Protocol security and algebraic properties: decision results for a bounded number of sessions

Sergiu Bursuc[1] and Hubert Comon-Lundh[1,2]

[1] LSV, ENS Cachan & CNRS & INRIA
[2] RCIS, AIST Tokyo

Abstract. We consider the problem of deciding the security of cryptographic protocols for a bounded number of sessions, taking into account some algebraic properties of the security primitives, for instance Abelian group properties. We propose a general method for deriving decision algorithms, splitting the task into 4 properties of the rewriting system describing the intruder capabilities: locality, conservativity, finite variant property and decidability of one-step deducibility constraints. We illustrate this method on a non trivial example, combining several Abelian Group properties, exponentiation and a homomorphism, showing a decidability result for this combination.

1 Introduction

Following the work of [16, 18, 7, 12] and many others, for a bounded number of protocol instances (sessions), the existence of attacks on some security properties (such as secrecy) of cryptographic protocols can be expressed as the existence of solutions of *deducibility constraints*. In this setting, the messages are abstracted by terms in some quotient term algebra. The semantics of deducibility constraints depends on an equational theory, that represents the intruder capabilities and the algebraic properties of the cryptographic primitives. Given such an equational theory E and assuming a fixed (bounded) number of sessions, we may guess an interleaving of actions of the given sessions. Then the security problem can be formulated as follows:

Input: given a sequence of variables $\{x_1, \ldots, x_n\}$ and an increasing sequence of finite sets of terms $T_1 \subseteq \ldots \subseteq T_n$ (representing the increasing intruder's knowledge) such that $Var(T_i) \subseteq \{x_1, \ldots, x_{i-1}\}$, and terms $u_1, v_1, \ldots, u_n, v_n$,
Question: do there exist contexts $C_1, \ldots, C_n$ and an assignment σ such that

$$C_1[T_1]\sigma =_E x_1\sigma \wedge \ldots \wedge C_n[T_n]\sigma =_E x_n\sigma \wedge u_1\sigma =_E v_1\sigma \wedge \ldots \wedge u_n\sigma =_E v_n\sigma?$$

The equations $u_i = v_i$ represent the tests performed by the agents. The variables x_i correspond to the terms forged by the intruder; as they occur in u_i, v_i, they must pass the tests of honest agents. The contexts $C_1, \ldots, C_n$ correspond to the computations (the *recipes*) of the attacker. We will give several examples.

As E-unification is a special case, the decidability of E-unifiability is a necessary condition for the decidability of deducibility constraints. This is not sufficient, as, for instance, when $E = AGh$, unification modulo E is decidable and unitary, while deducibility constraints are undecidable [11].

There are several equational theories for which deducibility constraints are decidable (in NP in most cases). This is the case for the so-called Dolev-Yao theory [17], for exclusive or [9, 4], for some theories partially modeling modular exponentiation [5, 18], etc. There are also a few results for classes of equational theories: combination results [6, 7] and monoidal theories [12]. We were faced however with a case study that does not fall into the scope of any of these general results. In this example, we need to model more properties of modular exponentiation, otherwise there is no honest execution of the protocol [3].

This particular example of application turns out to be non-trivial. We do not want however to build a new theory, only for this specific example. That is why, in the present paper, we try to draw some general method for solving deducibility constraints, in the presence of algebraic properties. We assume first that the equational theory is defined by an AC-convergent term rewriting system $\mathcal{R}$. Then we decompose the problem into 4 tasks:

Locality : if there is a context C such that $C[T] \xrightarrow[\mathcal{R}]{*} t$ and t is in normal form, then C can be chosen such that, for any subterm C' of C, $C'[T]\downarrow$ belongs to an a priori computable set of terms $\mathcal{D}(T, t)$. This yields (depending on the notion of subterm) decision algorithms for the "passive attacks" (given T, t, is there a C such that $C[T]\downarrow = t$)

Conservativity amounts, in the Dolev-Yao case, to the "small attack property": if there is a solution to the constraint C, then there is another solution σ in which the subterms of $x\sigma$ are instances of the subterms of C. This is the core of the decidability proof of [17]. In the case of other algebraic properties, the situation is more complicated as we have to replace the syntactic notion of subterms with a semantic one [7].

Finite variants : this property of a (AC) rewrite system is a slight generalization of (AC)-basic narrowing termination. It states that instantiation and rewriting can be commuted: it is possible to anticipate possible reductions triggered by instantiations. As shown in [8], this is satisfied by most equational theories that are relevant to security protocols. There are however important exceptions, such as homomorphism properties (typically AGh).

Pure deducibility constraints are deducibility constraints in which the contexts C_i that are applied to the left members, are restricted to be *pure*, i.e. to have only leaves as proper subterms. For a standard definition of subterms, this can occur only when C_i consists of a single function symbol. In the case of disjoint combinations [6], pure contexts consist of symbols of a single component theory .

We follow a similar approach as in [12]. However, in that paper, the authors only consider monoidal theories, together with Dolev-Yao rules. For instance, modeling modular exponentiation could not fall in the scope of [12], as (at least)

two associative-commutative symbols are necessary. The idea then would be, following [7], to use hierarchical combination results. However, our case study is not a hierarchical combination.

We borrow ideas from both papers and try to give a constraint solving procedure that can be applied to both hierarchical combinations and to our case study EP (*Electronic Purse*): both EP and the well-moded systems of [7] satisfy conservativity and EP is local. Then, together with the finite variant property, this allows us to reduce deducibility constraints to pure deducibility constraints (Section 3.4). Then we focus in the section 4 on pure deducibility constraint solving. This problem has much in common with combination problems. We do not provide however a general solution, but only consider our case study, trying to emphasize the main steps of this combination method.

Many proofs (including non trivial ones) are missing in this paper and can be found in [1].

2 Rewriting and security

2.1 Term Rewriting

We use classical notations and terminology from [13] on terms, unification, rewrite systems. We only give here the less standard definitions. $\mathcal{F}$ is a set of function symbols with their arity. $\mathcal{F}_{ac} \subseteq \mathcal{F}$ is a set of associative and commutative (AC in short) symbols. They are considered as varyadic. The sets of terms $\mathcal{T}(\mathcal{F}, X)$ and $\mathcal{T}(\mathcal{F})$ are defined as usual, except that we assume flattening: these algebras are rather quotients by the AC congruence for some symbols. Everywhere, equality has to be understood modulo AC. Replacements of subterms may require flattening the terms. $top(t)$ is the root symbol of t.

A substitution $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ maps each variable x_i to t_i and all other variables to themselves. It is extended to an endomorphism of $\mathcal{T}(\mathcal{F}, X)$.

Many functions from terms to terms (resp. from terms to sets of terms) are extended without mention to sets of terms by $f(S) \stackrel{\text{def}}{=} \{f(t) \mid t \in S\}$ (resp. $f(S) \stackrel{\text{def}}{=} \bigcup_{t \in S} f(t)$) and to substitutions: $f(\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}) \stackrel{\text{def}}{=} \{x_1 \mapsto f(t_1), \ldots, x_n \mapsto f(t_n)\}$ (resp. $f(\{x \mapsto t_1, \ldots, x_n \mapsto t_n\}) \stackrel{\text{def}}{=} f(\{t_1, \ldots, t_n\})$).

A term rewriting system $\mathcal{R}$ is convergent modulo $AC(\mathcal{F}_{ac})$ if it is terminating and Church Rosser modulo $AC(\mathcal{F}_{ac})$. In such a case, $s\!\downarrow$ is the normal form of the term s w.r.t. $\mathcal{R}$ (and modulo AC).

We often use the notation of contexts: $C[T]$ or $\zeta[T]$ when $T = (t_1, \ldots, t_n)$ is a sequence of terms. C, ζ are actually terms whose variables belong to $\{x_1, \ldots, x_n\}$ and $C[T]$ is another notation for $C\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$. The ordering on terms in T is only relevant to match the appropriate variable x_i: when we abstract the actual names x_i (as in the notation $C[T]$), we do not need to precise the ordering on terms in T and therefore, by abuse of notation, we use $C[T]$ when T is a set (and not a sequence).

2.2 A relevant equational theory

We will consider the following equational theory as a guideline in the paper. The set of function symbols $\mathcal{F}_{EP} = \{exp, h, +, J_+, e_+, \star, J_\star, e_\star, \bullet, J_\bullet, e_\bullet\}$ can be interpreted in the integers: exp is the (binary) exponentiation, h is some fixed-base exponentiation, $+$ is the addition, $\star$ and $\bullet$ both represent multiplication. We carefully chose the function symbols and designed an equational theory that includes sufficiently many arithmetic properties, so that the protocol can be executed [3], yet avoiding distributivity axioms that would yield undecidability. The introduction of two function symbols for the multiplication simplifies the proofs of Sections 4, 3.2, but we do not know if it is necessary.

For each $\circ \in \{+, \star, \bullet\}$, $\mathcal{R}_{AG(\circ)}$ is the rewrite system modulo $AC(\circ)$:

$$\begin{array}{cc} x \circ e_\circ \to x & x \circ J_\circ(x) \to e_\circ \\ J_\circ(x) \circ J_\circ(y) \to J_\circ(x \circ y) & J_\circ(e_\circ) \to e_\circ \\ J_\circ(J_\circ(x)) \to x & J_\circ(x) \circ x \circ y \to y \\ J_\circ(x) \circ J_\circ(y) \circ z \to J_\circ(x \circ y) \circ z & J_\circ(x \circ y) \circ x \to J_\circ(y) \\ J_\circ(x \circ y) \circ x \circ z \to J_\circ(y) \circ z & J_\circ(J_\circ(x) \circ y) \to x \circ J_\circ(y) \end{array}$$

We define $\mathcal{R}_{EP} = \mathcal{R}_0 \cup \mathcal{R}_{AG(+)} \cup \mathcal{R}_{AG(\star)} \cup \mathcal{R}_{AG(\bullet)}$, where:

$$\mathcal{R}_0 = \left\{ \begin{array}{cc} exp(h(x), y) \to h(x \star y) & J_\bullet(h(x)) \to h(J_+(x)) \\ exp(exp(x, y), z) \to exp(x, y \star z) & h(e_+) \to e_\bullet \\ h(x) \bullet h(y) \to h(x + y) & J_\bullet(h(x) \bullet y) \to h(J_+(x)) \bullet J_\bullet(y) \\ h(x) \bullet h(y) \bullet z \to h(x + y) \bullet z & exp(e_\bullet, x) \to h(e_+ \star x) \\ exp(x, e_\star) \to x & \end{array} \right.$$

Lemma 1. *$\mathcal{R}_{EP}$ is convergent modulo $AC(+, \star, \bullet)$.*

This has been mechanically verified using CiME [10].

2.3 Semantic subterms

Splitting a problem, for instance an equational theory, into subproblems, requires to gather together the symbols from the individual theories. This yields a notion of *semantic subterms* in which a *pure term* (resp. *pure context*) is a term t whose only subterms are its leaves and t itself. This notion must not be too rough, otherwise it does not yield any simplification. For instance, in the extreme case, if we define the semantic subterms as the leaves of a term, then every term is pure and we don't gain anything. On the other side, every rule of the rewriting system must have pure members (we have to decide to which subproblem it will belong).

A typical example of semantic subterms is given by a *mode assignment* as in [7]. We do not have space to describe it in detail here. Anyway, in our case study there is no appropriate mode assignment: they all yield a too rough notion of subterm in which, for instance, $exp(u, v)$ would not be a subterm of $exp(u, v) \bullet w$.

That is why we need another (refined) definition of semantic subterms for our example.

The set of subterms of t is defined recursively from its immediate subterms (called *factors*): $\mathtt{Sub}_{EP}(t) \stackrel{\text{def}}{=} \{t\} \cup \mathtt{Sub}_{EP}(\mathrm{Fact}_{EP}(t))$. We now focus on the definition of factors.

Intuitively, the factors retrieve the "alien" subterms. The factors of both sides of the rules in $\mathcal{R}_{EP}$ must be variables. Roughly, in the following definition, we carry a state (the index of $Fact$) that memorizes which part of a member of a rule has been recognized so far. For instance $a+b$ is a factor of $(a+b) \bullet u$, since there is no rule that can break $a+b$ without removing first the $\bullet$. $a+b$ is not a factor of $h(a+b) \bullet u$ since, depending on u, this can be rewritten into a term $h(a+b+v) \bullet w$, of which $a+b$ is not a subterm.

Definition 1 (EP-Factors,$\top$). *We let $Fact_{EP}(t) = Fact_{\top(t)}(t)$ where $Fact_f$ and $\top(t)$ are defined as follows (we let $\circ$ be any symbol in $\{+, \star, \bullet\}$):*

- $\top(t) = \circ$, *if* $top(t) \in \{\circ, e_\circ, J_\circ\}$, $\top(h(t)) = \bullet$, *if* $\top(t) = +$ *and* $\top(h(t)) = exp$, *if* $\top(t) = \star$. *In all other cases,* $\top(t) = top(t)$.
- $Fact_\circ(C_\circ[t_1, \ldots, t_n]) = \bigcup_{i=1}^{n} Fact_\circ(t_i)$ *if* $C_\circ \in \mathcal{T}(\{\circ, J_\circ, e_\circ\}, \mathcal{X})$
- $Fact_\bullet(h(t)) = Fact_+(t)$ *and, in all other cases,* $Fact_\circ(t) = \{t\}$ *if* $\top(t) \neq \circ$
- $Fact_{exp}(exp(u, v)) = Fact_{exp}(u) \cup Fact_\star(v)$, $Fact_{exp}(h(u)) = Fact_\star(u)$ *and, in all other cases,* $Fact_{exp}(t) = \{t\}$ *if* $\top(t) \neq exp$
- *For other function symbols* f, $Fact_f(f(t_1, ..., t_n)) = \{t_1, \ldots, t_n\}$ *and* $Fact_f(t) = \{t\}$ *if* $top(t) \neq f$.

Example 1. $\mathrm{Fact}_{EP}(exp(h(a \star b), c \star d)) = \{a, b, c, d\}$; $\mathrm{Fact}_{EP}(h(a \star b) \bullet h(a+c) \bullet (a+d)) = \{a \star b, a, c, a+d\}$.

2.4 Deducibility constraints

Definition 2 (Constraint systems). *A* deducibility constraint system C *is a finite set of equations S between terms of $\mathcal{T}(\mathcal{F}, X)$, together with a finite sequence of deducibility constraints $\{T_1 \stackrel{?}{\vdash} x_1, \ldots, T_n \stackrel{?}{\vdash} x_n\}$, where each T_i is a finite set of terms in $\mathcal{T}(\mathcal{F}, X)$, $x_1, \ldots, x_n$ are distinct variables and such that:*

Origination: $Var(T_i) \subseteq \{x_1, \ldots, x_{i-1}\}$, *for all* $1 \leq i < n$.
Monotonicity: $T_i \subseteq T_{i+1}$, *for all* $1 \leq i < n$.

Monotonicity expresses that an attacker never forgets any message. Origination expresses the commitment of the intruder to some message at each protocol step. Note that some variables of S may not belong to $\{x_1, \ldots, x_n\}$.

Example 2. The following is *not* a constraint system, as it violates the origination property:

$$\begin{cases} a, b \stackrel{?}{\vdash} z & z = x + y \\ a, b, x \stackrel{?}{\vdash} y \end{cases}$$

$$\mathcal{S}_1 = \begin{cases} a \overset{?}{\vdash} x & x = h(y+a) \\ a, x \bullet h(b) \overset{?}{\vdash} y & \end{cases}$$

is a constraint system. This corresponds to an agent expecting a message that matches $h(y+a)$ and replying $h(y+a) \bullet h(b)$. Then we ask which values of y can be deduced.

All function symbols in $\mathcal{F}$ are public (i.e. available to the intruder), except some secret data, modeled by the constants from a set C_{priv} .

Definition 3 (Solution). *A* recipe *is any context $\zeta \in T(\mathcal{F} \setminus C_{priv}, X)$: this corresponds to the attacker's sequences of actions.*

A substitution σ is a solution *(resp. a* pure solution*) of the constraint system C if its domain contains all the free variables of C and*

- *for every $s = t \in S$, $s\sigma\downarrow = t\sigma\downarrow$*
- *For every $T \overset{?}{\vdash} s \in C$, there is a recipe (resp. a pure recipe) ζ such that $\zeta[T\sigma]\downarrow = s\sigma\downarrow$*

Example 3. Consider the system $\mathcal{S}_1$ of example 2. From $a \overset{?}{\vdash} x$, x is assigned a term constructed on a and public function symbols. Then, from $x = h(y+a)$, the same property must hold for y. If $b \in C_{priv}$, we cannot use $x \bullet h(b)$ in the second constraint: the solutions assign to y any term whose only private constant symbol is a and then assign $h(y\sigma + a)\downarrow$ to x.

We are interested in the satisfiability of constraints systems: "is there an attack on the protocol ?".

3 The four main properties

3.1 Locality

(Generalized) Locality relies on the notion of semantic subterms and on a function $\mathcal{D}$ that may add or remove a few contexts. $\mathcal{D}$ is defined by a finite set of replacement rules $u_i \mapsto v_i$, $1 \leq i \leq n$. Then $\mathcal{D}(t) = \{v_i\theta \mid t = u_i\theta\}$.

Definition 4. *A rewriting system $\mathcal{R}$ is* local *w.r.t. $\mathcal{D}$ and $\mathtt{Sub}()$ if, for every recipe C, every finite set of terms T in normal form, there is a recipe ζ such that $\zeta[T]\downarrow = C[T]\downarrow$ and, $\forall \zeta' \in \mathtt{Sub}(\zeta)$, $\zeta'[T]\downarrow \in \mathcal{D}(\mathtt{Sub}(T)) \cup \mathcal{D}(\mathtt{Sub}(\zeta[T]\downarrow))$.*

In words: if t can be deduced from T, then there is a recipe ζ such that $\zeta[T]\downarrow = t$ and, adding the pure components of ζ one-by-one and normalizing at each step, we stay close to the subterms of T and t. The notion of "closeness" is provided by $\mathcal{D}$. Since we can compute beforehand all terms that are close to subterms of T or t, the locality can be used for solving the *passive attacker problem*: "given T, t in normal form, is there a recipe ζ such that $\zeta[T]\downarrow = t$?": we saturate by pure deducibility the set of terms that are close to a subterm of T or t.

The equational theories that are relevant to security are local [9, 4, 7, 12]. In the EP case, we showed a locality property, for another notion of subterm [3]. Here, $\mathcal{D}_{EP}$ is defined by the replacements $\{x \mapsto x; x \mapsto h(x); h(x) \mapsto x\}$.

Lemma 2. *EP is local (w.r.t. $\mathtt{Sub}_{EP}()$ and $\mathcal{D}_{EP}$).*

Example 4. Let $T = \{h(h(a) + J_+(b)), b\}$ and let $t = h(h(a) \star b)$. The smallest recipe ζ s.t. $\zeta[T]\downarrow = t$ is $\zeta = exp(x_1 \bullet h(x_2), x_2)$. If we consider $\zeta' = x_1 \bullet h(x_2) \in \mathtt{Sub}_{EP}(\zeta)$, we get $\zeta'[T]\downarrow = h(h(a))$, which is not in $\mathtt{Sub}_{EP}(T) \cup \mathtt{Sub}_{EP}(t)$. However, $\zeta'[T]\downarrow \in \mathcal{D}_{EP}(\mathtt{Sub}_{EP}(T))$

3.2 Conservativity

Example 5. Let $T_1 = \{a, b\}$, $T_2 = \{a, b, c, x + a\}$ and consider the deducibility constraint :

$$a, b \overset{?}{\vdash} x \quad \wedge \quad a, b, c, x + a \overset{?}{\vdash} y$$

and a solution $\sigma = \{x \mapsto a \star b; y \mapsto b + c\}$. There is a recipe in normal form $\zeta = ((x_4 + J_+(x_1)) \star J_\star(x_1)) + x_3$ such that $\zeta[T_2]\sigma\downarrow = y\sigma$ (remember that x_i refers to the ith element in the list of terms):

$$\zeta[T_2]\sigma = (((x + a + J_+(a)) \star J_\star(a)) + c)\sigma \to ((x \star J_\star(a)) + c)\sigma \to b + c$$

The last redex however is an overlap between $\zeta[T_2]\downarrow$ and σ. It consists in retrieving some subterm of $x\sigma$ (here b). This is an un-necessarily complicated way of getting b: there is a simpler recipe $\zeta' = x_2 + x_3$ that would yield also $y\sigma$.

Roughly, the subterms of $x\sigma$ can be obtained from T_1, hence, by monotonicity, they can be obtained from T_2: there is a ζ' such that $\zeta'[T_2]\sigma\downarrow = y\sigma$ and such that no rewriting step retrieves a subterm from $x\sigma$. This is what conservativity formalizes. However, in general, we may need to retrieve some subterms of $x\sigma$, but only along fixed paths, specified by a function $\mathcal{D}$.

We assume again a function $\mathcal{D}$ that is defined by a finite set of replacement rules (it does not need to be the same as for locality).

Definition 5. *E is conservative (w.r.t. $\mathcal{D}$ and $\mathtt{Sub}()$) if, for any satisfiable deducibility constraint system C, there is a solution σ such that $\mathtt{Sub}(C\sigma\downarrow) \subseteq \mathcal{D}(\mathtt{Sub}(C)\sigma\downarrow)$.*

Lemma 3. *EP is conservative w.r.t. $\mathcal{D}_{EP}$ and $\mathtt{Sub}_{EP}()$.*

The proof of this lemma is non-trivial and requires several intermediate steps that we cannot display here (see [1] for more details). The main ideas are the following: assume that σ is a solution of C and $t \in \mathtt{Sub}_{EP}(x\sigma), t \notin \mathcal{D}_{EP}(\mathtt{Sub}_{EP}(C)\sigma\downarrow)$. Then we replace t with an arbitrary constant c, yielding again a solution. Roughly, if $\zeta[T]\sigma\downarrow = x\sigma$, then replacing t with c in both members is possible, and yields $\zeta[T]\sigma'\downarrow = x\sigma'$, except when t or some $u \in \mathcal{D}_{EP}(t)$ occurs in the derivation, i.e. when, for some $\zeta' \in \mathtt{Sub}(\zeta)$, $\zeta'[T]\sigma\downarrow \in \mathcal{D}_{EP}(t)$. Then we show that we can construct another recipe for the term u, in which all private constants, which are irrelevant in the derivation $\zeta\sigma\downarrow = u$, are replaced with public constants.

Example 6. $\mathcal{D}_{EP}$ is necessary in Lemma 3:

$$\mathcal{C} = \begin{cases} h(a+b) \bullet c & \overset{?}{\vdash} x \\ h(a+b) \bullet c, exp(x \bullet J_{\bullet}(c), c \star d) & \overset{?}{\vdash} y \qquad y = h(z \star d) \end{cases}$$

$\sigma = \{x \mapsto h(a+b) \bullet c, y \mapsto h((a+b) \star c \star d), z \mapsto (a+b) \star c\}$ is a solution. $t = a + b \in \mathtt{Sub}_{EP}(\mathcal{C}\sigma\downarrow)$. $t \in \mathcal{D}_{EP}(\mathtt{Sub}_{EP}(\mathcal{C}))\sigma\downarrow \backslash \mathtt{Sub}_{EP}(\mathcal{C})\sigma\downarrow$

The proof given in [1] actually covers classes of equational theories and semantic subterms, of which EP and well-moded systems of [7] are instances.

3.3 Finite Variant Property

This notion is introduced in [8]. It states that possible reductions on instances of t can be anticipated: they are instances of the *finite variants of* t. A typical example in which we get finite variants are rewrite systems for which (basic) narrowing terminates. More examples are available in [8, 14].

Definition 6. *An AC-convergent rewrite system R has* the finite variant property*, if, for any term t, there is a finite set of terms $V(t) = \{t\theta_1\downarrow, \ldots, t\theta_k\downarrow\}$ such that $\forall\sigma.\exists\theta, \exists u \in V(t).t\sigma\downarrow = u\theta$.*

The monoidal theories do not have necessarily this property: an equation $h(x+y) = h(x) + h(y)$ cannot be oriented in any way that yields a finite set of variants for both $h(x)$ and $x+y$. In [7] there is an hypothesis called "reducibility of the theory", that is similar to (but weaker than) the finite variant property.

Lemma 4. *EP has the finite variant property.*

Example 7. Consider the term $x + J_+(a)$. Its finite variants (for EP) are $e_+ = (a + J_+(a))\downarrow$, $y = (y + a + J_+(a))\downarrow$,$J_+(a) = (e_+ + J_+(a))\downarrow$, $J_+(y+a) = (J_+(y) + J_+(a))\downarrow$, $x + J_+(a)$. Note that orienting the inverse rule in the other direction, though yielding an AC-convergent rewrite system for Abelian group, would not yield the finite variant property, as noticed in [8].

3.4 A decision algorithm for deducibility constraints

A *Pure deducibility constraint* is a deducibility constraint, of which only pure solutions are considered (see Definition 3).

Theorem 1. *If E is local, conservative and has the finite variant property, then the satisfiability of deducibility constraints is reducible to the satisfiability of pure deducibility constraints.*

Proof sketch: We assume w.l.o.g. that the functions $\mathcal{D}$ for conservativity and locality are identical (this may require merging the two sets of replacements).

Let C be the constraint system and σ be a solution of C. By conservativity, C has a solution σ_1 such that $\mathtt{Sub}(C\sigma_1\downarrow) \subseteq \mathcal{D}(\mathtt{Sub}(C)\sigma_1\downarrow)$. Thanks to the finite variant property, there is a variant $C\theta_1\downarrow$ such that $\mathtt{Sub}(C)\sigma_1\downarrow \subseteq \mathtt{Sub}(C\theta_1\downarrow)\sigma_2$ for some substitution σ_2 such that $\sigma_1 = \theta_1\sigma_2$. Then C is satisfiable iff some system $CS_1 = C\theta_1\downarrow$ has a solution σ_2 such that $\mathtt{Sub}(C)\sigma_1\downarrow \subseteq \mathcal{D}(\mathtt{Sub}(CS_1)\sigma_2)$.

Now, we pull the substitution out of the scope of $\mathcal{D}$ as follows. If $\mathcal{D}$ is defined by the replacements $u_i \mapsto v_i$, we guess which patterns u_i might be introduced by the substitution σ_2. Let θ_2 be a substitution such that, for every x, $x\theta_2$ is a renaming of some term in $\mathtt{Sub}(u_1, \ldots, u_n)$. C is satisfiable iff some $CS_2 = CS_1\theta_2$ has a solution σ_3 such that $\mathcal{D}(\mathtt{Sub}(CS_1)\sigma_2) \subseteq \mathcal{D}(\mathtt{Sub}(CS_2))\sigma_3$.

By locality, the intermediate steps in the proofs can be assumed to belong to $\mathcal{D}(\mathtt{Sub}(C\sigma_1\downarrow)) \subseteq \mathcal{D}(\mathcal{D}(\mathtt{Sub}(CS_2))\sigma_3)$. As before, we pull out σ_3: there is a substitution θ_3, and $CS_3 = CS_2\theta_3$ such that $\mathcal{D}(\mathtt{Sub}(C\sigma_1\downarrow)) \subseteq \mathcal{D}(\mathcal{D}(\mathtt{Sub}(CS_3)))\sigma_4$, for some σ_4. Then, we non-deterministically choose $\theta = \theta_1\theta_2\theta_3$, add to the equational part of C the equations $x = x\theta$ for each variable of C. Then, we guess which terms in $\mathcal{D}(\mathcal{D}(\mathtt{Sub}(CS_3)))$ are deducible, and in which order: we insert some deducibility constraints (iteratively) replacing $T_i \overset{?}{\vdash} x_i \ \wedge \ T_{i+1} \overset{?}{\vdash} x_{i+1}$ with a system

$$T_i \overset{?}{\vdash} x_i \ \wedge \ T_i \overset{?}{\vdash} z_1 \ \wedge \ T_i, z_1 \overset{?}{\vdash} z_2 \ \wedge \ldots \wedge \ T_{i+1}, z_1, \ldots, z_m \overset{?}{\vdash} x_{i+1}$$

where $z_1, \ldots, z_n$ are new variables and $z_1 = v_1 \wedge \ldots \wedge z_m = v_m$ is added to the equational part, for the guessed $v_1, \ldots, v_m \in \mathcal{D}(\mathcal{D}(\mathtt{Sub}(CS_3)))$.

By locality, any solution of the original system can be extended to the new variables into a *pure* solution of the resulting system. □

Example 8. Consider the deducibility constraint $(a, b, c \in C_{priv})$ in EP:

$$C = \begin{cases} a \star b & \overset{?}{\vdash} x \\ a \star b, exp(x, c), J_\star(b \star c), h(a) + c & \overset{?}{\vdash} y \end{cases}$$

$\sigma = \{x \mapsto h(a \star b); y \mapsto a\}$ is a solution. Let us see the branch of the above transformation yielding a pure constraint of which an extension of σ is a solution.

First, we compute a variant, guessing that $x = h(z)$ and the normal form of $exp(x, c)$ is $h(z \star c)$. Next, we guess the identity for θ_2, θ_3 and we guess which terms in $\{a, b, c, z, h(a), h(b), h(c), h(z)\} \subseteq \mathcal{D}(\mathcal{D}(\mathtt{Sub}_{EP}(CS_3)))$ are deducible and in which order. For instance we get the pure system:

$$C' = \begin{cases} a \star b & \overset{?}{\vdash} x & x \ = h(z) \\ a \star b, exp(x, c), J_\star(b \star c), h(a) + c & \overset{?}{\vdash} z_1 & z_1 = h(a) \\ a \star b, exp(x, c), J_\star(b \star c), h(a) + c, z_1 & \overset{?}{\vdash} z_2 & z_2 = c \\ a \star b, exp(x, c), J_\star(b \star c), h(a) + c, z_1, z_2 & \overset{?}{\vdash} y & \end{cases}$$

$\sigma' = \sigma \uplus \{z_1 \mapsto h(a); z_2 \mapsto c\}$ is a pure solution of C': we use the pure recipes $exp(x_2, x_3)$, $x_4 + J_+(x_5)$, $x_1 * J_\star(x_3 \star x_6)$ to obtain respectively $z_1\sigma', z_2\sigma', y\sigma'$. (Each time x_i is replaced with the ith term in T_j)

4 Pure deducibility constraints

In this section, we consider only the case study EP. We first fix the pure recipes in this case. Then, we can guess, for each elementary constraint, which type of recipe is used and we rely on a combination method.

W.r.t. classical combination methods, there are two additional difficulties. First, we cannot introduce new variables for abstracting subterms, as we might lose the origination property of the constraints, without which pure deducibility constraints become undecidable [2]. Second, the theories are not disjoint nor hierarchical: we cannot carelessly generate constraints in theory 2 while solving constraints in theory 1. Compared with [7], we do not have a bound on the number of steps performed in a higher theory.

Instead of abstracting with new variables, we show in Lemma 6 that, when $\top(u\sigma\downarrow) \neq \top(u)$, there must be another, strictly smaller (w.r.t. a well chosen ordering $\geq$), term v in $\mathcal{D}_{EP}(\mathtt{Sub}_{EP}(C))$ such that $u\sigma\downarrow = v\sigma\downarrow$. Then we add $u = v$ to the equational part of the constraint and replace u with v everywhere. Moreover, $\geq$ is defined in such a way that the resulting system still satisfies origination. After successive replacements, we "stabilize" the root symbol, allowing to fix the alien subterms in each individual constraint.

The rest of this section is devoted to the proof of the following theorem. The complexity can be derived from a careful analysis of each step.

Theorem 2. *For the equational theory EP, pure deducibility constraints can be decided in NP.*

4.1 Reduction to three recipe types

We consider 3 basic types of deducibility constraints: $T \overset{?}{\vdash}_\circ x$ for $\circ \in \{+, \star, \bullet\}$. A solution of such a constraint is a substitution σ such that

- if $\circ \in \{+, \star\}$, then there is a $\zeta_\circ \in \mathcal{T}(\{\circ, J_\circ, e_\circ\}, \mathcal{X})$ such that $\zeta_\circ[T]\sigma\downarrow = x\sigma$
- if $\circ = \bullet$, then there exist $\zeta_\bullet \in \mathcal{T}(\{\bullet, J_\bullet, e_\bullet\}, \mathcal{X}), \zeta_+ \in \mathcal{T}(\{+, J_+, e_+\}, \mathcal{X})$ such that $\zeta_\bullet[T]\sigma \bullet h(\zeta_+[T])\sigma\downarrow = x\sigma$.

We further restrict the pure deducibility constraints, reducing again the relevant recipes. Let a *basic deducibility constraint* be a conjunction of equations and of a deducibility constraint $T_1 \overset{?}{\vdash}_{\circ_1} x_1, \ldots, T_n \overset{?}{\vdash}_{\circ_n} x_n$ (still satisfying origination and monotonicity).

Lemma 5. *For any deducibility constraint C, we can effectively compute a finite set of basic deducibility constraints $C_1, \ldots, C_n$ such that the set of solutions of C is the union of the solutions of $C_1, \ldots, C_n$.*

To prove this, we first note that there are only 6 possible pure recipe types and reduce three of them to basic deducibility constraints.

Example 9. Let $a, b \overset{?}{\vdash} x$ and consider the (non-basic) pure recipe $exp(x_1, \zeta_\star)$. Replacing x_1 with a, we get the basic constraint $a, b \overset{?}{\vdash}_\star y \wedge x = exp(a, y)$. Similarly, if the pure recipe is $h(\zeta_\circ)$ ($\circ \in \{+, \star\}$), we get $a, b \overset{?}{\vdash}_\circ y \wedge x = h(y)$.

In addition, we will split $\overset{?}{\vdash}_\bullet$: $T \overset{?}{\vdash}_\bullet x$ becomes $[T; T] \overset{?}{\vdash}_\bullet x$ and, by definition, σ is a solution of $[T_1; T_2] \overset{?}{\vdash}_\bullet x$ if there are pure recipes $\zeta^1_\bullet$ and ζ^2_+ such that $\zeta^1_\bullet(T_1\sigma) \bullet h(\zeta^2_+(T_2\sigma))\downarrow = x\sigma$.

4.2 Guessing top symbols and equalities

We wish to guess the head function symbol of a term (after instantiation and normalization). Such guesses are recorded using additional constraints $H(u) \in S_t$, where S_t is either a finite set of function symbols or "Others". We also write $H(u) = f$ instead of $H(u) \in \{f\}$.

A substitution σ satisfies $H(u) \in S_t$ if $\top(u\sigma\downarrow) \in S_t.\sigma$ satisfies $H(u) \in$ "*Others*" if $u\sigma\downarrow$ is a constant, not occurring in $\mathcal{R}_{EP}$.

For each variable x in C, we guess a constraint $H(x) = f$. By abuse of notation, we let $\top(x)$ be f when the constraint system contains $H(x) = f$.

As in combination procedures, we also guess all equalities between terms in $\mathcal{D}_{EP}(\mathtt{Sub}_{EP}(C))$. Such guesses are recorded by adding equalities to the equational part of the constraint. After this step, we may only consider solutions σ such that, for any $u, v \in \mathcal{D}_{EP}(\mathtt{Sub}_{EP}(C))$, if $u\sigma\downarrow = v\sigma\downarrow$, then $u =_{Eq(C)} v$: all identities that are triggered by the substitution applications are already consequences of the equational part $Eq(C)$ of C.

4.3 Stabilizing the root symbol

If $x, y \in Var(C)$, we let $x \succ_C y$ if, for every constraint $T \overset{?}{\vdash} x \in C$, y is a variable of T. This defines an ordering $\succeq_C$, thanks to the origination property. It is extended to symbols of $\mathcal{F}$ by $x \succ_C f$ for $x \in Var(C)$, $f \in \mathcal{F}$ and $f \succ_C h$ for $f \in \mathcal{F} \setminus \{h\}$. Then $\geq$ is the multiset path ordering [13] on the precedence $\succeq_C$.

When the top symbol of a subterm is not stable by substitution and normalization, it equals a smaller subterm of the system:

Lemma 6. *For every $u \in \mathcal{Sub}_{EP}(C)$ and every solution θ, if $\top(u\theta\downarrow) \neq \top(u)$, there is a $v \in \mathcal{D}_{EP}(\mathcal{Sub}_{EP}(C))$ s.t. $v < u$ and $u\theta\downarrow = v\theta\downarrow$.*

Now, we perform the following transformation: for every $u \in \mathtt{Sub}_{EP}(C)$ such that $u =_{Eq(C)} t$ and $t \in \mathcal{D}_{EP}(\mathtt{Sub}_{EP}(C))$ and $u > t$, replace u with t in the left sides of deducibility constraints. Since equalities have been guessed, by lemma 6, after iterating the above replacements, the root symbols are stable, and the resulting system is still a constraint system, thanks to the definition of $\geq$.

Example 10.

$$\left.\begin{array}{ll} a \bullet b, c & \overset{?}{\vdash}_+ x \\ a \bullet b, c, x + J_+(c) & \overset{?}{\vdash}_\bullet y \end{array}\right\} \Rightarrow \left\{\begin{array}{l} a \bullet b, c \overset{?}{\vdash}_+ x \\ a \bullet b, c \overset{?}{\vdash}_\bullet y \end{array}\right.$$

Assume $x\sigma = a \bullet b + c$. Then the equality $x + J_+(c) = a \bullet b$ is part of the equations of C and, since $x + J_+(c) > a \bullet b$, the transformation yields the system on the right.

4.4 Eliminating variables from left hand sides: reducing deducibility constraints to linear Diophantine equations

At this stage, we could consider the alien subterms in the T_i's as constants and translate the deducibility constraints into Diophantine systems. This yields however non-linear Diophantine equations, as shown in [2]: we need to rely on origination and the Abelian group properties for further simplifications.

Let $T_i \overset{?}{\vdash}_\circ x_i$ be a constraint of C. In this step, we eliminate variables from $Fact_\circ(T_i)$, if they are headed with $\circ$ and introduced by a $\circ$ constraint.[3]

Let $\circ \in \{+, \star, \bullet\}$, $T_i \overset{?}{\vdash}_\circ x_i$ (resp. $[T_i, T_i] \overset{?}{\vdash}_\bullet x_i$) in C. We define $\mathsf{fv}_i^\circ(u)$ as follows: we let $\mathcal{X}_i^\circ$ be the set of variables x_j, $j < i$ such that $\top(x_j) = \circ$ and $T_j \overset{?}{\vdash}_\circ x_j \in C$ (resp. $[T_j, T_j] \overset{?}{\vdash}_\bullet x_j \in C$) and $u = \zeta_\circ[x_{i_1}, \ldots, x_{i_k}, u_1, \ldots, u_m]$ be such that $\mathrm{Fact}_\circ(u) \cap \mathcal{X}_i^\circ = \{x_{i_1}, \ldots, x_{i_k}\}$. Then $\mathsf{fv}_i^\circ(u) \overset{\text{def}}{=} \zeta_\circ[e_\circ, \ldots, e_\circ, u_1, \ldots, u_m]{\downarrow}$.

If $\circ \in \{+, \star\}$ and $T_i \overset{?}{\vdash}_\circ x_i \in C$, we replace every $u \in T_i$ with $\mathsf{fv}_i^\circ(u)$. If $\circ = \bullet$ and $[T_i, T_i] \overset{?}{\vdash}_\bullet x_i \in C$, we replace any $u = u' \bullet h(u'')$ in the first copy of T_i with $\mathsf{fv}_i^\bullet(u') \bullet h(\mathsf{fv}_i^+(u''))$ and every u in the second copy with $\mathsf{fv}_i^+(u)$.

Example 11. An instance of the above transformation is:

$$a \overset{?}{\vdash} x \quad \wedge \quad a, b + x \overset{?}{\vdash} y \quad \Longrightarrow \quad a \overset{?}{\vdash} x \quad \wedge \quad a, b \overset{?}{\vdash} y$$

This preserves the solutions: if ζ_+ is such that $\zeta_+[a, b + x\sigma]{\downarrow} = y\sigma$, then, rebuilding $x\sigma$ from a, we can build a new recipe ζ'_+ such that $\zeta'_+[a, b] = y\sigma$. Conversely, we can build ζ_+ from ζ'_+ by subtracting $x\sigma$ when necessary. These ideas were already applied to deduction constraints modulo Abelian groups in [18].
Similarly $x_1 \bullet a \bullet h(x_2 + b)$ would be replaced with $a \bullet h(b)$, if $\top(x_1) = \bullet$ and $\top(x_2) = +$.

The left hand sides of deducibility constraints might no longer be linearly ordered by inclusion, but we no longer need this property in further sections.

Lemma 7. *The above transformations preserve the solutions and result in a basic constraint C' such that:*

[3] We can prove actually that the variables that are headed with $\circ$ and not introduced by a $\circ$ constraint have been eliminated in the previous step.

– *if* $T_i \overset{?}{\vdash}_\circ x_i \in C'$ *and* $\circ \in \{+, \star\}$, *then for every* $x \in Fact_\circ(T_i)$, $\top(x) \neq \circ$.
– *if* $[T_i', T_i''] \overset{?}{\vdash}_\bullet x_i \in C'$, *then:*
  • *for every* $x \in Fact_+(T_i'')$, $\top(x) \neq +$.
  • *for every* $u = u_1 \bullet \ldots u_n \bullet h(v_1 + \ldots + v_m) \in T_i'$ *such that* $Fact_\bullet(u) = \{u_1, \ldots, u_n, v_1, \ldots, v_m\}$, *for every* $x \in \mathcal{X}$, $x \in \{u_1, \ldots, u_n\} \Rightarrow \top(x) \neq \bullet$ *and* $x \in \{v_1, \ldots, v_m\} \Rightarrow \top(x) \neq +$.

4.5 Turning deduction constraints into linear Diophantine equations

We do the following transformation of deducibility constraints into equations. This transformation is possible and correct by lemmas 6 and 7. For $\circ \in \{+, \star\}$,

$$\sum_i t_i^1, \ldots, \sum_i t_i^n \overset{?}{\vdash}_\circ x \quad \Longrightarrow \quad x = \sum_{i,j} \lambda_j t_i^j$$

if, $\forall i, j. \top(t_i^j) \neq \circ$ and $\lambda_1, \ldots, \lambda_n$ are new formal integer variables, representing the number of times each term is selected in the recipe.
For $\overset{?}{\vdash}_\bullet$, we use here a multiplicative notation for $\bullet$ and an additive one for $+$:

$$[\ (\textstyle\prod_i t_i^1) \bullet h(\sum_i u_i^1),\ \ldots,\ (\prod_i t_i^n) \bullet h(\sum_i u_i^n)\ ;\ \sum_i v_i^1, \ldots, \sum_i v_i^m\ ] \ \overset{?}{\vdash}\ x$$
$$\Rightarrow \begin{cases} x = x_1 \bullet h(x_2) \\ x_1 = \prod_{j=1}^n \prod_i (t_i^j)^{\lambda_j} \\ x_2 = \sum_{j=1}^n \sum_i \lambda_j u_i^j + \sum_{j=1}^m \sum_i \mu_j v_i^j \end{cases}$$

If $\forall i, j. \top(t_i^j) \notin \{\bullet, h\}$ & $\top(u_i^j), \top(v_i^j) \neq +$ and $\lambda_1, \ldots, \lambda_n, \mu_1, \ldots, \mu_m$ are integer variables.

Example 12.

$$[\ a \bullet h(2b),\ a^3 \bullet b\ ;\ 2a + 3b, 2b\ ] \ \overset{?}{\vdash}_\bullet\ x$$

is turned into $x = x_1 \bullet h(x_2)$, $x_1 = a^{\lambda_1} \bullet a^{3\lambda_2} \bullet b^{\lambda_2} = a^{\lambda_1 + 3\lambda_2} \bullet b^{\lambda_2}$, $x_2 = 2\lambda_1 b + 2\mu_1 a + 3\mu_1 b + 2\mu_2 b = 2\mu_1 a + (2\lambda_1 + 3\mu_1 + 2\mu_2)b$.

4.6 Solving the system of equations

We now have to solve a system of equations $E = E_1 \cup E_2$, where E_1 contains equations of the form $x = \Sigma_\circ \lambda_i \alpha_i t_i, H(t_i) \neq \circ$ and E_2 is a set of usual equations. After applying the finite variant property, our procedure for solving E is similar to unification procedures in the union of disjoint theories, except that we have in addition the linear Diophantine equations coming from the deducibility constraints. We recall here very briefly the main steps.

Step 1: apply the finite variant property. Equations modulo EP are reduced to equations in a combination of 3 AC theories.

Step 2: guess equalities, theories and an occurrence ordering. Since new equalities can be introduced in step 1, we guess once more the equalities between the subterms of E.

Step 3: turn the system into linear diophantine equations. The equations modulo AC yield linear Diophantine systems. That is where constraints of E_1 are inserted. After the above steps, equations in E_1 are of the form:

$$\beta_1 u_1 \circ \ldots \circ \beta_m u_m = \lambda_1 \alpha_1 t'_1 \circ \ldots \circ \lambda_n \alpha_n t'_n,\ \forall i. H(t'_i) \neq \circ$$

Since equalities have been guessed, we can simplify the equations and turn it into a linear system. For instance, if $\circ = +$ and $u_i = \sum_{j=1}^{n} u_j^i t'_j$:

$$\alpha_1\lambda_1 = \beta_1\mu_1^1 + \ldots + \beta_m\mu_1^m + \gamma_1 \ \wedge\ \ldots\ \wedge\ \alpha_n\lambda_n = \beta_1\mu_n^1 + \ldots + \beta_m\mu_n^m + \gamma_n$$

5 Conclusion

We gave a general method for deciding deducibility constraints in the presence of algebraic properties of security primitives and apply it to a non-trivial example: from Theorems 1 and 2, we deduce:

Theorem 3. *In the case of the equational theory EP, deducibility constraints are decidable (in NP).*

There is still much to do. First, we need to undestand better the combination mechanism of section 4. We chose to explain the steps of the procedure in some detail here since we feel that there should be some more general mechanism, that would be applicable to other combination problems. Next, though we have general conditions on the semantic subterms and on the rewriting systems, that imply conservativity, we did not display these conditions here, since we feel that there should be simpler conditions. For instance, it should be possible, from the rewrite system, to automatically infer the definition of subterms, in such a way that we get conservativity. This is an ambitious goal, as it would yield a systematic way of splitting an equational theory into (non disjoint, yet combinable) equational theories.

Acknowledgements

We thank Stéphanie Delaune for her numerous comments and contributions to an earlier draft of this work, as well as the anonymous referees for their valuable suggestions.

References

1. S. Bursuc and H. Comon-Lundh. Protocols, insecurity decision and combination of equational theories. Technical Report 02, Laboratoire Spécification et Vérification, February 2009. Available at `http://www.lsv.ens-cachan.fr/Publis/RAPPORTS_LSV/PDF/rr-lsv-2009-02.pdf`.

2. S. Bursuc, H. Comon-Lundh, and S. Delaune. Associative-commutative deducibility constraints. In *Proc. STACS'07*, volume 4393 of *Lecture Notes in Computer Science*, pages 634–645. Springer, 2007.
3. S. Bursuc, H. Comon-Lundh, and S. Delaune. Deducibility constraints, equational theory and electronic money. In *Rewriting, Computation and Proof — Essays Dedicated to Jean-Pierre Jouannaud on the Occasion of his 60th Birthday*, Lecture Notes in Computer Science. Springer, 2007.
4. Y. Chevalier, R. Kuester, M. Rusinowitch, and M. Turuani. An NP decision procedure for protocol insecurity with xor. In Kolaitis [15].
5. Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. Deciding the security of protocols with Diffie-Hellman exponentiation and products in exponents. In *Proc. FST/TCS, Mumbai*, volume 2914 of *Lecture Notes in Computer Science*, 2003.
6. Y. Chevalier and M. Rusinowitch. Combining Intruder Theories. In *Proc. ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*. Springer, 2005.
7. Y. Chevalier and M. Rusinowitch. Hierarchical combination of intruder theories. In *Proc. Rewriting Techniques and Application*, volume 4098 of *Lecture Notes in Computer Science*, pages 108–122, 2006.
8. H. Comon-Lundh and S. Delaune. The finite variant property: How to get rid of some algebraic properties. In *Proc. 16th Rewriting Techniques and Applications*, volume 3467 of *Lecture Notes in Computer Science*, 2005.
9. H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in preence of exclusive or. In Kolaitis [15].
10. E. Contejean and C. Marché. Cime: Completion modulo e. In *Proc. Rewriting Techniques and Applications*, volume 1103 of *Lecture Notes in Computer Science*, pages 416–419, 1996.
11. S. Delaune. An undecidability result for AGh. *Theoretical Computer Science*, 368(1-2):161–167, Dec. 2006.
12. S. Delaune, P. Lafourcade, D. Lugiez, and R. Treinen. Symbolic protocol analysis for monoidal equational theories. *Information and Computation*, 206(2-4):312–351, Feb.-Apr. 2008.
13. N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 243–309. North-Holland, 1990.
14. S. Escobar, J. Meseguer, and R. Sasse. Effectively checking the finite variant property. In *Proc. Rewriting Techniques and Applications*, 2008.
15. P. Kolaitis, editor. *Eighteenth Annual IEEE Symposium on Logic in Computer Science*, Ottawa, Canada, June 2003. IEEE Computer Society.
16. J. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *Proc. 8th ACM Conference on Computer and Communications Security*, 2001.
17. M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is np-complete. In *Proc.14th IEEE Computer Security Foundations Workshop*, Cape Breton, Nova Scotia, June 2001.
18. V. Shmatikov. Decidable analysis of cryptographic protocols with products and modular exponentiation. In *Proc. European Symposium on Programming (ESOP'04)*, volume 2986 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004.