instead of $m\sigma$. The domain of a substitution σ 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 using n times the same message t. 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 [8]. The intruder controls the network: he could intercept any message, forge messages using its initial knowledge and previously intercepted messages, and send these messages to other agents usurping an agent's identity. To forge new messages, the intruder uses Dolev-Yao theory $\vdash$. If E is a finite set of messages (usually called an *environment*) and m is a message, $E \vdash m$ means that m could be deduced from E using the classical inferences. In this paper, we only consider symmetric cryptography. However, all of our results would still hold when considering public key cryptography but we will keep this hypothesis for simplicity's sake.

Dolev-Yao constraints are predicates built using classical logic operators and a new tertiary operator noted $\Vdash$. This operator is used in atomic predicates like $T \Vdash m[U]$ where T (environment), m (message) and U (environment) could be non closed. This predicates' intuition is that there exists a proof of $T \vdash m[U]$ which is defined as the classical $\vdash$ except for the decode rule.

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

The only keys allowed to decode a message are the keys in U and these keys are not to be proven deducible.

Definition 2.2 (Constraints) *Dolev-Yao* constraints *are defined according to the following grammar:*

$$C ::= \bot | \top | C \vee C | C \wedge C | C_A$$

$$C_A ::= T \Vdash m[U] | m \neq n$$

Where T and U are finite sets of messages, m and n are messages.

For a constraint C, the sets T appearing as left part of an atomic constraint $T \Vdash m[U]$ in C are called environments of C. The classical notion of model is used for operators like $\vee$ or $\neq$. We will extend it by saying that σ is a model of $T \Vdash m[U]$ written $\sigma \models T \Vdash m[U]$ iff $T\sigma$, $m\sigma$ and $U\sigma$ are closed and there exists a proof of $T\sigma \vdash m\sigma[U\sigma]$.

Definition 2.3 (Model) *A substitution σ is a model for constraint C iff $C\sigma$ is closed and $\sigma \models C$ where $\sigma \models .$ is the smallest predicate verifying the classical inferences for $\top$, $\wedge$ and $\vee$, and the two following inferences:*

$$\frac{m\sigma \neq n\sigma}{\sigma \models m \neq n}$$

$$\frac{T\sigma \vdash_T m\sigma[U\sigma]}{\sigma \models T \Vdash m[U]}$$

Using standard boolean rules, any constraint could be transformed to $C = \bigvee_i Co_i$ where $Co_i = \bigwedge C_A$. The Co_i constraints are called the conjunctions composing C. We will usually suppose that our constraints follow this form as it is needed to check that they are well-formed or quasi well-formed.

Definition 2.4 (Well-Formed Constraint) *A constraint C is said to be* well formed *iff none of the environment is empty and for any conjunction Co composing C, 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 $x \in var(m')$, $U' \subseteq U$ and $T' \subsetneq T$ (Variable Introduction).*

A constraint that only satisfies the Variable Introduction requirement and such that there exists a same closed message m in all its environment will be said quasi well-formed*.*

It is easy to remark that for conjunctions in a well-formed constraint, there exists a constraint $T_x \Vdash m_x[U_x]$ in Co such that T_x is minimal and $x \in var(m_x)$. This notation T_x will be used in the following. An immediate property about T_x is that $x \notin var(T_x)$. Moreover, we could notice that any well-formed constraint is quasi well-formed.

As we have now proper definitions for Dolev-Yao constraints, we will give a very classical example of constraint. To achieve this, it is practical to be able to formalize the classical Dolev-Yao constraint: message m is deducible from the set of message T written $T \Vdash m$. This constraint could be described using constraint with $[]$ by quantifying on the order upon which keys are compromised. m will be deducible iff there exist k_1,...,k_n distinct keys in T or m such that k_1 is deducible without any decoding, k_2 deducible using k_1 only and so on. At the end, m needs to be deducible from T using only keys k_1 to k_n.

Proposition 2.1 *Let m be a message and T a finite set of messages. Then, for any substitution σ, the following equivalence holds.*

$$T\sigma \vdash m\sigma \Leftrightarrow \sigma \models \bigvee_{k_1,...,k_n \in keys(T,m)} \big(T \Vdash k_1[] \wedge T \Vdash k_2[k_1] \wedge ... \wedge T \Vdash m[k_1,...,k_n]\big)$$

This property could be extended to predicates with several $\Vdash$ operators. If environments of the predicate are ordered, it is possible to assume that keys compromised for an environment T are compromised for T' verifying $T \subseteq T'$. Let us call the environments $T_1 \subseteq T_2 \subseteq ...T_n$. Thus, we quantify on the previous order that gives us k_1,...,k_α and for each environment T_j on the value of i_j such that in T_j, keys k_1 to k_{i_j} are compromised (and so, we only have to consider $i_1 \leq i_2 \leq ... \leq i_n$).

Proposition 2.2 *For i between 1 and n, let m_i be a message and T_i a finite set of messages. If $\bigwedge_{1 \leq i \leq n} T_i\sigma \Vdash m_i$ is well-formed and $T_1 \subseteq T_2 \subseteq ...T_n$, then for any substitution σ, the following equivalence holds.*

$$\begin{aligned}\bigwedge_{1\leq i\leq n} T_i\sigma \vdash m_i\sigma \Leftrightarrow \sigma \models \bigvee_{\substack{k_1,...,k_\alpha \in keys(T,m)\\ 1\leq i_1 \leq i_2 \leq ... \leq i_n \leq n}} & \big(T_1 \Vdash k_1[] \wedge ... \wedge T_1 \Vdash m_1[k_1,...,k_{i_1}]\big)\\ & \wedge\big(T_2 \Vdash k_{i_1+1}[k_1,...,k_{i_1}] \wedge ... \wedge T_2 \Vdash m_2[k_1,...,k_{i_2}]\big)\\ & \wedge ...\\ & \wedge\big(T_n \Vdash k_{i_{n-1}+1}[k_1,...,k_{i_{n-1}}] \wedge ... \wedge T_n \Vdash m_n[k_1,...,k_{i_n}]\big)\end{aligned}$$

This property gives the following result: secrecy is equivalent to satisfiability of a well-formed Dolev-Yao constraint.

An Example: Needham-Schroeder Constraint

Secrecy with a bounded number of sessions in Dolev-Yao model could be linked to satisfiability of a well-formed constraint. This is explained in [7] for example. The usual example of attack that could be discovered using formal methods is an attack on Needham-Schroeder protocol [13] found by Lowe in 1995 [10]. The constraint NS which satisfiability is equivalent to the existence of this attack is given below. As it describes an attack in Dolev-Yao model, the environments describe the knowledge of the intruder and thus verifies Environment Inclusion and Variable Introduction. That is why, this constraint is well-formed and its satisfiability could be verified. The two sessions considered here occur between A and the intruder C and between C usurping A's identity and B. To keep it as simple as possible, this example uses public key cryptography. Let E be the initial knowledge of the intruder C, $E = \{A, B, C, K_A, K_B, K_C, K_C^{-1}\}$.

$$\begin{aligned} NS = \quad & E, \{N_C, A\}_{K_C} \Vdash \{x, A\}_{K_B}\\ \wedge \quad & E, \{N_C, A\}_{K_C}, \{x, N_B\}_{K_A} \Vdash \{N_C, y\}_{K_A}\\ \wedge \quad & E, \{N_C, A\}_{K_C}, \{x, N_B\}_{K_A}, \{y\}_{K_C} \Vdash N_B\end{aligned}$$

3 Decidability for Well-Formed Constraints

In this section, we will provide an original decision procedure for satisfiability of well-formed constraints. For such constraints using only $\Vdash$, it is well known that the satisfiability problem is NP-complete: see for example [14], [12] or an extension to XOR in [7]. Our decision procedure will work using a rewriting system over constraints. We will have to prove that our system terminates, that satisfiability for our normal forms is decidable and eventually that our system is correct and complete, i.e. models of a constraint are exactly the models of this constraint once rewritten. Knowing that, we are able to provide a decision procedure for satisfiability which is to rewrite the constraint up to its normal form and then to check satisfiability on this normal form. Compared to other proofs of decidability of satisfiability for this kind of constraints (that are easy to find in the literature, see [12] as a starting point), our main objective is to provide a very simple, yet extensible, decision procedure.

Definition of the rewriting system needs some explanation. Our rewriting system is triggered by a condition. As we will check for termination without considering that condition, this will not cause any problem. The first step in our decision procedure is to test all possible unifications among sub-terms of the different environments and messages. This is done using a large quantification ($\vee$ operator). After that step, two different non-closed messages (that are sub-messages of the original constraint) are supposed to be instantiated with different closed messages (i.e. they cannot be unified). Using that, an atom a is deducible from T (using U) iff a appears in T (protected only by keys occurring in U). The same kind of idea is used for $\{\}$ whereas the case of $\langle\rangle$ is trivial.

Definition 3.1 *The rewriting system $\rightarrow$ is defined over well-formed constraints by the three following rules that will only apply if for all $x \in var(T)$, $T_x \Vdash x[U_x]$ occurs in the same conjunction with $U_x \subseteq U$.*

$$\begin{aligned}
T \Vdash a[U] &\rightarrow \top && (1)\\
T \Vdash a[U] &\rightarrow \bot && (2)\\
T \Vdash f(m_1, ..., m_n)[U] &\rightarrow \top && (3)\\
T \Vdash f(m_1, ..., m_n)[U] &\rightarrow \bot && (4)\\
T \Vdash \langle m, n\rangle[U] &\rightarrow T \Vdash m[U] \wedge T \Vdash n[U] && (5)\\
T \Vdash \{m\}_n[U] &\rightarrow \top && (6)\\
T \Vdash \{m\}_n[U] &\rightarrow T \Vdash m[U] \wedge T \Vdash n[U] && (7)
\end{aligned}$$

- *Rule 1 is applied iff a occurs in T protected only by keys that appear in U, else rule 2 is applied.*
- *Rule 3 is applied iff $f(m_1, ..., m_n)$ occurs in T protected only by keys that appear in U, else rule 4 is applied. Thus, rule 1 and 2 are useless if we consider atoms as constant functions.*
- *Rule 5 is the intuitive rule for pairs.*
- *Rule 6 is applied iff $\{m\}_n$ occurs in T protected only by keys that appear in U (this is the case where $\{m\}_n$ could be obtained using decompositions), else rule 7 is applied (in the other case, $\{m\}_n$ is obtained with a composition operation).*

The condition expressed over T aims to force the execution order to follow $\subseteq$. Therefore if $T_1 \subseteq T_2$, then the pattern $T_1 \Vdash m_1$ will be entirely rewritten before processing $T_2 \Vdash m_2$.

Last remark, these rules are easy to extend to the case of public key cryptography. In this case, in the condition of application of the different rules, "protected only by keys that appear in U" is to be replaced by protected only by keys which inverses appear in U.

The first thing to check is that after applying $\rightarrow$ to a well-formed constraint, the result remains well-formed. This is expressed by the following property.

Proposition 3.1 *If C and C' are constraints such that $C \rightarrow C'$ and C is well-formed, then C' is well-formed.*

The second thing is to prove that our rewriting system terminates. Then, it will be possible to transform any constraint to a constraint in normal form that has exactly the same models. This is easy to show as when rewriting, right members of each constraint are discomposed or the constraint is rewritten to $\bot$ or $\top$. Thus, by taking a simple measure over constraint $s(C)$ defined by:

$$s(T_1 \Vdash m_1[U_1] \wedge ... \wedge T_n \Vdash m_n[U_n]) = \sum_{i=1}^{n} s(m_i)$$

This gives us $s(\bot) = s(\top) = 0$. And s is recursively defined over messages by:

$$\begin{array}{rcl} s(a) & = & 1 \\ s(f(...)) & = & 1 \\ s(\langle m_1, m_2 \rangle) & = & s(m_1) + s(m_2) + 1 \\ s(\{m_1\}_{m_2}) & = & s(m_1) + s(m_2) + 1 \end{array}$$

Using an order based on this measure, the following property is immediate.

Proposition 3.2 *The rewriting system $\rightarrow$ terminates.*

After transforming a constraint to its normal form, we want to check that both have the same sets of models. Formally, if $C \rightarrow C'$ then for any substitution σ, σ is a model of C iff σ is a model of C'. This will prove that if the normal form is satisfiable then, the original constraint is satisfiable too and that the reciprocal is true.

Proposition 3.3 *The rewriting system $\rightarrow$ is correct and complete for well-formed constraints.*

Note that, to prove this result, the Variable Introduction hypothesis is used. Else, rules would not be correct and complete anymore. For example, if we consider the following constraint:

$$a, \{b\}_a \Vdash x[a] \wedge a, \{b\}_a, x \Vdash b[]$$

This constraint would be rewritten to $\bot$ as b does not appear in $a, \{b\}_a, x$ unprotected. But, by using $x = b$, we know that this constraint is satisfiable. That is why, most of the rules are only correct and complete if the Variable Introduction hypothesis is respected.

Eventually, the last step is to show that satisfiability for our normal forms is easily decidable. First, normal forms of well-formed constraints are well-formed constraints. They have the form:

$$\bigvee \Big((T \Vdash x[U]) \wedge \big(\bigwedge m \neq n \big) \Big)$$

At this point, every normal form that is well-formed is satisfiable if the lowest environment T_0 is non empty (this is always true by definition of well-formed) and if the conjunction of inequalities is satisfiable. To prove that, let t be a message in T_0 (t always exists as T_0 is not empty). As the environment T_0 is closed (this could be seen by applying the Variable Introduction hypothesis to T_0), t is closed. Then, for every variable in an atomic constraint, we will use a message of the form t^n. It is clear that the conjunction of predicates using $\Vdash$ is satisfied. And by changing values of the different n, if the $\neq$ part is satisfiable, we will satisfy it by giving to n a value lower than the number of inequalities plus the number of variables. The intuition is that, given a number n, if a conjunction $m_1 \neq n_2$ is not satisfied, then it satisfied for any other number $n' \neq n$.

We will use the σ^k notation which signification $x\sigma^k$ is the tuple composed by k times $x\sigma$.

$$x\sigma^k = (x\sigma)^k = \langle x\sigma, ..., x\sigma \rangle$$

Proposition 3.4 (Solving Inequalities) *Let P be the constraint $m_1 \neq n_1 \wedge ... \wedge m_j \neq n_j$ where m_i and n_i are messages. If P is satisfiable, then for any substitution σ such that $P\sigma$ is closed and taking distinct values for any variables in $var(P)$, there exists an integer k such that $k \leq j + 1$ and σ^k is a model of P.*

Proof: The proof use the following lemma: let m and n be two messages, σ be a substitution such that $m\sigma$ and $n\sigma$ are closed, and that $x\sigma = y\sigma \Rightarrow x = y$. If there exists two different integers i and j such that $m\sigma^i = n\sigma^i$ and $m\sigma^j = n\sigma^j$, then we have $m = n$. ■ This kind of property is very general and could be applied to other operators instead of $=$. For example, the same thing holds for the $\sim$ operator introduced in [11] and this will allow to prove decidability of satisfiability for a similar extension of the predicates defined in this paper. This decidability (and NP-completeness) will prove that the opacity problem when considering an active intruder (Dolev-Yao's model) and a finite number of session is NP-complete. Eventually, satisfiability is equivalent to satisfiability of the $\neq$ part. This last satisfiability is easy to check using for example negation of this conjunction. This gives a constraint using only equalities. Then, by applying classical unification, it is possible to check that this final constraint is always satisfied (hence is rewritten to $\top$ using unification) or not (and thus that its opposite is satisfiable or not).

Finally, the main result of this paper is a consequence of all the former properties.

Theorem 3.1 (Main Theorem) *Satisfiability of well-formed constraints is decidable.*

This result is not really new, it could already be found in [1] for instance. However, the demonstration used here seems to be easier to adapt to other cases. In particular, having first order symbols like f, this theorem will apply quite directly to show decidability of opacity in a next section.

An Example: Needham-Schroeder Constraint

The former rewriting system has been implemented in a prototype using ocaml. However, it has been adapted so that it does not have to test all the possible unifications. This makes the method easier to understand but is very negative for performance. This is the reason why our algorithm does only make unification when it need them. For example, the rules 3 and 4 would be replaced by a rule trying all the possible unification of $f(..)$ and the reachable patterns starting with f in the environment. The advantage of this method is that by looking at the unification requested by the rewriting system, we obtain values for the different parameters and this algorithm does not have to look to all possible unifications but only to "plausible ones". In the formal proof and other works, the general first idea is to guess all possible equalities between sub-messages and to get rid of them using the most great unifier (see [7] on how to adapt this to exclusive or). It has been tested on the Needham-Schroeder constraint. Its result is of course that this constraint is satisfiable and the only possible variables' values are: $x = N_C$ and $y = N_B$. During this check, five possible unifications have been tested. Three of them involved two messages that could not be unified and so only two cases were explored. One quickly lead to satisfiability whereas the other one was not satisfiable. Adding to the initial constraint the inequality $y \neq N_B$, we immediately have that the constraint is not satisfiable anymore. This gives us a possible fix for this protocol: if agent A was able to determine the origin of nonce N_B (by replacing it by a pair nonce, identity of the nonce's creator), then it would not be possible to perform this attack anymore. This is the idea behind Lowe fixed version of this protocol appearing in [10]. This fix adds the agent's identity to the second message, hence allows to check if the identity of the nonce's creator is valid.

4 Quasi Well-Formed Constraints

The previous theorem could be extended to quasi well-formed constraints. There are at least two possible ways to demonstrate this.

- The first solution is to notice that the only place where we use the first hypothesis is to prove that satisfiability for normal forms is decidable. This is the reason why we added the last hypothesis for quasi well-formed constraints, thus there exists a closed message common to all the environments. The modifications to perform are to demonstrate the modified property:

  Proposition 4.1 *If C and C' are constraints such that $C \to C'$ and C is quasi well-formed, then C' is quasi well-formed.*

Termination is proved for any kind of constraints so this will not be a problem. Correctness and completeness could be adapted too.

Proposition 4.2 *The rewriting system $\rightarrow$ is correct and complete for quasi well-formed constraints.*

Eventually, when considering the normal form, the proof does not change: using tuples of message m, it is possible to verify satisfiability as message m is deducible from any of the involved environments.

- The other way is to remark that a quasi well-formed conjunction is equivalent to a conjunction of well-formed conjunctions. In the initial conjunction, the atomic constraint $T \Vdash m[U]$ could be replaced by $T \Vdash m[U] \wedge \bigwedge T' \Vdash m'[U']$ where the conjunctions introducing the different variables in T are added and those introducing variables in the different m' (to formalize that, we would need a fix point). As the number of conjunctions in the initial constraint is finite, the results are well-formed and finite conjunctions. The rest of this demonstration will be shorter as it will use the results of the former section. This is why, these conjunctions could be rewritten. And so, we obtain an equivalent normal form which is a quasi well-formed constraint. As we have seen before, satisfiability of this constraint is rather easy to check. There remains the inequalities part to check but this could still be done using the method described above.

Both of these proof skeletons allow us to prove the following theorem.

Theorem 4.1 *Satisfiability of quasi well-formed constraints is decidable.*

More than satisfiability, we provide a decision procedure that has been implemented and seems to perform quite well.

An Example: Needham-Schroeder Constraint

As said earlier, our main motivation for this extension is its application to opacity. However, there are other possible uses for it. The Environment Inclusion hypothesis held because the environments represented the intruder's knowledge. As the intruder could only learn new messages, this set could only grow and the hypothesis is not restrictive. However, by suppressing this hypothesis, it is now possible to model distinct knowledge with the different environments. Concretely, this extension easily allows to model two intruders that are not able to communicate between each other. For example, let us consider a modified version of Dolev-Yao attack. There are two intruders C_1 and C_2 such that there is a session between A and C_1 and another one between C_2 (usurping A's identity) and B. We want to know if it is possible for one of the intruder to deduce one of the secret N_A or N_B. Let T_0 be the initial knowledge of our intruders $\{A, B, C_1, C_2, K_{C_1}, K_{C_2}, K_{C_1}^{-1}, K_{C_2}^{-1}\}$. Then, the constraint becomes:

$$\begin{array}{lll} & & T_0 \Vdash \{A, x\}_{K_B} \\ \wedge & & T_0, \{A, N_A\}_{K_B} \Vdash \{x', y'\}_{K_A} \\ \wedge & (& T_0, \{x, N_B\}_{K_A} \Vdash N_B \\ & \vee & T_0, \{A, N_A\}_{K_B}, \{y'\}_{K_{C_1}} \Vdash N_A) \end{array}$$

This constraint appears to be quasi well-formed but is not well-formed. By using our decision procedure, we know that this constraint is not satisfiable. Thus, the two intruders need to communicate so that the classical attack over Dolev-Yao's protocol could be performed.

5 NP-Completeness

We will now discuss the complexity of our approach. First, satisfiability of well-formed constraints is NP-hard (see for example [15] or [14]). 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's 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 [14]. The authors of this paper proved that given a satisfiable well-formed constraint without inequalities, there exists a model which 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. Its result is a substitution σ such that the size of $x\sigma$ is polynomial in the size of the constraint. When solving the inequalities part, the size remains polynomial: let $|m|_{DAG}$ be the number of distinct sub-terms in m. Then it is easy to prove that $|m^n|_{DAG} = |m|_{DAG} + n - 1$. So in the worst case, we only added the number of inequalities and variables to the size of our model. Thus the DAG size of our model remains polynomial in the size of the initial problem. That is why, we could conclude that satisfiability of well-formed constraints is NP-complete.

For quasi well-formed constraints, the exact same decision algorithm could be used. This proves that satisfiability for these constraints is also NP-complete.

6 Application to Opacity

In this section, we will give a quick look at how the former method could be used to prove that opacity is decidable when considering Dolev-Yao model. In other papers, we only studied the case of a passive intruder. Here, we will start to extend decidability results to the case of active intruders.

First, let us give an extended definition of $\vdash$ to the case of two environments and two messages $E, E \vdash n, n'$. This means that n and n' are deducible from E and E' by making the same operations on both environments. For example, $\{a, b\}, \{c, d\} \vdash \langle a, b\rangle, \langle c, d\rangle$ is true as both messages are obtained by pairing the first and second messages in the environments whereas $\{a, b\}, \{c, d\} \vdash \langle a, a\rangle, \langle c, d\rangle$ is false as the messages could not be obtained by the same operations. $E, E \vdash n, n'$ is defined on closed messages n, n' and closed environments E and E' by:

$$\frac{}{\{n_1, ..., n_k\}, \{n'_1, ..., n'_k\} \vdash n_i, n'_i} \qquad \frac{E, E' \vdash n_1, n'_1 \quad E, E' \vdash n_2, n'_2}{E, E' \vdash \langle n_1, n_2\rangle, \langle n'_1, n'_2\rangle}$$

$$\frac{E, E' \vdash \langle n_1, n_2\rangle, \langle n'_1, n'_2\rangle}{E, E' \vdash n_1, n'_1} \qquad \frac{E, E' \vdash \langle n_1, n_2\rangle, \langle n'_1, n'_2\rangle}{E, E' \vdash n_2, n'_2}$$

$$\frac{E, E' \vdash \{n_1\}_{n_2}, \{n'_1\}_{n'_2} \quad E, E' \vdash n_2, n'_2}{E, E' \vdash n_1, n'_1} \qquad \frac{E, E' \vdash n_1, n'_1 \quad E, E' \vdash n_2, n'_2}{E, E' \vdash \{n_1\}_{n_2}, \{n'_1\}_{n'_2}}$$

Note that E and E' must have the same length (in fact, as these sets will be supposed similar, this will not be a problem). Moreover, as we want to perform the same operations on both environments, we need the environments to be ordered. They could be represented using lists.

The definition of opacity (given for example in [11]) is that an intruder is not able to distinguish a run where the property is satisfied from a run where it is not. To distinguish two messages, the intruder could decompose them, according to his knowledge but if he does not know the key k for example, he will not be able to make the difference between two different messages encoded by this key k. However, in this section, we will only consider a very simplified version of opacity. Let us consider a finite protocol P with a unique parameter v (called the vote) that could only take two values yes (protocol P_y) or no (protocol P_n). We will say that the value of v is not opaque (in fact, the property $v = yes$ is not opaque) iff the intruder is able to produce yes and no by doing the same operations on both protocols. The intruder has to perform the exact same sequence of actions: for example if in the first protocol, he sends a constant A then receives a message that is a pair and eventually sends the left part of the pair, he will have to do the same actions for the second protocol. This will be equivalent to satisfiability of the following constraint:

$$T_0, T_0 \Vdash m_0, m'_0 \wedge T_0; n_0, T_0; n'_0 \Vdash m_1, m'_1 \wedge ... \wedge T_n, T'_n \Vdash yes, no$$

Where $T_i = T_0; m_0; ...; m_{i-1}$. The messages m_i and m'_i are produced by the intruder and sent to protocols P_y and P_n. Then, the protocols replies are messages n_i and n'_i. Both sessions are carried in the same manner and we ask that at the end, the intruder is able to tell on which session v was equal to yes and on which session v was equal to no.

To link our new constraint to Dolev-Yao constraints, let us introduce an explicit binary function f for our extended $\vdash$ operator. The constraint could be written

$$f(T_0, T_0) \Vdash f(m_0, m'_0) \wedge f(T_0; n_0, T_0; n'_0) \Vdash f(m_1, m'_1) \wedge ... \wedge f(T_n, T'_n) \Vdash f(yes, no)$$

Next, we will distribute the f function inside our constraint by using the following $distrib$ operator recursively defined by:

$$distrib_f(f(\langle m, n\rangle, \langle m', n'\rangle)) = \langle distrib_f(f(m, m')), distrib_f(f(n, n'))\rangle$$

$$distrib_f(f(\{m\}_n, \{m'\}_{n'})) = \{distrib_f(f(m, m'))\}_{distrib_f(f(n,n'))}$$

Else we have $distrib_f(m) = m$.

Applying the $distrib_f$ function creates a constraint equivalent to the original one but which is a Dolev-Yao constraint. This is formalized in the following property which could be proved using induction over the proofs' structures. If E and E' are two environments, n and n' are two messages such that f does not appear in E, E', n or n', then the following equivalence is true:

$$E, E' \Vdash n, n' \Leftrightarrow distrib_f(f(E, E')) \Vdash distrib_f(f(n, n'))$$

So, opacity is equivalent to satisfiability of a Dolev-Yao constraint. Moreover, this constraint is, by construction, well-formed. Thus, opacity as defined before is decidable and we have an NP-complete algorithm to check satisfiability. However, we only studied a very restrictive version of opacity but we believe that this result could be extended to the general opacity problem.

7 Conclusion

In this paper, we gave a simple decision procedure to prove satisfiability of extended versions of usual Dolev-Yao constraints. Using a rewriting system, this algorithm is easy to implement. Moreover, we believe that this method could easily be used for other constraints derived from Dolev-Yao's one. This has been done through a prototype that performed well on simple examples. An immediate application of these new predicates is to check the secret in presence of two intruders that are not allowed to communicate. Then quasi well-formed constraint will be useful as they allow to describe two "branches" of environments that do not have to be ordered by set inclusion. Each of these branches will be described by a well-formed constraint. Further works include of course linking more precisely these constraints to the opacity problem in the case of an active intruder. However, there remains an open question: is satisfiability of general Dolev-Yao constraints decidable ? We did not manage to prove that in the general (no hypothesis at all) case.

Acknowledgment

The author wishes to thank Romain Janvier for his reading of a preliminary version of this paper and Hubert Comon for his DEA course "Cryptographic Protocols' Proofs": the decision method used here has been inspired by the content of this course during year 2003.

References

[1] 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. 3

[2] The Avispa Project. http://www.avispa-project.org/, 1999. 1

[3] L. Bozga, Y. Lakhnech, and M. Périn. Abstract interpretation for secrecy using patterns. Technical report, EVA : http://www-eva.imag.fr/, 2002. 1

[4] E.M. Clarke, S. Jha, and W. Marrero. Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In *IFIP Working Conference on Programming Concepts and Methods*, 1998. 1

[5] H. Comon-Lundh. and V. Cortier. Security properties: Two agents are sufficient. Technical report, LSV, 2002. 1

[6] H. Comon-Lundh and V. Cortier. New decidability results for fragments of first-order logic and application to cryptographic protocols. In *14th Int. Conf. Rewriting Techniques and Applications (RTA'2003)*, volume 2706 of *LNCS*, 2003. 1

[7] 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. 2, 3, 3

[8] 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

[9] Jean Goubault-Larrecq. A method for automatic cryptographic protocol verification. In *International Workshop on Formal Methods for Parallel Programming: Theory and Applications*, volume 1800 of *LNCS*, 2000. 1

[10] G. Lowe. An attack on the Needham-Schroeder public-key authentification protocol. *Information Processing Letters*, 56(3):131–133, 1995. 2, 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. (document), 1, 3, 6

[12] Jonathan K. Millen and Vitaly Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *ACM Conference on Computer and Communications Security*, pages 166–175, 2001. 1, 3

[13] 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. 2

[14] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *IEEE Computer Security Foundations Workshop*, 2001. 1, 3, 5

[15] 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. 5