New decidability results for fragments of first-order logic and application to cryptographic protocols

Hubert Comon-Lundh and Véronique Cortier **

Laboratoire Spécification et Vérification, CNRS
Ecole Normale Supérieure de Cachan,
{comon,cortier}@lsv.ens-cachan.fr

Abstract. We consider a new extension of the Skolem class for first-order logic and prove its decidability by resolution techniques. We then extend this class including the built-in equational theory of exclusive or. Again, we prove the decidability of the class by resolution techniques.
Considering such fragments of first-order logic is motivated by the automatic verification of cryptographic protocols, for an arbitrary number of sessions; the first-order formalization is an approximation of the set of possible traces, for instance relaxing the nonce freshness assumption.
As a consequence, we get some new decidability results for the verification of cryptographic protocols with exclusive or.

1 Introduction

The verification of cryptographic protocols deserved a lot of attention in the past few years, because of the huge application domain of secure communications via public channels. In this context, the full automation of verification tools is important because, in general, the same protocol appears in multiple contexts in a slightly altered form; each instance has to be verified since it is never clear whether a small modification has an impact on the security property or not.

Such verification problems are typically relevant to model checking: given a protocol P and a security property ϕ, does P satisfy ϕ ? And indeed, model-checking tools have been used successfully to find some attacks (the most famous one is due to G. Lowe [21]). However, *proving* the correctness of a protocol is much harder for several reasons. First of all, we must be very precise on the semantics of protocols and security properties; there is still today a debate on these aspects. Next, whatever model of the protocols is chosen, it is both infinite in depth (traces have an unbounded length, because arbitrarily many instances of the protocol, also called *sessions* can be involved) and infinitely branching (depending on an attacker's input). Finally most of the protocols use *nonces*, which are supposed to be randomly generated numbers. As demonstrated by several authors [10, 16, 1], this yields undecidability of model checking, even in very restricted cases.

There are two possible research directions, which proved to be relevant: either consider a bounded number of sessions, which is sufficient to restore decidability as shown e.g. in [26], or to consider an abstraction of the model, which may be sufficient for proving the protocol correct, but may also output "dummy" attacks. This line of research is followed by e.g. [28, 3, 5] and that is also what we consider in this paper.

** Partially supported by INRIA project SECSI and RNTL project EVA.

A first abstraction consists in replacing nonces with terms depending on the context. That is what is done in all abstraction techniques we know. Then, protocols can be modeled within first-order logic [4, 28, 3, 8] and the satisfaction of most popular security properties such as secrecy and authentication reduces to satisfiability of a set of clauses (see e.g. [13]).

However, even for protocols without nonces (or considering the above abstraction), the verification of simple properties remains undecidable (e.g. [10]). On the other hand, experiments using general purpose automatic theorem provers such as SPASS, show that, most of the time, the proof search terminates. Trying to explain this phenomenon reduces to finding decidable fragments of first-order logic in which most of the protocols can be expressed (with the above sketched abstraction for nonces). For instance, we have shown in [9] that, for a significant class of protocols, the confidentiality problem can be reduced to the solvability of a class of set constraints with equality, itself shown to be decidable using tree automata with memory.

On the other hand, all automated verification results rely, so far, on the *perfect cryptography assumption*, which, roughly, says that the message algebra is a free term algebra. Such an hypothesis is too strong since many protocols use cryptographic primitives which do have algebraic properties. A typical example is the exclusive or. As an example, Bull's authentication protocol was proved to be secure with the perfect cryptography assumption, while there is an attack when the algebraic properties of `xor` are considered [24, 27]. Up to our knowledge, there are very few results on verification of cryptographic protocols with `xor`: the only other result is a proof of decidability in case of a bounded number of sessions [14, 6].

The work described in this paper has two motivations: on one hand to explain the reasons why first-order theorem provers often terminate on protocol verification, on the other hand study the extensions considering the algebraic properties of exclusive or and an unbounded number of sessions.

We already realized in [9] that one reason for undecidability, which does not occur in practice, is the agents ability to copy and locally modify two distinct pieces of a message, hence enabling the simulation of two counters machines. That is why we will consider here protocols in which an agent can copy "blindly" at most one piece of the message he receives. "Blindly" has the following (informal) meaning: protocols consist in message exchange between (say) two agents. Upon receiving some message m, agent A breaks m into pieces, decrypting what she can decrypt. Each piece she gets is either known to her (it can be a public value such as an agent name or a nonce she generated earlier,...) or something she does not know (a cyphertext that she cannot decrypt, a nonce generated by the other party). Such data are represented by variables: an intruder could for instance replace them by arbitrary values. If the message that A is supposed to send makes use of such variables, we say that she copies "blindly" their content.

Such an hypothesis on the uniqueness of blind copies seems relevant since most of the protocols of [7] falls into the class. On the model side, this corresponds to first-order clauses involving at most one variable. We will give more details in section 4. This is why we consider here the fragment of first-order logic consisting in clauses which contain at most one variable. Actually, we have to consider a larger fragment, because we need to express for instance intruder capabilities, which do not fall in this category. More precisely, we consider a clausal fragment in which every clause C either contains at most one variable or is such that every subterm t is either ground, a variable, or contains all variables of the clause. We prove that this fragment of first-order logic is decidable in section 2, using ordered resolution techniques. This fragment is actually similar to the extension $\mathcal{S}^+$ of the Skolem class as defined in [18]. Still, it is different since for instance, we will allow literals $P(x)$ in multiple variables clauses. We also allow arbitrary ground literals.

Our main result is however the extension of this decidable class, considering the algebraic properties of `xor`: we prove in section 3 the decidability of fragment of first-order logic, which contains both the above class and the equality axioms for `xor`. Therefore we design a set of deduction rules and an ordered strategy, which we prove complete and terminating.

One difficulty here is that there is almost no ordering on terms with variables, which is stable by substitution. Hence we use an ordering which is stable by "non-collapsing" substitutions, restoring completeness using a rule similar to narrowing. Another difficulty is to control the number of variables occurring in clauses, which we need for termination. To this end, we impose stronger restrictions on resolution and factorization, restoring completeness by *extensions*.

Termination relies on technical results on unification with associativity, commutativity, identity and nilpotence (ACUN) and free symbols (which is known to be in NP [23]), typically concerning the sizes of mgus. Finally, completeness is obtained via classical semantic trees methods.

In section 4, we show how the previous results apply to the verification of cryptographic protocols, hence providing the first decidability result for an unbounded number of sessions, and considering the algebraic properties of `xor`. We illustrate the result, proving the correctness of a simple protocol. Our result is disjoint from [14, 6]. On one hand, we consider an unbounded number of sessions (while it is bounded in [14, 6]). On the other hand, in [14, 6], there is no hypothesis on the number of blind copies and the result relies on constraint solving techniques and locality properties in the spirit of [22]. Note that first-order logic is not relevant for a bounded number of sessions since it would require to give a bound on the number of times a clause is used (e.g. using rigid variables).

Due to space limitations, many proofs are only given in a technical report [12].

2 A simple decidable fragment of first-order logic

2.1 Definitions

Let $\mathcal{F}$ be a finite set of function symbols, $\mathcal{V}$ a set of variables and $\mathcal{P}$ a finite set of unary predicates. For every clause C (resp. every term u), $V(C)$ (resp. $V(u)$) is the set of variables of C (resp. of u). $V(u,v)$ denotes $V(u) \cup V(v)$. If L is a positive literal, we write $L' = \pm L$ for $L' \in \{L, \neg L\}$. If u and t are terms of $\mathcal{T}(\mathcal{F} \cup \mathcal{V})$ and if x is in $V(u)$, $u[t/x]$ is the term u where every occurrence of x has been replaced by t.

Definition 1. *A clause set S belongs to the class $\mathcal{C}$ if for every clause C in S, either C contains at most one variable or, for every literal L in C :*

1. *either $L = \pm P(x_i)$ for some $P \in \mathcal{P}$;*
2. *or $L = \pm P(u[f(x_1, \ldots, x_n)/y])$ for some $P \in \mathcal{P}$ and some $f \in \mathcal{F}$ such that $\{x_1, \ldots, x_n\} = V(C)$ and u is some term of $\mathcal{T}(\mathcal{F} \cup \{y\})$.*

We may write that a clause C is in $\mathcal{C}$ instead of saying that the set $\{C\}$ is in $\mathcal{C}$ to express that C is a clause of the form described above.

This class is incomparable with the class $\mathcal{S}^+$ as described in [18]. We believe that, with some additional technical details, we can extend our result so that our class contains $\mathcal{S}^+$. This is however not relevant for our application nor for the extension of the next section. As examples of sets of clauses that can be expressed in this class, let us mention for instance two-way alternating tree automata (see e.g. [11], chapter 7); since the emptiness of the automaton can also be expressed as a clause in the class, the decidability of $\mathcal{C}$ implies the emptiness decidability for two-way alternating automata.

If $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{V})$, $|t|$ is the *depth* of t (maximal size of its positions). For $x \in \mathcal{V}$, $|t|_x$ is the maximal depth of an occurrence of x in t. By convention, it is 0 if $x \notin V(t)$. $|.|$ and $|.|_x$ are extended to literals by $|P(t)| = |t|$ and $|P(t)|_x = |t|_x$.

We will prove the decision result by ordered resolution, using the ordering derived from the following definition.

Definition 2. *Let A, B be two literals.*

$$A < B \quad \textit{if} \quad |A| < |B| \quad \textit{and if} \quad \forall x \in V(A) \cup V(B) \quad |A|_x < |B|_x.$$

$A \leq B$ if $A < B$ or $A = B$.

Note that when $A \leq B$, we have in particular that $V(A) \subseteq V(B)$.

A sufficient condition for completeness of ordered resolution is to use a *liftable* ordering [20, 18], also called *stable* ordering in [19].

Definition 3 (liftability). *An ordering $\leq_{\mathcal{R}}$ is* liftable *if, for all atoms A, B and all substitutions θ, $A \leq_{\mathcal{R}} B$ implies $A\theta \leq_{\mathcal{R}} B\theta$.*

Proposition 1. *$\leq$ is a liftable ordering.*

2.2 Decidability result

Theorem 1 (decidability of $\mathcal{C}$). *Let S be a finite set of clauses such that S belongs to $\mathcal{C}$. The satisfiability of S is decidable.*

Proof sketch

We use splitting (see e.g. [29]), ordered factorization and ordered binary resolution (see e.g. [2]), w.r.t. the partial ordering defined above, using a classical redundancy criterion [2], also called *a posteriori criterion* in e.g. [18]; we apply resolution on two clauses C_1 and C_2 only if no atom of the resolvent is greater than the resolved atom. Such an ordered strategy is complete [2, 18]. It only remains to show termination.

First, after splitting, we only generate clauses in $\mathcal{C}$. Then, define $\|C\|$ as the maximal depth of its literals and let N be the maximum of $\|C\|$ for clauses in S. We show that, for every generated clause C' (after splitting), either $\|C'\| \leq N$, or else C' is ground and $\|C'\| \leq 2 \times N$. This is a consequence of simple lemmas on the unifiers of terms containing at most one variable, for instance:

Lemma 1. *Let u, v be two terms such that $V(u) = \{x\}$ and $V(v) = \{y\}$. If they are unifiable with mgu σ, then either $u\sigma$ is ground and $|u\sigma| \leq 2 \times \max(|u|, |v|)$ or else $|u\sigma| \leq \max(|u|, |v|)$.*

Then, thanks to the ordered strategy, which only unifies maximal literals, we get the bound on $\|C\|$ for the generated clauses C.

The termination follows from the fact that there are only finitely many clauses in $\mathcal{C}$ whose size is bounded and to which splitting does not apply.

3 An extension including the exclusive or

We are going to extend the result of the previous section, including algebraic properties of a binary symbol. We will proceed as in the previous section: we define an ordering and consider an ordered deduction strategy. There are however several additional problems:

- for termination purposes, we need to keep control of the number of variables in each clause. Therefore we restrict the applicability of e.g. resolution and restore completeness, adding extension rules.
- it is a hard task to find an ordering which is both liftable and compatible with the theory of `xor`. We use an ordering, which is stable only by substitutions which do not introduce any simplification. Considering substitutions which introduce redexes is handled separately as a pre-processing step
- for termination purposes, we need to control the size of unifiers, relying on the particular equational theory we consider. We will see the analogs of lemma 1 in section 3.2.

3.1 Definition of the class of clauses

In this part, we extend our class of clauses $\mathcal{C}$ to a class of clauses $\mathcal{C}^{\oplus}$ including the algebraic properties of $\oplus$ which are described in figure 1. The two last equations

$$\begin{array}{cc} x \oplus (y \oplus z) = (x \oplus y) \oplus x & x \oplus y = y \oplus x \\ x \oplus 0 = x & x \oplus x = 0 \end{array}$$

Fig. 1. Equational theory of the `xor` function symbol

can be oriented from left to right and we get a convergent rewrite system modulo associativity and commutativity, provided we add the extended rule $y \oplus x \oplus x \to y$ (see e.g. [15] for definitions). For any term t in $T(\mathcal{F} \cup \{\oplus\} \cup \mathcal{V})$, we write $t\downarrow$ as its normal form w.r.t. these rules.

Formally, we consider a finite set $\mathcal{F}$ of function symbols containing the constant symbol 0, a set $\mathcal{V}$ of variables and a finite set $\mathcal{P}$ of predicate symbols. $\mathcal{C}^{\oplus}$ is a class of clauses extending $\mathcal{C}$, described below.

Definition 4. *A clause set S belongs to $\mathcal{C}^{\oplus}$ if for every clause C in S, either C contains at most one variable or for every literal L in C :*

1. *either $L = \pm P(x_i)$ for some $P \in \mathcal{P}$;*
2. *or $L = \pm P(u[f(x_1, \ldots, x_n)/y])$ for some $P \in \mathcal{P}$ and $f \in \mathcal{F}$ such that $\{x_1, \ldots, x_n\} = V(C)$ and u is some term of $\mathcal{T}(\mathcal{F} \cup \{\oplus\} \cup \{y\})$;*
3. *or $C = \neg P(x_1) \vee \neg P(x_2) \vee P(x_1 \oplus x_2)$ for some $P \in \mathcal{P}$.*

Remarks : Note that for the second type of clauses ($\pm P(u[f(x_1, \ldots, x_n)/y])$), f is forbidden to be $\oplus$ but $\oplus$ may occur in u.
We will see in section 4 that the special clause $C_0 \stackrel{\text{def}}{=} \neg I(x) \vee \neg I(y) \vee I(x \oplus y)$ is used to encode the ability of the intruder to compute the `xor` of two terms.

In the following, S_0 denotes the set of clauses in S which are of the third type in the above definition.

From now on, $=$ denotes the equality between terms (or literals) modulo the (AC) properties of `xor` while $=_{\oplus}$ denotes the equality between terms (or literals) modulo the whole equational theory of `xor`.

Following the AC property of $\oplus$, we assume terms are written in flatten form: $\oplus$ may be considered as a variadic function symbol. Subterms are defined accordingly. For instance the subterms of $f(a \oplus b \oplus g(x))$ are $f(a \oplus b \oplus g(x)), a \oplus b \oplus g(x), a, b, g(x), x$. $a \oplus b$ and $a \oplus g(x)$ are not subterms.

We extend $|.|$ and $|.|_x$ on terms of $\mathcal{T}(\mathcal{F} \cup \{\oplus\} \cup \mathcal{V})$. Informally, since $\oplus$ is now a variadic symbol, it may in particular have only one argument, in which case we don't write it, hence don't count it in the size of the terms; that is why the following measure computes the length of the longest path, not taking $\oplus$ into account.

Definition 5. $\|.\|$ *is defined inductively by:*
1. $\|a\| = 1$ *if* $a \in \mathcal{V}$ *or if* a *is a constant symbol of* $\mathcal{F}$*;*
2. $\|f(t_1, \ldots, t_k)\| = 1 + \max_{1 \leq i \leq k} \|t_i\|$ *for* $f \in \mathcal{F}$*;*
3. $\|t_1 \oplus \cdots \oplus t_n\| = \max_{1 \leq i \leq n} \|t_i\|$ *if the head symbol of each* t_i *is not* $\oplus$.

Then $|t|$ *is defined as* $\|t \downarrow\|$. $|.|_x$ *is defined in the same way except for point 1:* $\|x\|_x = 1$ *and* $\|a\|_x = 0$ *if* $a \in \mathcal{V}\_\{x\}$ *or if* a *is a constant symbol of* $\mathcal{F}$. *This is also extended to clauses by:* $|C| = \max_{L \in C} |L|$ *and* $|\pm P(t)| = |t|$.

Then the definition of $\leq$ (definition 2) is unchanged. However, $\leq$ is no longer a liftable ordering.

Example 1. Let $L_1 = P(a \oplus b)$, $L_2 = P(f(x \oplus a) \oplus f(b \oplus a))$ and $x\theta = b$. Then $L_1 < L_2$ but $L_1\theta = P(a \oplus b) \not< L_2\theta = P(0)$.

Actually, there are few orderings which are liftable and compatible with the rules of figure 1. For instance there is no such ordering which contains the subterm ordering: we would have $x \oplus f(a) > a$, but then $(a \oplus f(a)) \oplus f(a) > a$! That is why we introduce the notion of *narrow-liftable* ordering and *collapse-free* substitution.

Definition 6. *A substitution* σ *is* normalized *if, for every variable* x, $x\sigma$ *is in normal form. A substitution* σ *is* collapse-free *w.r.t. a set of terms* S *if, for every* $t \in S$, $t\sigma \downarrow = t \downarrow \sigma$.

We will write NS for the set of normalized substitutions and $CF(C_1, \ldots, C_n)$ for the set of collapse-free substitutions w.r.t. the set of subterms occurring in the clauses $C_1, \ldots, C_n$, which are supposed to be irreducible.

Definition 7. *An ordering* $\leq_{\mathcal{R}}$ *is* narrow-liftable *if, for every atom* A, B *and every substitution* θ*, which is collapse-free w.r.t.* B*,* $A <_{\mathcal{R}} B$ *implies* $A\theta <_{\mathcal{R}} B\theta$.

Proposition 2. $\leq$ *is a narrow-liftable ordering on literals of clauses of* $\mathcal{C}^{\oplus}$.

3.2 Some useful results on unification

It is well known that unifiability modulo the theory of figure 1 is NP-complete in the presence of free function symbols and that unification is finitary [23]. We need however finer results (the analogs of lemma 1) to control the size of terms.

Lemma 2. *If* $u \neq_{\oplus} v$ *and* $Var(u, v) \subseteq \{x\}$*. Then either* u *and* v *are not unifiable (modulo the rules of figure 1) or else any (normalized) unifier* $\sigma = \{x \mapsto w\}$ *is such that* w *is a ground term and either* w *is a subterm of* $u \oplus v$ *or else* $w = w_1 \oplus w_2$ *is a normal form such that* w_1 *and* $x \oplus w_2$ *are subterms of* u *or* v*. Moreover,* $|x\sigma| \leq \max\{|u|, |v|\}$.

Note that $|u\sigma|$ may be strictly greater than $|u|$ and $|v|$.

Example 2. Let us consider $u = h^2(x) \oplus h^2(a) \oplus x$ and $v = h^2(x)$. The most general unifier of u and v is $\sigma(x) = h^2(a)$ and $u\sigma = h^4(a)$.

Lemma 3. *If $Var(u) \cap Var(v) = \emptyset$ and $Var(u) \subseteq \{x\}, Var(v) \subseteq \{y\}$, $u \neq_{\oplus} v$, then either u and v are not unifiable (modulo the rules of figure 1) or else every most general unifier θ of u, v is, up to variable renaming, such that:*

- *either there are ground subterms $w_1, \ldots, w_k$ of u, v such that $x\theta = w_1 \oplus \ldots \oplus w_k$ (resp. $y\theta = w_1 \oplus \ldots \oplus w_k$) and $y\theta$ is ground (resp. $x\theta$ is ground)*
- *or $x\theta = z \oplus t_1 \oplus \cdots \oplus t_k$, $y\theta = (u_1 \oplus \cdots \oplus u_n \oplus w_1 \oplus \cdots \oplus w_m)\theta$, where the t_i's and the w_i's are ground subterms of u, v, $n \geq 1$ and the u_i's are non-ground subterms of u or the converse, exchanging the roles of x and y (resp.of u and v).*

Example 3. $g(a \oplus f(y \oplus f(a)), f^n(y)) = g(x, f^n(x))$ has a solution $x = y = a \oplus f(a)$. Instantiating the original terms, their measure is growing.

A similar technique allows us to conclude when u or v is equal to $u'[x \to f(x_1, \ldots, x_n)]$. See the appendix for details.

We design the ordering $\leq$ in such a way that it is stable by collapse-free substitutions. Therefore, we have to show how it is possible to consider only such substitutions. A general result in [14] allows to focus on collapse-free substitutions, roughly guessing the shared parts and performing possible simplification beforehand. That is also what we (roughly) do here. However, we need also to control the size of the resulting clauses, taking advantage of our additional assumptions.

Lemma 4. *For every clause $C \in \mathcal{C}^{\oplus}$, there is a finite set of clauses $\{C_1, \ldots, C_n\}$, denoted by* $\mathsf{narrow}(C)$*, such that :*

$$\{C\sigma\downarrow \mid V(C\sigma) = \emptyset, \sigma \in NS\} = \bigcup_{i=1}^{n} \{C_i\sigma \mid V(C_i\sigma) = \emptyset, \sigma \in CF(C_1, \ldots, C_n)\}$$

Moreover, if $C \notin S_0$, every C_i falls in one of the three following cases: $C_i = C$, or C_i is ground and $|C_i| \leq 2 \times |C|$, or $V(C) = \{x\}$ and $C_i = C\{x \mapsto y \oplus t_i\}\downarrow$ for some sum t_i of ground subterms of C.

3.3 The decidability result

The goal of this section is to prove the following (main) result:

Theorem 2 (decidability of $\mathcal{C}^{\oplus}$). *Let S be a finite set of clauses such that S belongs to $\mathcal{C}^{\oplus}$. The satisfiability of S is decidable.*

Thanks to lemma 4 we can restrict our attention to collapse-free substitutions, provided that we apply the rule which replaces C with the set of clauses C_i constructed in lemma 4. This rule is called the *narrowing rule*.

But restricting ourself to "collapse-free" ordered resolution is still not sufficient to ensure termination. Indeed, only the repetitive resolution of renamings of C_0 with themselves yields an infinite set of clauses. That is why we will disallow resolution steps between clauses in S_0, restoring completeness using *extensions*. The situation is similar to the transitivity rule for which a special inference rule is designed: ordered chaining [2]. Extensions aim at inferring $P(s \oplus u) \vee C \vee D$ from $P(s \oplus t) \vee C$ and $P(t \oplus u) \vee D$ when t is maximal among s, t, u. They consist in a short-cut of two resolution steps.

Example 4. Let us consider $C_1 = \neg P(s(h) \oplus x)$ and $C_2 = P(s(h(y)) \oplus s(y))$. Applying the rule Extension 1, we get the clause $C_3 = \neg P(h(y) \oplus s(y))$:

$$\frac{\neg P(s(x) \oplus x) \quad P(s(h(y)) \oplus s(y))}{\neg P(h(y) \oplus s(y))} \quad \text{with } t_1 = s(x),\ u_1 = s(h(y)) \text{ and } \theta = \{x \mapsto h(y)\}.$$

C_3 could have been obtained by the two successive binary resolution rules if we had allowed for resolution with clauses in S_0.

$$\frac{\neg P(s(x) \oplus x) \quad \neg P(z) \vee \neg P(z') \vee P(z \oplus z')}{\neg P(s(h(y)) \oplus s(y)) \vee \neg P(h(y) \oplus s(y))} \quad \text{with } x = h(y),\ z = s(h(y)) \oplus s(y) \text{ and } z' = h(y) \oplus s(y)$$

$$\text{and} \quad \frac{\neg P(s(h(y)) \oplus s(y)) \vee \neg P(h(y) \oplus s(y)) \quad P(s(h(y)) \oplus s(y))}{\neg P(h(y) \oplus s(y))}.$$

Deduction rules are displayed in Figure 2. As usual (see e.g. [18]), repeatedly applying the deduction rules of figure 2 together with a splitting rule yields a set of sets of clauses: $\mathcal{E}_0 = \{S\}$ and $\mathcal{E}_{i+1}$ is obtained:

- either by replacing $S_j \in \mathcal{E}_i$ by $S_j \cup \{C\}$ if C can be inferred from S_i using a rule of figure 2,
- or by replacing some $S_j \cup \{C \vee C'\} \in \mathcal{E}_i$ with two sets $S_j \cup \{C\}$ and $S_j \cup \{C'\}$ if $Var(C) \cap Var(C') = \emptyset$.

We also remove redundant clauses at each step. For our purpose, it is sufficient to remove clauses $L \vee L \vee C$ when $L \vee C$ is in the set of clauses.

Lemma 5 (Correctness). *The narrowing rule and the deduction rules of figure 2 are correct (the set of models of one of the clause sets in $\mathcal{E}_i$ is the same as the set of models of one of the clause sets in $\mathcal{E}_{i+1}$) and, if every clause set in $\mathcal{E}_i$ is in $\mathcal{C}^{\oplus}$, then every clause set in $\mathcal{E}_{i+1}$ is in $\mathcal{C}^{\oplus}$.*

Lemma 6 (Termination). *The sequence $\mathcal{E}_i$ is finite when starting from $\mathcal{E}_0 = \{S\}$ and $S \in \mathcal{C}^{\oplus}$.*

Proof. (sketch) The sequence $\mathcal{E}_i$ is finite iff applying the rules of figure 2 together with the rule $C \vee C' \to C$ when $Var(C) \cap Var(C') = \emptyset$ terminates when starting from S.

We are going to give an upper bound on the size of a clause C in a set of $\mathcal{E}_i$. Let T the set of ground subterms of S and $N \stackrel{\text{def}}{=} \max_{L \in C, C \in S} |L|$. We show by induction on i that, for every clause C of a set of $\mathcal{E}_i$, either C is ground and $|C| \leq 2N$, or C is not ground and $|C| \leq N$, or C contains exactly one variable x and $|C\{x \mapsto x \oplus t\}| \leq N$ for some $t \in T$.

To prove this, we investigate all possible cases (each deduction rule) and we rely on lemma 2 and 3 (detailed proof in [12]).

Then, we show that there are only finitely many ground clauses such that $|C| \leq 2N$ (this relies on the nilpotence of $\oplus$) and only finitely many non-ground clauses in $\mathcal{C}^{\oplus}$ such that $|C| \leq N$ or $|C\{x \mapsto x \oplus t\}| \leq N$ for some $t \in T$.

Binary Resolution

$$\frac{\neg P(t) \vee C \quad P(u) \vee C'}{C\sigma \vee C'\sigma}$$

If σ is collapse-free w.r.t. literals in C, C', $\sigma \in mgu(t,u)$, $P(t)\sigma \not< (C \vee C')\sigma$.

Factorization

$$\frac{L_1 \vee L_2 \vee C}{(L_1 \vee C)\sigma}$$

If $\sigma \in mgu(L_1, L_2)$, σ is collapse-free w.r.t. literals in the clause, and $L_1\sigma \not< C\sigma$.

Narrowing

$$\frac{C}{(C\sigma)\downarrow}$$

If $(C\sigma)\downarrow \in \mathsf{narrow}(C)$.

Explosion

$$\frac{P(t \oplus u) \vee C}{(P(t \oplus u) \vee C)\sigma}$$

If t is ground, $u\sigma$ is ground and $u\sigma < t$.

Extension 1

$$\frac{\neg P(t) \vee C \quad P(u_1 \oplus u_2) \vee C'}{(C \vee C' \vee \neg P(t_2 \oplus u_2))\theta} \text{ If } \begin{cases} P(x \oplus y) \vee \neg P(x) \vee \neg P(y) \in S_0 \\ t = t_1 \oplus t_2 (\text{or } t = t_1 \text{ and } t_2 = 0) \\ Var(t) = Var(t_1), \theta \in mgu(t_1, u_1) \\ \theta \text{ collapse-free w.r.t. } t, u_1 \oplus u_2, t_2 \oplus u_2 \\ (C \vee C' \vee \neg P(t_2 \oplus u_2))\theta \not> t_1\theta. \end{cases}$$

Extension 2

$$\frac{P(t) \vee C \quad P(u_1 \oplus u_2) \vee C'}{(C \vee C' \vee P(t_2 \oplus u_2))\theta} \text{ If } \begin{cases} \neg P(x) \vee \neg P(y) \vee P(x \oplus y) \in S_0 \\ t = t_1 \oplus t_2 (\text{or } t = t_1 \text{ and } t_2 = 0) \\ Var(t) = Var(t_1), \theta \in mgu(t_1, u_1) \\ \theta \text{ collapse-free w.r.t. } t, u_1 \oplus u_2, t_2 \oplus u_2 \\ (C \vee C' \vee \neg P(t_2 \oplus u_2))\theta \not> t_1\theta. \end{cases}$$

All rules only apply to non-splittable clauses, not belonging to S_0.

Fig. 2. Deduction rules.

Thanks to lemma 6, the sequence $\mathcal{E}_i$ is finite. We let $\mathcal{E}^*(S)$ be its limit, when starting from $\mathcal{E}_0 = \{S\}$, $S \in \mathcal{C}^{\oplus}$.

Lemma 7 (Completeness). *Let $S \in \mathcal{C}^{\oplus}$. S is unsatisfiable if and only if for every set $S' \in \mathcal{E}^*(S)$, $\perp \in S'$.*

Proof (sketch): our deduction system is correct, thus if $\perp \in S^*$ for every set $S^* \in \mathcal{E}^*(S)$ then S is not satisfiable.

Assume S is not satisfiable and assume $\perp \notin S^* \in \mathcal{E}^*$.

We extend our partial ordering $\leq$ on literals to a total ordering $\tilde{\leq}$ on ground literals in the following way.

Let $\leq$ be any total ordering on the predicates $\mathcal{P}$ and on the function symbols $\mathcal{F}$. We extend $\leq$ on $\mathcal{F} \cup \{\oplus\}$ by $\oplus < f$ for all $f \in \mathcal{F}$. We then let $m(t)$ be the triple $(|t|, top(t), Sub(t))$ where $top(t)$ is the top symbol of t and $Sub(t)$ are its immediate (strict) subterms. For two ground terms in normal form, we let $t \tilde{<} t'$ if $m(t) < m(t')$ where the triples are lexicographically ordered, using the ordering on $\mathcal{F}$ for the second component, the lexicographic extension of $\tilde{\leq}$ on the subterms when the top symbol is not $\oplus$ and the multiset extension of $\tilde{\leq}$ otherwise.

Let L_1, L_2 be two ground positive literals : $L_1 = P_1(t_1)$ and $L_2 = P_2(t_2)$. Then $L_1 \tilde{<} L_2$ if either $t_1 \tilde{<} t_2$; or $t_1 = t_2$ and $P_1 < P_2$.

By definition, $\tilde{<}$ extends $<$ and $\tilde{\leq}$ is a total ordering. The Herbrand base is totally ordered accordingly as well as partial interpretations. As usual in semantic tree methods, since S is unsatisfiable, by correctness S^* is unsatisfiable, hence its semantic tree is finite (the set of partial interpretations which do not falsify a clause of S^*).

Then we consider a partial interpretation $\mathcal{I}$ whose two extensions to $P_1(v)$ falsify a clause of S^* and which is minimal w.r.t. the lexicographic ordering on partial interpretations. (This is a "leftmost" node whose two sons are failure nodes in the semantic tree). The lexicographic ordering on partial interpretations is defined by $I >_{lex} J$ if, when $P(u)$ is the maximal element of the Herbrand base such that I and J coincide on literals strictly smaller than $P(u)$, $I(P(u)) = 1$, $J(P(u)) = 0$.

By factorization we may assume that the two clauses C_1, C_2 falsified by the two extensions of $\mathcal{I}$ are such that $P_1(v) \vee C_1' = C_1\sigma_1$ and $\neg P_1(v) \vee C_2' = C_2\sigma_2$ for some $C_1, C_2 \in S^* \cup S_0$ such that $C_1', C_2' < P_1(v)$. By narrowing, we may assume that σ_1 is collapse-free w.r.t. C_1 and σ_2 is collapse-free w.r.t. C_2. We distinguish four cases: either $C_1, C_2 \in S_0$, or $C_1 \in S_0$ and $C_2 \notin S_0$, $C_1 \notin S_0$ and $C_2 \in S_0$ or else $C_1, C_2 \notin S_0$.

These cases are described in more details in the appendix, we sketch here the reasons why it works:

Case $C_1, C_2 \in S_0$: We prove directly that there is another smaller clause falsified by $\mathcal{I}$, simply recombining the terms in the right order. This corresponds to the uselessness of extensions of extensions.

Case $C_1 \in S_0, C_2 \notin S_0$: Let $C_1\sigma_1 = \neg P_0(x)\sigma_1 \vee \neg P_0(y)\sigma_1 \vee P_0(x \oplus y)\sigma_1$, $x\sigma_1 = v_1$, $y\sigma_1 = v_2$, and $(x\sigma_1 \oplus y\sigma_1) \downarrow = v$. There exist v_1', v_2', v' such that $v = v_1' \oplus v_2'$, $v_1 = v_1' \oplus v'$ and $v_2 = v_2' \oplus v'$ without any collapse or $v = v_1 \oplus v_2$ without any collapse. We only consider the first case since the second one is similar. By hypothesis, $v_1 \tilde{<} v, v_2 \tilde{<} v$ and therefore $\mathcal{I}(P_0(v_1)) = \mathcal{I}(P_0(v_2)) = 1$. Assume w.l.o.g that $P_0(v_1) \tilde{\leq} P_0(v_2)$. Now, by minimality of the interpretation $\mathcal{I}$ (w.r.t.

lexicographic ordering), the partial interpretation $\mathcal{J}$ which coincides with $\mathcal{I}$ on literals strictly smaller than $P_0(v_1)$ and such that $\mathcal{J}(P_0(v_1)) = 0$ falsifies a clause $C_3 = P_0(u) \vee C'$ of S^*. We consider again two cases, depending on whether this clause is in S_0 or not.

Assume $C_3 \notin S_0$ and that no factorization can be applied. Also, by narrowing, $C_3\sigma_3$ does not contain any redex and $v_1 = u\sigma_3$. Moreover, $P_0(v_1)$ is maximal in $C_3\sigma_3$. We are going to show that we can apply **Extension 1** (possibly after **Explosion**) to C_2 and C_3 yielding a clause falsified by $\mathcal{I}$. We let $C_2 = \neg P_0(t) \vee C$. We have $v = t\sigma_2$ and σ_2 is collapse-free thus $t = t_1 \oplus t_2$ such that $t_1\sigma_2 = v_1'$ and $t_2\sigma_2 = v_2'$. In the same way, $u = u_1 \oplus u_2$ such that $u_1\sigma_3 = v_1'$ and $u_2\sigma_3 = v'$. This means in particular that t_1, u_1 are unifiable. By **Explosion**, we may assume that $V(t) = V(t_1)$ and, by lemma 3, that there is a $\theta \in mgu(t_1, u_1)$ such that $\sigma_2 \uplus \sigma_3 = \theta\theta'$. Moreover, let w be the maximal strict direct subterm of v. Since $v_2 \widetilde{<} v_1 \widetilde{<} v$, w is a strict direct subterm of v_1' thus $v_2 = v_2' \oplus v' \widetilde{<} v_1'$. The inequality $v_2 = v_2' \oplus v' \widetilde{<} v_1'$ gives $t_2\sigma_2 \oplus u_2\sigma_3 \widetilde{<} t_1\sigma_2$, hence $(t_2 \oplus u_2)\theta\theta' \widetilde{<} t_1\sigma_2$. It follows that $(t_2 \oplus u_2)\theta \not> t_1$. In addition, θ is collapse-free w.r.t. t, $t_1 \oplus t_2$, $t_2 \oplus u_2$ and the clauses C and C'. Then, we can apply **Extension 1** and there is a clause $(C \vee C' \vee \neg P_0(t_2 \oplus u_2))\theta$, which is already falsified by $\mathcal{I}$.

The case $C_3 \in S_0$ yields to the previous case where $C_1, C_2 \in S_0$.

Case $C_1 \notin S_0, C_2 \in S_0$: this case is symmetric to the previous one, replacing **Explosion 1** with **Explosion 2**.

Case $C_1, C_2 \notin S_0$. We simply use **Resolution**; there is a smaller clause which is already falsified by $\mathcal{I}$.

4 Application to cryptographic protocols

We assume the reader familiar with the notion of agent, nonce, intruder, ... In this paragraph, we show how security properties for a class of protocols can be expressed as the satisfiability of a set of clauses $S \in \mathcal{C}^{\oplus}$. We also propose a simple (new) cryptographic protocol, which we prove correct using our technique.

We have presented in [13] a clausal model of cryptographic protocols. This model is a generalization of Paulson's model [25] and the strand spaces model [17]. Unfortunately, it is much too expressive for decidability results. That is why we use here an abstraction where the freshness of nonces is no longer guaranteed. It may induce false attacks but is correct: if an abstracted protocol is proven correct then it is correct in the general model.

Messages are terms constructed over the alphabet $\mathcal{F} = \{< \_, \_ >, \{\_\}_\_, h\}$ and a finite set of constants C, depending on the protocol.

- $< m_1, m_2 >$ represents the concatenation of the two messages m_1 and m_2;
- $\{m_1\}_{m_2}$ represents the term m_1 encrypted by m_2;
- $h(m)$ represents the hash of m.

Note that we allow compound keys for example. We also could express asymmetric encryption but for the sake of simplicity, we do not present this in this paper. As explained in the introduction, this representation implicitly uses the perfect encryption assumption :

$$\{m\}_k = \{m'\}_{k'} \quad \Rightarrow \quad m = m' \ \& \ k = k'.$$

To relax this assumption, we add the $\oplus$ symbol together with its equational theory (described in Fig. 1) : $m_1 \oplus m_2$ represents the message m_1 xored with the message m_2. The `xor` function is widely used to encrypt messages by block [7]. It can also be used to implement a computationally cheap encryption: if K is a private key, then, instead of encrypting m with K, we may simply xor m and K. This is the case in the Bull protocol described in [24], which requires abstraction to fit in our class. We also propose the following protocol, which aims at sending a secret S_{ab}, shared by agents a, b, without using explicit encryption (hence using fewer time resources):

$$A \to B : N_a \oplus K_{ab}$$
$$B \to A : N_b \oplus N_a$$
$$A \to B : S_{ab} \oplus N_b$$

At the first step, the agent A sends a nonce N_a xored with the shared key between A and B.

We consider a predicate I which represents the set of messages possibly known to the intruder. Abstracting nonces by constants, the first rule of our protocol can be represented by the following clause:

$$\Rightarrow I(n^1_{ab} \oplus K_{ab}), \tag{1}$$

where n^1_{ab}, K_{ab} are constant symbols. At the second step, the agent B can retrieve N_a by xoring the message he received by K_{ab}. Then he generates a new nonce N_b and sends the message $N_b \oplus N_a$. This can be represented by:

$$I(z) \Rightarrow I(z \oplus n^2_{ba} \oplus K_{ab}), \tag{2}$$

where n^2_{ba} is a constant symbol. Eventually, when the agent A receives B's message, she can retrieve N_b and send a secret S_{ab} by xoring it with N_b:

$$I(z) \Rightarrow I(z \oplus n^1_{ab} \oplus S_{ab}). \tag{3}$$

These three clauses belong to our class $\mathcal{C}^{\oplus}$. Applying the reduction result of [13], we may assume that there are only two honest agents a, b and one dishonest agent c. We assume here that an honest agent is not allowed to speak with himself since we think this hypothesis is more realistic. Then, all clauses corresponding to the protocol rules are displayed in Figure 3. We use a finite set of constants $C = \bigcup_{i \in \{ab,ba,ac,ca,cb,bc\}} \{n^1_i, n^2_i, S_i\} \cup \{K_{ab}, K_{ac}, K_{bc}\}$.

Such a representation can be generalized to arbitrary protocols, and we stay within $\mathcal{C}^{\oplus}$ as soon as, at each step, at most one part of the message is blindly copied.

It remains to describe the intruder capabilities: he sees every message sent through the network and may send new messages. He knows private keys of dishonest agents. In addition, he is able to compose and decompose messages. Intruder capabilities can be encoded by clauses of $\mathcal{C}^{\oplus}$. In particular, the ability of the intruder to apply the `xor` function is described by the clause $\neg I(x) \vee \neg I(y) \vee I(x \oplus y)$. Some of the clauses are described Fig. 4. Actually, only the first three rules are relevant for our example since we only use the $\oplus$ symbol.

Now, we want to prove that S_{ab} remains unknown to the intuder when a and b are honest. Such a property may be expressed by the clause: $\phi_0 \overset{\text{def}}{=} \neg I(S_{ab})$. Let $\mathcal{C}_P$ be the clauses described in Fig. 3 and Fig. 4. The protocol satisfies our security

First rule:

$$\begin{array}{lll} \Rightarrow I(n^1_{ab} \oplus K_{ab}) & \Rightarrow I(n^1_{ba} \oplus K_{ab}) & \Rightarrow I(n^1_{ac} \oplus K_{ac}) \\ \Rightarrow I(n^1_{ca} \oplus K_{ac}) & \Rightarrow I(n^1_{cb} \oplus K_{bc}) & \Rightarrow I(n^1_{bc} \oplus K_{bc}) \end{array}$$

Second rule:

$$\begin{array}{ll} I(z) \Rightarrow I(z \oplus n^2_{ab} \oplus K_{ab}) & I(z) \Rightarrow I(z \oplus n^2_{ba} \oplus K_{ab}) \\ I(z) \Rightarrow I(z \oplus n^2_{ac} \oplus K_{ac}) & I(z) \Rightarrow I(z \oplus n^2_{ca} \oplus K_{ac}) \\ I(z) \Rightarrow I(z \oplus n^2_{bc} \oplus K_{bc}) & I(z) \Rightarrow I(z \oplus n^2_{cb} \oplus K_{bc}) \end{array}$$

Third rule:

$$\begin{array}{ll} I(z) \Rightarrow I(z \oplus n^1_{ab} \oplus S_{ab}) & I(z) \Rightarrow I(z \oplus n^1_{ba} \oplus S_{ba}) \\ I(z) \Rightarrow I(z \oplus n^1_{ac} \oplus S_{ac}) & I(z) \Rightarrow I(z \oplus n^1_{ca} \oplus S_{ca}) \\ I(z) \Rightarrow I(z \oplus n^1_{bc} \oplus S_{bc}) & I(z) \Rightarrow I(z \oplus n^1_{cb} \oplus S_{cb}) \end{array}$$

Fig. 3. Rules representing our protocol for three participants a, b and c.

$\Rightarrow I(K_{ac})$ $\Rightarrow I(K_{bc})$	The intruder knows all keys of compromised agents.
$I(x), I(y) \Rightarrow I(x \oplus y)$	The intruder may apply the `xor` function to any messages.
$I(x), I(y) \Rightarrow I(\{x\}_y)$	The intruder can encrypt a known message with a known key.
$I(\{x\}_y), I(y) \Rightarrow I(x)$	The intruder can retrieve the clear text of a message encrypted with a known key.

Fig. 4. Some of the clauses defining I.

property if and only if $\mathcal{C}_P \cup \{\phi_0\}$ is satisfiable: we are back to a satisfiability problem. Such a reduction actually works for any purely negative security property [13].

As a consequence, the security of our abstracted protocol can be decided by our decision procedure. And the answer is yes: our protocol preserves secrecy !

Proposition 3. *The set of clauses representing our protocol together with the security property $\mathcal{C}_P \cup \{\phi_0\}$ is satisfiable.*

Proof. We split the set of constants Γ into the set of (supposedly) secret data Γ_1 and known data Γ_2: $\Gamma_1 = \{n^1_{ab}, n^1_{ba}, n^2_{ab}, n^2_{ba}, S_{ab}, S_{ba}, K_{ab}\}$ and $\Gamma_2 = \Gamma \backslash \Gamma_1$. Jumping to the fixed point, we consider a set of terms T (resp. T') such that an even (resp. odd) number of "secrets" data is xored:

$$T = \{u_1 \oplus \cdots \oplus u_n \oplus t_1 \oplus \cdots \oplus t_k \mid n \text{ is even}, u_i \in \Gamma_1, t_j \in \Gamma_2, u_i, t_j \text{ distinct}\}.$$

Then we consider the following set of clauses:

$$\begin{aligned} S^* \stackrel{\text{def}}{=} &\{I(m) \mid m \in T\} \cup \{\neg I(z \oplus m_1) \vee I(z \oplus m_2) \mid m_1 \oplus m_2 \in T\} \\ &\cup \{\neg I(m_1) \vee I(m_2) \mid m_1 \oplus m_2 \in T\} \cup \{\neg I(m) \mid m \in T'\}. \end{aligned}$$

S^* contains $\mathcal{C}_P \cup \{\phi_0\}$, thus it is sufficient to prove that S^* is satisfiable (actually S^* is obtained from $\mathcal{C}_P \cup \{\phi_0\}$ by applying our deduction rules thus S^* is satisfiable iff $\mathcal{C}_P \cup \{\phi_0\}$ is satisfiable). S^* is saturated by our inference rules (see [12] for more details).Applying theorem 2, since $\bot \notin S^*$, it follows that $\mathcal{C}_P \cup \{\phi_0\}$ is satisfiable. $\Box$

Since the abstraction is an upper approximation, the above proposition shows that the protocol is secure.

Note: Instead of using the reduction result of [13], we could have introduced an arbitrary number of participants by adding new variables. For example, the second rule of our protocol could be represented by the clause:

$$A(x), A(y), I(z) \Rightarrow I(z \oplus n_2(x, y) \oplus K(x, y)),$$

where x nd y are variables representing agents. Such a clause does not belong to our class $\mathcal{C}^{\oplus}$ but we could extend $\mathcal{C}^{\oplus}$ to clauses with *basic variables* (like in [9]), representing restricted data like agents or nonces. We believe that the resulting class, which extends $\mathcal{C}^{\oplus}$, is still decidable.

5 Conclusion and perspectives

We have proved the decidability of a new first-order logic fragment, including some algebraic properties. This result applies to the automatic verification of cryptographic protocols.

There are few extensions to be considered: first, adding basic variables, as explained in the above note would be useful for the application. On the theoretical side, there is no reason to restrict each clause of S_0 to use a single predicate symbol: it should be possible to allow clauses such as $\neg P_1(x) \vee \neg P_2(y) \vee P_3(x \oplus y)$ where P_1, P_2, P_3 are distinct. We didn't consider this extension here for sake of simplicity and because we do not need it in the application. Yet another natural extesion would be to consider the class $\mathcal{C}$, adding the algebraic properties of $\oplus$. This seems to be related to the above extension.

Finally, the complexity of the decision result looks prohibitive. Before implementing the decision procedure, we need some refinements. First, we actually use a refinement of the ordering used in section 3: we established a general termination result, however, completeness holds for any ordering which is narrow liftable and compatible with the ordering used in the completeness proof on the ground level.

A last question is of course to get similar results for other equational theories. In this paper, however, we heavily rely on the particular theory of `xor`.

References

1. R. Amadio and W. Charatonik. On name generation and set-based analysis in the dolev-yao model. In *Proc. CONCUR 02*, volume 1877 of *LNCS*, pages 380–394. Springer-Verlag, 2002.
2. L. Bachmair and H. Ganzinger. Resolution theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 2, pages 19–100. North Holland, 2001.
3. B. Blanchet. Abstracting Cryptographic Protocols by Prolog Rules (invited +talk). In P. Cousot, editor, *8th International Static Analysis Symposium (SAS'2001)*, volume 2126 of *LNCS*, pages 433–436, Paris, France, July 2001. Springer Verlag.
4. D. Bolignano. Towards the mechanization of cryptographic protocol verificatio. In *9th. Conf. on Computer Aided Verification*, volume 1254 of *LNCS*, 1997.
5. L. Bozga, Y. Lakhnech, and M. Périn. Abstract interpretation for secrecy using patterns. In *Proc. TACAS*, LNCS, 2003. To appear.
6. Y. Chevalier, R. Kuester, M. Rusinowitch, and M. Turuani. An np decision procedure for protocol insecurity with xor. In *IEEE Logic in Comp. Science, to appear*, 2003.
7. J. Clark and J. Jacob. A survey of authentication protocol literature: Version, 1997.
8. E. Cohen. TAPS: A first-order verifier for cryptographic protocols. In *13th Computer Security Foundations Workshop (CSFW'00)*. IEEE Computer Society Press, 2000.

9. H. Comon and V. Cortier. Tree automata with one memory, set constraints and cryptographic protocols. Technical Report LSV-01-13, LSV, 2001. To appear in TCS.
10. H. Comon, V. Cortier, and J. Mitchell. Tree automata with memory, set constraints and ping pong protocols. In *Proc. ICALP 2001*, volume 2076 of *LNCS*. Springer Verlag, July 2001.
11. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: `http://www.grappa.univ-lille3.fr/tata`, 2002.
12. H. Comon-Lundh and V. Cortier. New decidability results for fragments of first-order logic and application to cryptographic protocols. Technical Report LSV-03-3, LSV, ENS de Cachan, Cachan, France, January 2003. 30 pages.
13. H. Comon-Lundh and V. Cortier. Security properties: two agents are sufficient. In *Proc. European Symposium on Programming*, volume 2618 of *LNCS*, pages 99–113. Springer Verlag, 2003.
14. H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *Proc. IEEE Logic in Comp. Science, to appear*, 2003.
15. N. Dershowitz. Rewriting. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 9. North Holland, 2001.
16. N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov. Undecidability of bounded security protocols. In *Proc. Workshop on formal methods in security protocols*, 1999.
17. F. T. Fabrega, J. Herzog, and J. Guttman. Strand spaces: Proving security protocol correct. *Journal of Computer Security*, 7:191–230, 1999.
18. C. Fermuller, A. Leitsch, U. Hustadt, and T. Tamet. Resolution decision procedure. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 25, pages 1793–1849. North Holland, 2001.
19. J. Goubault-Larrecq and I. Mackie. *Proof Theory and Automated Deduction*, volume 6 of Applied Logic Series. Kluwer Academic, 1997.
20. R. Kowalski and P. Hayes. Semantic trees in automated theorem proving. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 87–101. Edinburgh University Press, 1969.
21. G. Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. In Margaria and Steffen, editors, *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 1055 of *LNCS*, pages 147–166, 1996.
22. D. McAllester. Automatic recognition of tractability in inference relations. *J. ACM*, 40(2):284–303, 1993.
23. P. Narendran, Q. Guo, and D. Wolfram. Unification and matching modulo nilpotence. In *Proc. CADE-13*, volume 1104 of *LNCS*, pages 261–274, 1996.
24. L. Paulson. Mechanized proofs for a recursive authentication protocol. In *Proc. 10th IEEE Computer Security Foundations Workshop*, pages 84–95, 1997.
25. L. C. Paulson. The Inductive Approach to Verifying Cryptographic Protocols. *Journal of Computer Security*, 6(1):85–128, 1998.
26. M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *14th IEEE Computer Security Foundations Workshop*, 2001.
27. P. Ryan and S. Schneider. An attack on a recursive authentication protocol: A cautionary tale. *Information Processing Letters*, 65(1):7–10, 1998.
28. C. Weidenbach. Towards an automatic analysis of security protocols in first-order logic. In H. Ganzinger, editor, *Proc. 16th Conference on Automated Deduction*, volume 1632 of *LNCS*, pages 314–328, 1999.
29. C. Weidenbach. Combining superposition, sorts and splitting. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 27. North Holland, 2001.