H. Comon–Lundh and V. Shmatikov

Constraint solving, exclusive or and the decision of confidentiality for security protocols assuming a bounded number of sessions

Research Report LSV–03–1, Jan. 2003

Laboratoire Spécification et Vérification

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

Ecole Normale Supérieure de Cachan
61, avenue du Président Wilson
94235 Cachan Cedex France

http://www.lsv.ens–cachan.fr/Publis/
Research Report LSV–03–1, Lab. Spécification et Vérification, CNRS & ENS de Cachan, France, Jan. 2003

Constraint solving, exclusive or and the decision of confidentiality for security protocols assuming a bounded number of sessions.

H. Comon-Lundh
LSV, INRIA and ENS Cachan
61 Avenue du Président Wilson
94235 Cachan cedex France
Hubert.Comon@lsv.ens-cachan.fr

V. Shmatikov
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025 U.S.A.
shmat@csl.sri.com

Abstract

We present decidability results for the verification of cryptographic protocols in the presence of equational theories corresponding to `xor` *and Abelian groups. Since the perfect cryptography assumption is unrealistic for cryptographic primitives with visible algebraic properties such as* `xor`*, we extend the conventional Dolev-Yao model by permitting the intruder to exploit these properties. We show that the reachability problem is NP-complete for the extended intruder theories in the cases of* `xor` *and Abelian groups. This result follows from a normal proof theorem. Then, we show how to lift this result in the* `xor` *case: we consider a symbolic constraint system expressing the reachability (*e.g.*, secrecy) problem for a finite number of sessions. We prove that such constraint system is decidable, relying in particular on an extension of combination algorithms for unification procedures. As a corollary, this enables automatic symbolic verification of cryptographic protocols employing* `xor` *for a fixed number of sessions.*

1 Introduction

In this paper, we demonstrate that the reachability problem is decidable for cryptographic protocols that employ primitives with equational properties corresponding to `xor` and arbitrary Abelian groups. While the reachability problem for cryptographic protocols has received a lot of attention, in most approaches [2, 4, 12] the underlying cryptographic primitives are modeled, following the so called "Dolev-Yao" model [6], as abstract data types without any algebraic properties. This simplification may be realistic for information-theoretically secure crypto schemes, but it is not valid for primitives such as `xor`, products, etc.

Algebraic properties of modular exponentation are exploited by popular cryptographic protocols such as those based group Diffie-Hellman [18, 10]. Vernam cipher and cipher-block chaining mode for block ciphers rely on `xor` [11]. Their properties—associativity, commutativity, cancellation with inverses—are *not*, in general, hidden from the intruder, and abstracting away the intruder's ability to exploit them results in missed attacks. For example, the original version of Bull's recursive authentication protocol was formally proved correct in the Dolev-Yao model, but the protocol used `xor` for encryption and was thus vulnerable to an attack that exploited the self-cancellation property [14, 16]. Automatic verification of such protocols *must* take into account the fact that the cryptographic primitives obey certain equational theories. Though some procedures have been designed for verifying protocols in the presence of algebraic properties (e.g. [10]), to our knowledge, there is no relevant decision result in this case. The main contribution of this paper is to develop decision results for protocol security in the presence of relevant equational theories.

1.1 Intruder deduction problem

The first question in automating cryptographic protocol verification is the following *intruder deduction problem*:

> Given a finite set of messages T and (presumably) secret s, can the intruder deduce s from T?

Formally, messages are terms over a finite alphabet, which includes, for instance, encryption. We write $T \stackrel{?}{\vdash} s$ as a shorthand for the intruder deduction problem. This decision problem depends on the deduction capabilities of the intruder. The most widely used deduction relation in this context is shown in figure 1 (for the case of a symmetric-key cipher) and known, following [6], as the *Dolev-Yao model*: the intruder can form pairs and ciphertexts from known terms, decompose pairs, and decrypt ciphertexts only when he knows the decryption key. This assumes *perfect cryptography*: the set of messages is supposed to be a free algebra,

Pairing (P)
$$\frac{T \vdash u \quad T \vdash v}{T \vdash \langle u, v \rangle}$$

Encryption (E)
$$\frac{T \vdash u \quad T \vdash v}{T \vdash [u]_v}$$

Unpairing (UL,UR)
$$\frac{T \vdash \langle u, v \rangle}{T \vdash u} \qquad \frac{T \vdash \langle u, v \rangle}{T \vdash v}$$

Decryption (D)
$$\frac{T \vdash [u]_v \quad T \vdash v}{T \vdash u}$$

Axioms (A)
$$\frac{u \in T}{T \vdash u}$$

Figure 1. The Dolev-Yao intruder capabilities

which, of course, is an unrealistic hypothesis as many cryptographic primitives do have some algebraic properties.

$T \stackrel{?}{\vdash} s$ can be decided in polynomial time for the Dolev-Yao intruder (*i.e.*, the deduction rules of figure 1). We observe that this result can be easily derived from a theorem by D. McAllester [9]: the intruder theory given by the rules of figure 1 is *local*. This means that $T \vdash s$ *iff* there is a proof which only involves subterms of terms in T, s. This is true if we add any set of function symbols with rules allowing the intruder to apply these function symbols to any term and, for some of them, to decompose the terms. We may also add other encryption functions, in particular those modeling a public-key cipher. It is also not hard to show that $T \stackrel{?}{\vdash} s$ is complete for PTIME.

In the first part of this paper, we consider the same question, relaxing, however, the perfect cryptography assumption. Our main motivation is to develop a decision technique for protocol security that will not miss attacks such as the Ryan-Schneider attack on the recursive authentication protocol [16] in which the attacker exploits algebraic properties of the encryption scheme (see section 5).

We introduce the $\oplus$ symbol, which is interpreted either as `xor`, or as an arbitrary Abelian group operator. We prove a locality theorem for the new set of deduction rules. However, due to the associativity and commutativity of $\oplus$, locality only implies that $T \stackrel{?}{\vdash} s$ is in NP. We actually show that this problem is NP-complete.

1.2 Protocol verification via reachability analysis

A *trace-based* security property is a property whose violation can be expressed by a sequence of protocol messages. For example, secrecy is violated if the intruder can use the protocol to construct a sequence of messages at the end of which he outputs the secret value. Authentication is violated by a sequence of messages at the end of which an honest participant becomes (falsely) convinced that she is communicating with someone other than the intruder.

For trace-based security properties, verifying whether the protocol is secure is equivalent to deciding whether a particular sequence of protocol messages representing the attack or violation (*e.g.*, any sequence resulting in the intruder outputting the secret value) is reachable, that is, the intruder can use the protocol to construct this sequence. Protocol verification problem can thus be stated as a reachability problem for a message sequence [2]. A message sequence is reachable if the intruder is able to generate all messages in the sequence in the right order. For example, to violate secrecy, the intruder must be able to generate the secret value from the messages available to it at the end of the sequence. Therefore, the reachability problem is simply an ordered set of intruder deduction problems.

In general, messages exchanged as part of the protocol may contain variables and the message sequence representing the attack is thus *symbolic*. A variable represents a term whose value is initially unknown to the recipient. Therefore, the intruder is free to instantiate the variable with any term as long as instantiations are consistent. Analyzing reachability in the symbolic case is equivalent to deciding whether the symbolic sequence in question is feasible, *i.e.*, whether there exists an instantiation of variables that solves all intruder deduction problems simultaneously [8, 4, 12]. A precise definition can be found in section 3.

In general, reachability is undecidable. Undecidability results from several factors: the intruder's ability to generate fresh random data (nonce generation), the unboundedness of the number of sessions, the ability to form pairs, and the unboundedness of term sizes. Only the unbounded number of sessions is essential for undecidability. Removing one of the other three is not sufficient for decidability [5, 1, 7]. This is why the problem of reachability with a *bounded number of sessions* has drawn some attention.

Reachability in free term algebra. Several authors showed that reachability is decidable for a bounded number of sessions, *assuming perfect cryptography*. This line of research culminates with M. Rusinowitch and M. Turuani's result that this problem is co-NP-complete [15]. The decision technique used in [15, 12] consists, roughly, in first guessing an interleaving of the sessions, thus reducing the question to a single session, then expressing the

problem as a rigid reachability constraint solving problem: $T_1 \stackrel{?}{\vdash} u_1, ..., T_n \stackrel{?}{\vdash} u_n$ where each individual constraint $T_i \stackrel{?}{\vdash} u_i$ is a lifting of the ground intruder deduction problem to terms with variables. This constraint solving problem is proved to be in NP if the intruder's capabilities are modeled by the deduction rules of figure 1.

In the second part of this paper, we introduce a new symbolic constraint system in which we can express the constraints $T_i \stackrel{?}{\vdash} u_i$ and prove that these constraints are decidable in the presence of `xor`. As a consequence, reachability (and thus the failure of secrecy or authentication) is decidable for a bounded number of sessions.

Constraint solving proceeds in several steps. The first one requires an extended narrowing procedure: roughly, we want to anticipate all possible simplifications occurring after any instantiation or combination of subterms of the original constraint. Our algorithm is inspired by combination techniques for unification algorithms [3, 17] and unification modulo ACUN [13]. In the future, it should be possible to extend it to other equational theories, typically the theory of Abelian groups, for which in this paper we already prove NP-membership in the ground case.

2 Ground reachability in the presence of $\oplus$

2.1 The setting

We assume that messages are terms built over an alphabet $\mathcal{F}$ of function symbols containing constants, pairing $\langle _, _ \rangle$, encryption $[_]_$, a unary symbol $I(\ )$ and a binary function symbol $\oplus$ used in infix notation. In this paper we will only consider symmetric-key encryption, but everything can be extended to several encryption symbols, including public/private-key cryptosystems and to other free symbols as well, such as one-way functions.

We extend the intruder deduction rules of figure 1 as displayed in figure 2.

For $=_E$, we consider the congruence generated by the equations of figure 3, which we will call AG (for *Abelian groups*) or the congruence generated by these equations and the equation $I(x) = x$, which will be called the *`xor` theory*.

If we orient from left to right the equations of figure 3 other than associativity and commutativity, we get a convergent rewrite system modulo associativity and commutativity of $\oplus$. This is also the case if we add the rule $I(x) \to x$. Hence every term t has a unique normal form $t\downarrow$ up to associativity and commutativity. Terms are kept in normal form. More precisely, all terms in the context T are supposed to be in normal form and a normalization takes place after every inference rule. This will be implicit most of the time.

All rules of figure 1 and, in addition:

$$\text{Xor (X)} \quad \frac{T \vdash u \quad T \vdash v}{T \vdash u \oplus v}$$

$$\text{Inversion (I)} \quad \frac{T \vdash u}{T \vdash I(u)}$$

$$\text{Equality} \quad \frac{T \vdash t \quad t =_E u}{T \vdash u}$$

Figure 2. Intruder capabilities with equational theory

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

Figure 3. $\oplus$ axioms

If T is a finite set of terms, let $\mathsf{St}(T)$ be the least set of terms such that:

- If $t \in T$ then $t \in \mathsf{St}(T)$
- If $t \in \mathsf{St}(T)$ then $I(t)\downarrow \in \mathsf{St}(T)$
- If $\langle u, v \rangle \in \mathsf{St}(T)$ then $u, v \in \mathsf{St}(T)$
- If $[u]_v \in \mathsf{St}(T)$ then $u, v \in \mathsf{St}(T)$
- If $u_1 \oplus \ldots \oplus u_n \in \mathsf{St}(T)$ and $u_1, \ldots, u_n$ are not headed with $\oplus$, then $u_1, \ldots, u_n \in \mathsf{St}(T)$

When $I(x) = x$, $\mathsf{St}(T)$ is exactly the subterms of the *flatten* forms of terms in T, *i.e.*, the subterms when $\oplus$ is viewed as a variadic function symbol. In all cases, the number of elements in $\mathsf{St}(T)$ is linear in the size of T (the *size* of a set of terms is defined, as usual, as the sum of the number of nodes in each member of T).

2.2 Minimal size proofs

Our goal in this subsection and the next one is to show that, if $T \vdash s$, then there is a normal form proof in which only terms in $\mathsf{St}(T \cup \{s\})$ appear. For example, one way to

derive a from $\{a \oplus b, b, c\}$ (here $\oplus$ is interpreted as `xor`) is to construct $\langle b, c\rangle$, then $a \oplus b \oplus \langle b, c\rangle$, then $a \oplus \langle b, c\rangle$, and finally a. But this proof is not minimal since it uses the "unnecessary" term $\langle b, c\rangle$. The minimal proof consists in simply adding b and $a \oplus b$ to obtain a.

Since we will need to gather the $\oplus$ rule applications together, we replace the former (X) rule with the following general $\oplus$ rule, whose number of premises is unbounded:

$$\text{(GX)}\quad \frac{T \vdash u_1 \quad \dots \quad T \vdash u_n}{T \vdash u_1 \oplus \ldots \oplus u_n}$$

Define $\mathsf{SS}(T)$ to be the closure of $\mathsf{St}(T)$ under $\oplus$:

$$\mathsf{SS}(T) = \{t_1 \oplus \ldots \oplus t_n \mid \{t_1, \ldots, t_n\} \subseteq \mathsf{St}(T)\}$$

Definition 1 *A* proof *of $T \vdash u$ is a tree labeled with sequents $T \vdash v$ and such that:*

- *Every leaf of the tree is labeled with $v \in T$;*
- *Every node labeled with $T \vdash v$ has n sons $T \vdash s_1, \ldots, T \vdash s_n$ such that $\dfrac{T \vdash s_1 \cdots T \vdash s_n}{T \vdash v}$ is an instance of one of the inference rules of figure 2, with rules (X) replaced by rule (GX);*
- *The root is labeled with $T \vdash u$.*

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

Lemma 1 *If there is a minimal size proof $\mathcal{P}$ of one of the following forms:*

$$\frac{\begin{array}{c}\vdots\\ T \vdash \langle u, v\rangle\end{array}}{T \vdash u} \qquad \frac{\begin{array}{c}\vdots\\ T \vdash \langle v, u\rangle\end{array}}{T \vdash u} \qquad \frac{\begin{array}{c}\vdots\\ T \vdash [u]_v\end{array} \quad \begin{array}{c}\vdots\\ T \vdash v\end{array}}{T \vdash u}$$

Then $\langle u, v\rangle \in \mathsf{St}(T)$ (resp. $\langle v, u\rangle \in \mathsf{St}(T)$, resp. $[u]_v \in \mathsf{St}(T)$).

Proof: Assume that the last rule is (UL). The other cases are similar. By minimality, there is no subtree of $\mathcal{P}$ whose root is labeled with $T \vdash v$ and which contains itself a strict subtree whose root is labeled with $T \vdash v$. Let

$$\mathcal{P} = \frac{\dfrac{\mathcal{P}_1 \ \cdots \ \mathcal{P}_n}{T \vdash \langle u, v\rangle}}{T \vdash u}$$

$T \vdash u$ does not occur in $\mathcal{P}_1, \ldots, \mathcal{P}_n$. Let us prove then that, for any subtree

$$\mathcal{P}' = \frac{\mathcal{P}'_1 \ \cdots \ \mathcal{P}'_n}{T \vdash w},$$

if $\langle u, v\rangle$ is a subterm of w, then, for at least one i, root of $\mathcal{P}'_i$ is labeled $T \vdash w'$ or $w' \in T$, and $\langle u, v\rangle$ is a subterm of w'.

We investigate all cases for constructing the root of $\mathcal{P}'$:

If the last rule is (UL) or (UR) or (D) or (A), or (I), this is straightforward.

If the last rule is (GX), $\langle u, v\rangle$ being a subterm of $(u_1 \oplus \ldots \oplus u_n)\downarrow$, it must be a subterm of some u_i.

If the last rule is (P), then, since $T \vdash u$ does not occur in $\mathcal{P}$, w cannot be $\langle u, v\rangle$ itself. Then $w = \langle w_1, w_2\rangle \neq \langle u, v\rangle$ and, since $\langle u, v\rangle$ is a subterm of w, it must be either a subterm of w_1, or a subterm of w_2.

If the last rule is (E), it is similar to (P).

It follows that at least one leaf of $\mathcal{P}$ is labeled $w \in T$ and $\langle u, v\rangle$ is a subterm of w. This implies that $\langle u, v\rangle \in \mathsf{St}(T)$. □

2.3 Normal proofs

Definition 2 *A proof $\mathcal{P}$ of $T \vdash u$ is* normal *if*

- *either $u \in \mathsf{St}(T)$ and every non-leaf node of $\mathcal{P}$ is labeled $T \vdash v$ with $v \in \mathsf{St}(T)$*
- *or $\mathcal{P} = C[\mathcal{P}_1, \ldots, \mathcal{P}_n]$ where every proof $\mathcal{P}_i$ is a normal proof of some $T \vdash v_i$ with $v_i \in \mathsf{St}(T)$ and C is built using the inference rules P,E,GX,I only.*

Lemma 2 *If there is a proof of $T \vdash u$, then there is a normal proof of $T \vdash u$.*

Proof: Consider first the case where $u \in \mathsf{St}(T)$.

We prove the result by induction on the size of the proof (*i.e.*, number of nodes) of $T \vdash u$.

If the proof consists in an application of (A) only, the result is straightforward.

Let P be a proof of $T \vdash u$. Assume w.l.o.g. that if $T \vdash v$ labels the root of a subtree P' of P, then no strict subtree of P' is labeled with $T \vdash v$ (if not, use the hypothesis).

Consider all possible cases for the last inference:

If the last rule is (P), then $u = \langle u_1, u_2\rangle$. Observe that $u_1, u_2 \in \mathsf{St}(u) \subseteq \mathsf{St}(T)$. By induction hypothesis, there is a normal proof of $T \vdash u_1$ that involves only terms in $\mathsf{St}(T)$, and a normal proof of $T \vdash u_2$ that involves only terms in $\mathsf{St}(T)$, hence the result.

If the last rule is (E), it is similar to (P).

If the last rule is (I), then $u = I(v) \downarrow$ and there is a proof of $T \vdash v$, which is strictly smaller than $\mathcal{P}$. Moreover, $\mathsf{St}(T)$ is closed under inverses, hence $v \in \mathsf{St}(T)$. It follows, by induction hypothesis, that there is a normal proof $\mathcal{P}_1$ of $T \vdash v$. Completing $\mathcal{P}_1$ by (I) rule, we get a normal proof of $T \vdash u$.

If the last rule is (UL) (resp. (UR), resp. (D)), assume that $\mathcal{P}$ is minimal (otherwise, the induction hypothesis applies for the shorter proof of $T \vdash u$). By lemma 1,

$$\mathcal{P} = \dfrac{\dfrac{\vdots}{T \vdash \langle u, v\rangle}}{T \vdash u}$$

and $\langle u, v\rangle \in \mathsf{St}(T)$ (resp. $\langle v, u\rangle \in \mathsf{St}(T)$, resp. $[u]_v \in \mathsf{St}(T)$).

Now, we apply the induction hypothesis and obtain that there is a proof of $T \vdash \langle u, v\rangle$ (resp. $T \vdash \langle v, u\rangle$, resp. $T \vdash [u]_v$) that involves only terms in $\mathsf{St}(T)$.

If the last rule is (GX) and we assume u is in normal form, consider the maximal context C such that $C[\mathcal{P}_1, \ldots, \mathcal{P}_n] = \mathcal{P}$ and every node of C is obtained by a (GX) rule or an (I) rule (such a context does exist since at least the root node is obtained in this way). We transform the proof, gathering together (GX) inferences and commuting (I) and (GX) in such a way that (I)'s occur before (GX). (For details, see the proof transformation rules in appendix A).

The roots of $\mathcal{P}'_1, \ldots, \mathcal{P}'_n$ are respectively labeled with $T \vdash u_1 \oplus v_1, \ldots, T \vdash u_n \oplus v_n$ and $u = u_1 \oplus \ldots \oplus u_n$, $(v_1 \oplus \ldots \oplus v_n) \downarrow = 0$. We only consider here normal forms (each $u_i \oplus v_i$ is assumed to be in normal form), with the exception that some of the u_i, v_i can be 0.

Proposition 2.1 *If $(u_i \oplus v_i) \downarrow$ is headed with $\oplus$, then $(u_i \oplus v_i) \downarrow \in \mathsf{St}(T)$.*

Proof: By maximality of C, the last inference rule of $\mathcal{P}_i$ cannot be (GX) or (I). Since $u_i \oplus v_i$ is headed with $\oplus$, it cannot be (P) or (E). If it is (A), then the result is straightforward. In all other cases, we can apply lemma 1. Then either $\mathcal{P}_i = \mathcal{P}'_i$ and $(u_i \oplus v_i) \downarrow \in \mathsf{St}(T)$ or else $\mathcal{P}'_i$ is obtained from $\mathcal{P}_i$ by applying an (I) rule and $(I(u_i) \oplus I(v_i)) \downarrow \in \mathsf{St}(T)$, which implies, by definition of $\mathsf{St}(T)$, that $I((I(u_i) \oplus I(v_i)) \downarrow) \downarrow = (u_i \oplus v_i) \downarrow \in \mathsf{St}(T)$. □

For each $u_i \oplus v_i$, there are four possible cases:

Case A $v_i \neq 0$ and $(u_i \oplus v_i) \downarrow$ is headed with $\oplus$

Case B $v_i = 0$ and u_i is headed with $\oplus$

Case C $v_i = 0$ and u_i is not headed with $\oplus$

Case D $u_i = 0$ and v_i is not headed with $\oplus$

Observe that $(u_i \oplus v_i) \downarrow \in \mathsf{St}(T)$ for all terms $u_i \oplus v_i$ that fall into cases A,B (by proposition 2.1), or C (since $(u_i \oplus v_i) \downarrow = u_i \in \mathsf{St}(u)$).

By induction hypothesis, this means that there exist normal proofs $\mathcal{P}''_i$ for $T \vdash (u_i \oplus v_i) \downarrow$ that belong to cases A, B, and C. (Note that, although $\mathcal{P}'$ may be larger than $\mathcal{P}$, each $\mathcal{P}'_i$ is strictly smaller than $\mathcal{P}$ since $\mathcal{P}$ contains at least one (GX) node in addition).

Now we will argue that a normal proof for $T \vdash u = u_1 \oplus \cdots \oplus u_n$ can be constructed using *only* terms $u_i \oplus v_i$ that fall into cases A, B, or C, and that all subproofs of $T \vdash v_i$ for $v_i \notin \mathsf{St}(T)$ (case D) can be simply removed from the proof of $T \vdash u$.

Suppose there is a total of m terms $u_i \oplus v_i$ that fall into cases A, B, or C ($m \leq n$). Rearrange the proofs so that $v_1, \ldots, v_k \in \mathsf{St}(T)$ (case A) and $v_{k+1}, \ldots, v_m = 0$ (case B or C).

For every term v in normal form, we define $\mathsf{atomic}(v)$ as follows:

- If v is not headed with $\oplus$, then $\mathsf{atomic}(v) = \{v\}$
- If $v = v_1 \oplus \ldots \oplus v_k$ and $v_1, \ldots, v_k$ are not headed with $\oplus$, then $\mathsf{atomic}(v) = \{v_1, \ldots, v_k\}$.

Consider an index $i > m$. Now, since $v_1 \oplus \ldots \oplus v_n = 0$, there is a v_j such that $I(v_i) \in \mathsf{atomic}(v_j)$. However, applying proposition 2.1 again, $(u_j \oplus v_j) \downarrow$ cannot be headed with $\oplus$. For, if it were headed with $\oplus$, $(u_j \oplus v_j) \downarrow \in \mathsf{St}(T)$, which implies $v_j \in \mathsf{SS}(T)$, hence $I(v_i) \in \mathsf{St}(T)$, which contradicts $v_i \notin \mathsf{St}(T)$. It follows that $v_j = I(v_i)$ and $u_j = 0$. Thus for every term v_i such that $i > m$, for every $j \leq m$, $I(v_i) \notin \mathsf{atomic}(v_j)$. It follows that $v_1 \oplus \ldots \oplus v_k = 0$.

Therefore, the normal proof for $u_1 \oplus \cdots \oplus u_k = u_1 \oplus v_1 \oplus u_2 \oplus v_2 \oplus \cdots \oplus u_n \oplus v_n$ can be constructed simply by adding $u_1 \oplus v_1, \ldots, u_k \oplus v_k$ in any order. Since $u_1 \oplus v_1, \ldots, u_k \oplus v_k \in \mathsf{St}(T)$, and $u = u_1 \oplus \cdots \oplus u_n$, this proof satisfies the definition of the normal proof.

Finally, construct the normal proof for u by adding normal proofs for $u_{k+1}, \ldots, u_m$ as shown in figure 4.

$$\text{(GX)}\quad \frac{\overset{P_1}{\overline{T \vdash u_1 \oplus v_1}} \quad \cdots \quad \overset{P_k}{\overline{T \vdash u_k \oplus v_k}} \quad \overset{P_{k+1}}{\overline{T \vdash u_{k+1}}} \quad \cdots \quad \overset{P_m}{\overline{T \vdash u_m}}}{T \vdash \underbrace{u_1 \oplus \ldots \oplus u_m}_{=u}}$$

Figure 4. Rearranging the sums

Now let us go back to the general case: consider a minimal size proof $\mathcal{P}$ of $T \vdash u$, not assuming that $u \in \mathsf{St}(T)$. The proof can be written $C[\mathcal{P}_1, \ldots, \mathcal{P}_n]$ where $\mathcal{P}_1, \ldots, \mathcal{P}_n$ are maximal subtrees of $\mathcal{P}$ whose roots are labeled with $T \vdash v_i$ respectively and $v_i \in \mathsf{St}(T)$ for every i.

Let us prove the result by induction on the size of C:

- If C is empty, then $u \in \mathsf{St}(T)$.
- If the last inference rule of C is (UL), (UR) or (D), then, by lemma 1, $u \in \mathsf{St}(T)$.
- In all other cases, we consider the sons of $\mathcal{P}$ and apply the induction hypothesis.

□

2.4 Ground reachability is NP-complete

Theorem 3 *Given a finite set of (ground) terms T and a ground term u, the derivability of $T \vdash u$ is NP-complete, both in the* `xor` *case and in the case of Abelian groups.*

Proof: (sketch) Define *immediate derivability* as follows: $T \vdash u$ is immediately derivable if $\exists\ u_1, \ldots, u_n \in T$ such that $T \vdash u$ is obtained from $T \vdash u_1, \ldots, T \vdash u_n$ by a single application of an inference rule from (E,P,I,GX).

To prove membership in NP, we proceed as follows:

1. Guess a subset S of $\mathsf{St}(T \cup \{u\})$ containing u. Guess an ordering $s_1 > \ldots > s_n$ on $S \cap \mathsf{St}(T)$ and an ordering $u_1 > \ldots > u_m$ on the remaining part of S. Informally, we guess the order in which the subterms are derived.
2. For every $i = 1, ..., n$, check that $T \cup \{s_1, \ldots, s_{i-1}\} \vdash s_i$ is immediately derivable.
3. For every $i = 1, ..., m$, check that $T \cup \{s_1, \ldots, s_n\} \cup \{u_1, \ldots, u_{i-1}\} \vdash u$ is immediately derivable using rules from (E,P,I,GX) only.

This algorithm is in NP since there are at most $O(|\mathsf{St}(T \cup \{u\})|)$ steps (*i.e.*, polynomially many) and each step can be completed in non-deterministic polynomial time.

Now, let us show that this algorithm succeeds if and only if $T \vdash u$ is derivable. It is straightforward to see that if the algorithm succeeds, then $T \vdash u$ is derivable. Now, for the converse, we rely on lemma 2 (details in appendix B).

Now consider the NP-hardness. We reduce 3-SAT as follows: assume a set of propositional variables $\mathcal{P}$ and a set of clauses $\mathcal{C}$. Let us defined the following terms:

- $t_P \overset{\text{def}}{=} e_P \oplus a_{P,C_1} \oplus \ldots \oplus a_{P,C_k}$ if P occurs positively in the clauses $C_1, \ldots, C_k$ of $\mathcal{C}$
- $u_P \overset{\text{def}}{=} e_P \oplus b_{P,C_1} \oplus \ldots \oplus b_{P,C_k}$ if P occurs negatively in the clauses $C_1, \ldots, C_k$ of $\mathcal{C}$
- $S_C \overset{\text{def}}{=} \{d_C \oplus w_\sigma \mid \sigma \in \Sigma_C\}$ for every clause $C \in \mathcal{C}$, where Σ_C is the set of non-empty subsets of literals occurring in C (there are 7 such subsets) and w_σ is the sum of $a_{P,C}$ for $P \in \sigma$, and of $b_{P,C}$ for $\neg P \in \sigma$. (For example, if $C = P \vee Q \vee \neg R$, $S_C = \{d_C \oplus a_{P,C}, d_C \oplus a_{Q,C}, d_C \oplus b_{R,C}, d_C \oplus a_{P,C} \oplus a_{Q,C}, d_C \oplus a_{P,C} \oplus b_{R,C}, d_C \oplus a_{Q,C} \oplus b_{R,C}, d_C \oplus a_{P,C} \oplus a_{Q,C} \oplus b_{R,C}\}$).
- $u \overset{\text{def}}{=} d_{C_1} \oplus \ldots \oplus d_{C_m} \oplus e_{P_1} \oplus \ldots \oplus e_{P_n}$ if $\mathcal{C} = \{C_1, \ldots, C_m\}$ and $\mathcal{P} = \{P_1, \ldots, P_n\}$

The expressions $a_{P,C}, e_P, b_{P,C}, d_C$ are constants. They can also be encoded using a single constant and pairing.

Let $T \overset{\text{def}}{=} \{t_P \mid P \in \mathcal{P}\} \cup \{u_P \mid P \in \mathcal{P}\} \cup \bigcup_{C \in \mathcal{C}} S_C$. The size of T, u is polynomial in the size of $\mathcal{P}, \mathcal{C}$.Then we prove that $T \vdash u$ *iff* $\mathcal{C}$ is satisfiable. (Details in app. B).

□

3 Symbolic reachability

For a fixed number of sessions, the reachability problem reduces (in NP) to the reachability problem for a single session by guessing an interleaving of the processes representing each protocol role and then guessing at which step of the combined process the property is violated [12].

For example, suppose we are interested in determining whether the protocol satisfies a secrecy property. The secrecy property is violated if and only if there exists a (possibly partial) protocol session, *i.e.*, a sequence of protocol

messages that respects the partial order prescribed for each participant's role by the protocol specification, at the end of which the intruder outputs the secret value. Such a sequence exists if and only if at each step i of the protocol where the intruder must send a message, the sent term can be deduced, using the deduction rules of figure 2, from the messages the intruder has received or eavesdropped before step i. If the terms that the intruder must send contain variables, he must be able to find an instantiation for each variable, consistent among all terms in which the variable appears, so that each sent term is deducible from the messages available to him.

Following the constraint solving approach of [2, 8, 4, 12], the symbolic reachability problem is stated as follows. Given n pairs of terms with variables $u_i \rightarrow v_i$, the corresponding message sequence is reachable (and the attack feasible) *iff* there is a substitution σ such that, for every i,

- $T_i\sigma \vdash u_i\sigma$
- $T_{i+1} = T_i \cup \{v_i\}$

where T_1 is the initial knowledge of the intruder, which we assume here to be represented as a finite set of terms.

That is why we consider reachability constraint systems (hereafter simply called *constraint*), which are finite conjunctions of expressions $T \Vdash t$ where T is a finite set of terms (with variables) and t is a term (with variable) and such that, for any two atomic constraints $T \Vdash t, T' \Vdash t'$ in C, either $T \subseteq T'$ or $T' \subseteq T$.

A *solution* of a constraint C is a substitution σ such that, for every $T \Vdash_i u$ in C, $T\sigma, u\sigma$ are ground and $T\sigma \vdash u\sigma$ is derivable (using the rules of figure 2), *i.e.*, the attacker can construct term u from the terms T available to it after all variables have been instantiated as specified by σ.

4 Constraint solving in the presence of `xor`

From now on, we only consider the `xor` theory, *i.e.*, the equations of figure 3 and the identity $I(x) = x$. We will discuss the additional difficulties in the Abelian group case.

To motivate our constructions, let us show some of the difficulties on examples.

Example 1
Consider the constraint $x \Vdash a$ where x is a variable. Then the set of solutions contains $a, \langle a, b\rangle, \langle [a]_b, b\rangle, \ldots$. It is infinite and cannot be represented as a set of instances of finitely many terms.

Example 2
$\{a\} \Vdash x \oplus y \oplus a, \{a, u(x, y), ...\} \Vdash v$. The constraint can be simplified to the equivalent one: $\{a\} \Vdash x', \{a\} \Vdash y', \{u(x' \oplus z, y' \oplus z), ...\} \Vdash v$, where x', y', z are new variables. In this example, though all variables initially occur on both sides, variable z occurs only on the left.

Example 3
Let $T = \{b_2 \oplus b_3, a_1 \oplus a_3, x \oplus a_2 \oplus a_3\}$ and $u = b_1 \oplus b_3$. A solution of $T \Vdash u$ is, for instance, $x \mapsto \langle b_1 \oplus b_2, a_1 \oplus a_3\rangle \oplus a_1 \oplus a_2$.

Our first goal is to get rid of possible instantiations that collapse some of the subexpressions by application of the rules of figure 3.

Definition 3 *A substitution is* normalized*, if, for every variable x, $x\sigma$ is irreducible by the rules of figure 3.*
A normalized solution σ of C is collapse free *if:*

- *For any two distinct terms u, v in $\mathsf{St}(C)$, $u\sigma \downarrow \neq v\sigma \downarrow$*
- *For any distinct terms $u_1, \ldots, u_n \in \mathsf{St}(C)$, $(u_1 \oplus \ldots \oplus u_n)\sigma \downarrow = u_1\sigma \oplus \ldots \oplus u_n\sigma$.*

Example 4
Consider the constraint $x \oplus a \Vdash y \oplus b$. $\{x \mapsto a \oplus b; y \mapsto b \oplus b\}$ is a non-normalized solution, $\{x \mapsto a \oplus b; y \mapsto 0\}$, $\{x \mapsto c \oplus b; y \mapsto a \oplus c\}$, $\{x \mapsto b; y \mapsto a\}$ are non-collapse-free solutions. Actually, there is no collapse-free solution to this constraint since x must contain a b, which implies that $(x \oplus b)\sigma \downarrow \neq x\sigma \oplus b$

4.1 First reduction step

The goal of this section is to transform any term t into a finite set of terms $S(t)$ such that the set of instances of t (by a normalized substitution) is the set $\{u\theta \mid u \in S(t), \theta \text{ is a collapse-free substitution}\}$.

The first idea is to guess all possible equalities between subterms and solve them relying on unification procedures. However, this is not sufficient.

Example 5
Consider the term $t = f(x \oplus y, x)$ (and the theory of `xor`). Subterms are $t, x \oplus y, x, y$. Guessing equalities between these terms, we get several unsolvable systems; only three identifications yield solvable equations: the empty identification, $x = y$ and finally $x \oplus y = x$, which yields $y = 0$. With such identifications, we get the three terms $f(x \oplus y, x), f(0, x), f(x, x)$. Consider now $\sigma = \{x \mapsto a \oplus b\ y \mapsto a\}$. $t\sigma$ contains a redex and $t\sigma \downarrow = f(b, a \oplus b)$ is not an instance of any of the above three terms.

We use a procedure which is similar to the combination of unification procedures of [17, 3], but we insert step 3:

1. Introduce a new variable x_t for each non variable $t \in \mathsf{St}(C)$. We record this variable introduction in a set of equalities E containing $x_{f(t_1,\ldots,t_n)} = f(x_{t_1}, \ldots, x_{t_n})$, where, by convention, when t_i is a variable, x_{t_i} is the variable t_i itself.

$$
\begin{array}{rcl}
x = f(x_1, \ldots, x_n) \in E & \Rightarrow & \overline{x} = f(\widetilde{x_1}, \ldots, \widetilde{x_n}) \\
x = x_1 \oplus \ldots \oplus x_n \in E & \Rightarrow & \overline{x} = (\widetilde{x_1} \oplus \ldots \oplus \widetilde{x_n})\sigma_0 \downarrow \\
x > x & \Rightarrow & \bot
\end{array}
$$

Figure 5. Checking the consistency of guesses

2. Guess an equivalence relation $=_S$ on variables x_t, $t \in \mathsf{St}(C) \setminus Var(C)$. For each equivalence class, let $\widetilde{x}$ be a representative of x, G_0 be the set of representatives of x_t where t not headed with $\oplus$, and $G_\oplus$ be the set of representatives for x_t where t is headed with $\oplus$. By convention, we let $\widetilde{x} = x$ for $x \in Var(C)$.

3. Guess for each variable $x \in Var(C)$ a decomposition

$$x = \sum_{y \in G_0} \epsilon_{x,y} y \oplus \sum_{x \in S \subseteq Var(C)} \epsilon_{x,S} y_S$$

   where $\epsilon_i \in \{0, 1\}$ and y_S are new variables. The equalities are recorded in a substitution σ_0.

4. Compute, for every variable $x \in \{\widetilde{x_t}, t \in \mathsf{St}(C)\} \cup Var(C)$, the normal form $\overline{x}$ defined as follows:

   - $\overline{x_{f(t_1,\ldots,t_n)}} \stackrel{\text{def}}{=} f(\widetilde{x_{t_1}}, \ldots, \widetilde{x_{t_n}})$
   - $\overline{x_{t_1 \oplus \ldots \oplus t_n}} \stackrel{\text{def}}{=} (\widetilde{x_{t_1}} \oplus \ldots \oplus \widetilde{x_{t_n}})\sigma_0 \downarrow$.
   - $\overline{x} \stackrel{\text{def}}{=} x\sigma_0$ if $x \in Var(C)$.

   And for other variables, $\overline{x_t} \stackrel{\text{def}}{=} \overline{\widetilde{x_t}}$

5. Compute the occurrence relation on variables as the smallest transitive relation s.t. $x > y$ if y occurs in $\overline{x}$.

6. Check satisfiability using the rules of figure 5, which express on one hand the independence of the representative choice for equivalence classes and, on the other hand, the absence of cycles.

Example 6
Assume that

$$\mathsf{St}(C) = \{z, y, z \oplus y, f(z \oplus y)\}.$$

Step 1 introduces two variables $x_{z\oplus y}$ and $x_{f(z\oplus y)}$ and we have the equalities $x_{z\oplus y} = z \oplus y$ and $x_{f(z\oplus y)} = f(x_{z\oplus y})$.

At step 2, assume we don't guess any equality.

At step 3 we guess the decompositions:

$$
\begin{array}{rcl}
z & = & \epsilon_{z,f(z\oplus y)} x_{f(z\oplus y)} \oplus \epsilon_{10} x_{10} \oplus \epsilon_{11} x_{11} \\
y & = & \epsilon_{y,f(z\oplus y)} x_{f(z\oplus y)} \oplus \epsilon_{01} x_{01} \oplus \epsilon_{11} x_{11}.
\end{array}
$$

At step 4, we compute:

$$
\begin{array}{rcl}
\overline{x_{z\oplus y}} & = & (\epsilon_{z,f(z\oplus y)} x_{f(z\oplus y)} \oplus \epsilon_{y,f(z\oplus y)} x_{f(z\oplus y)} \\
 & & \quad \oplus \epsilon_{10} x_{10} \oplus \epsilon_{01} x_{01}) \downarrow \\
\overline{x_{f(z\oplus y)}} & = & f(x_{z\oplus y}) \\
\overline{z} & = & \epsilon_{z,f(z\oplus y)} x_{f(z\oplus y)} \oplus \epsilon_{10} x_{10} \oplus \epsilon_{11} x_{11} \\
\overline{y} & = & \epsilon_{y,f(z\oplus y)} x_{f(z\oplus y)} \oplus \epsilon_{01} x_{01} \oplus \epsilon_{11} x_{11}.
\end{array}
$$

At step 5, we compute the occurrence relation. For instance, if we chose $\epsilon_{z,f(z\oplus y)} = 1$ and $\epsilon_{y,f(z\oplus y)} = 0$, then we get $x_{z\oplus y} > x_{f(z\oplus y)} > x_{z\oplus y}$, which will yield unsatisfiability at the next step. Similarly, choosing $\epsilon_{z,f(z\oplus y)} = 0$ and $\epsilon_{y,f(z\oplus y)} = 1$ yields unsatisfiability.

Assume now that we chose $\epsilon_{z,f(z\oplus y)} = 1$ and $\epsilon_{y,f(z\oplus y)} = 1$, then $\overline{x_{z\oplus y}} = \epsilon_{10} x_{10} \oplus \epsilon_{01} x_{01}$ and the system passes the satisfiability test.

For instance, if $\epsilon_{11} = 0$, $\epsilon_{01} = \epsilon_{10} = 1$, we get

$$
\begin{array}{rcl}
\overline{z} & = & x_{f(y\oplus z)} \oplus x_{10} \\
\overline{y} & = & x_{f(y\oplus z)} \oplus x_{01} \\
\overline{x_{z\oplus y}} & = & x_{10} \oplus x_{01} \\
\overline{x_{f(z\oplus y)}} & = & f(x_{z\oplus y})
\end{array}
$$

Lemma 4 *The set of equalities $\overline{x} = t$ defines a substitution θ on variables such that $x_t =_S x_u$ implies $t\theta \downarrow =_{AC} u\theta \downarrow$.*

It follows that the above procedure can be seen as a non-deterministic computation of a substitution θ.

Lemma 5 *If σ is a normalised solution of C, then there is a substitution θ, which is an output of the above procedure, there is a collapse-free solution σ' of $C\theta$ such that, for every $x \in Var(C)$, $x\sigma = x\theta\sigma'$.*

This lemma allows us to focus our attention on collapse-free solutions.

Extending the lemma to the case of Abelian groups is non-trivial (see appendix G). Nevertheless, we believe that there is a similar lemma for the Abelian groups, proceeding along the lines of the above example.

4.2 Second reduction step

Let $T_1 \subseteq \ldots \subseteq T_n$ be the left members of sequents in C. Guess for each i the set S_i of terms in $\mathsf{St}(T_i)$ that can be deduced from T_i. Guess a linear ordering $>_i$ on S_i. This guess has to be consistent: $S_i \subseteq S_{i+1}$ and $>_{i+1}$ extends $>_i$. Then replace each occurrence of T_i with $T_i \cup S_i$ and add, for every i and every $u \in S_i$ the constraint $T_i \cup \{v \in S_i \mid v <_i u\} \Vdash u$.

Then, close every left member by $\oplus$ (if $u \oplus v, v \oplus w \in T$, then $u \oplus w \in T$) and remove sequents $T \Vdash u$ such that $T \vdash u$ is derivable (with the rules of figure 2).

Pairing (P)	$T \Vdash \langle u, v\rangle$	$\rightsquigarrow$	$T \Vdash u, T \Vdash v$
Encryption (E)	$T \Vdash [u]_v$	$\rightsquigarrow$	$T \Vdash u, T \Vdash v$
Split (S)	$T \Vdash u \oplus v$	$\rightsquigarrow$	$T \Vdash u, T \Vdash v$
Xor (X)	$T, v \oplus w \Vdash u \oplus v$	$\rightsquigarrow$	$T, v \oplus w \Vdash u \oplus w$

Figure 6. Constraint transformation rules

We write $C \rightsquigarrow_2 C'$ if C' can be obtained from C by the above-described transformation.

Lemma 6

- *Every σ which is a collapse-free solution of C, is also a collapse free solution of some C' such that $C \rightsquigarrow_2 C'$.*
- *for every collapse free solution σ of C, there is some C' such that $C \rightsquigarrow_2 C'$ and, for every $T \Vdash u \in C'$, for every $v \in \mathsf{St}(T)$ such that $T\sigma \vdash v\sigma$ is derivable, then either $v \in T$ or $T \vdash v$ is not derivable.*
- *If $C \rightsquigarrow_2 C'$ and $T \Vdash u \in C'$, then T is closed by $\oplus$.*
- *If $C \rightsquigarrow_2 C'$, then $\mathsf{St}(C') \subseteq \mathsf{St}(C)$*
- *If $C \rightsquigarrow_2 C'$, then every solution of C' is a solution of C*

The proof can be found in appendix D.

4.3 Solved forms and completeness

Definition 4 *C is in* solved form *if there is an ordering $\geq_{occ}$ on $\mathsf{St}(C)$ such that:*

1. *for every sequent $T \Vdash u \in C$, u is either a variable or a sum $u_1 \oplus \ldots \oplus u_n$ and in the latter case:*
   (a) *for every $i = 1, \ldots n$, there is a variable $x_{u_i} \in Var(T)$ such that $x_{u_i} >_{occ} u_i$, and*
   (b) *either $x_{u_i} \in T$ or there is a term $x_{u_i} \oplus v_{u_i} \in T$ such that $x_{u_i} >_{occ} y$ for $y \in Var(v_{u_i})$*
2. *if $T \Vdash x \oplus v \in C$ (or $T \Vdash x \in C$) and $T, T' \Vdash u_1 \oplus \ldots \oplus u_n \in C$, then for every i, $x_{u_i} \neq x$*
3. *If $x \in Var(u)$, then $u \geq_{occ} x$.*

Lemma 7 *If σ is a collapse free solution of C (obtained by the previous transformation steps), then there is a C' such that $C \rightsquigarrow C'$ by the rules of figure 6 and C' is a solved form.*

Lemma 8 *Every solved form has a solution.*

4.4 The main result

Theorem 9 *The solvability of rigid reachability constraints is decidable (in the case of* `xor`*).*

Proof: Me may only consider normalized solutions. Given a rigid reachability constraint C, perform the first two reduction steps. They terminate and, according to lemmas 6, 5. C has a solution only if at least one of the resulting constraints C' has a collapse-free solution. And if some of the resulting constraints has a solution, then C has a solution.

Now, consider the rules of figure 6. They preserve the set $\mathsf{St}(C)$. Hence only finitely many distinct constraints can be derived using such rules. If the application of the rules does not terminate, that can only be due to a loop (with **X**). We simply avoid loops, keeping tracks of previous constraints. Then, by lemma 7, if C has a solution, then there is a constraint C' obtained by this (now terminating) set of rules such that C' is a solved form and C' has a solution. And conversely: the solutions of such C' are also solutions of C.

Then either all C' obtained in this way are not in solved form, in which case C has no solution or else there is a constraint C' in solved form, in which case C has a solution, thanks to lemma 8. □

Corollary 10 *In the presence of* `xor`*, the failure of secrecy or authentication is decidable for a fixed number of sessions.*

5 Example: Bull's authentication protocol

We illustrate the constraint solving technique of section 4 by applying it to Bull's recursive authentication protocol. The protocol was first proved correct [14], and then shown to be vulnerable [16] once properties of `xor` are taken into account. We consider the 3-party version of the protocol (see figure 7). It is assumed that $K_{a,b,c}$ are shared between S and A, B, C respectively, and h is a function symbol modeling a hash function. Among other goals, the protocol should prevent C from learning session key K_{ab}.

Secrecy of K_{ab} from C fails if the following constraint is satisfiable: $T = \{M_b, x, y, K_{ab} \oplus h(K_a, N_a), K_{ab} \oplus h(K_b, N_b), K_{bc} \oplus h(K_b, N_b), K_{bc} \oplus h(x, y)\} \Vdash K_{ab}$ (x, y represent terms under control of C). We guess that x, y are not instantiated, and the following order of subterm derivation: $h(x, y) < K_{bc} < K_{ab}$. Observe that $T \vdash h(x, y)$ in one step by applying function symbol h to x and y, $T \cup h(x, y) \vdash K_{bc}$ by (X), $T \cup h(x, y) \cup K_{bc} \vdash K_{ab}$ by two applications of (X): $((K_{ab} \oplus h(K_b, N_b)) \oplus (K_{bc} \oplus h(K_b, N_b))) \oplus K_{bc} = K_{ab}$. Therefore, after the second reduction step we obtain an empty constraint, proving that secrecy of K_{ab} is violated.

1. $A \to B : M_a = A, B, N_a, h(A, B, N_a)$
2. $B \to C : M_b = B, C, N_b, M_a, h(B, C, N_b, M_a)$
3. $C \to S : M_c = C, S, N_c, M_b, h(C, S, N_c, M_b)$
4. $S \to C : K_{ab} \oplus h(K_a, N_a), K_{ab} \oplus h(K_b, N_b),$ $K_{bc} \oplus h(K_b, N_b), K_{bc} \oplus h(K_c, N_c)$
5. $C \to B : K_{ab} \oplus h(K_a, N_a), K_{ab} \oplus h(K_b, N_b),$ $K_{bc} \oplus h(K_b, N_b)$
6. $B \to A : K_{ab} \oplus h(K_a, N_a)$

Figure 7. Recursive authentication protocol

6 Concluding remarks

We gave the first decision result for cryptographic protocols which does not assume the perfect cryptography. First, our work is incomplete in two respects:

(i) What is the exact complexity of solving the rigid reachability constraint in the `xor` case?

(ii) What about an analog of theorem 9 in the case of Abelian groups?

It should be possible to answer the second question with a significant extra effort and time since we believe that there is an analog of lemma 5 for Abelian groups.
The techniques introduced in section 2 and the work of McAllester [9] raise a more general question:

> For which (relevant) equational theories is the reachability problem in NP? decidable?

The techniques introduced in section 4.1 also raise a general question in unification theory:

> For which equational theories can we restrict our attention to collapse-free solutions?

In other words: can we extend lemma 5 to arbitrary theories for which unification is finitary?
Finally, it would be nice to abstract out the lifting argument of section 4:

> Is there a general relationship between the complexity of reachability in the ground case and the solvability of rigid reachability constraints?

References

[1] R. Amadio and W. Charatonik. On name generation and set-based analysis in the Dolev-Yao model. In *Proc. CONCUR '02*, volume 2421 of *LNCS*, pages 499–514, 2002.

[2] R. Amadio and D. Lugiez. On the reachability problem in cryptographic protocols. In *Proc. CONCUR '00*, volume 1877 of *LNCS*, pages 380–394, 2000.

[3] F. Baader and K. Schulz. Combining constraint solving. In *Constraints in Computational Logics (CCL '99)*, volume 2002 of *LNCS*, pages 104–158, 2001.

[4] M. Boreale. Symbolic trace analysis of cryptographic protocols. In *Proc. ICALP '01*, pages 667–681, 2001.

[5] H. Comon, V. Cortier, and J. Mitchell. Tree automata with one memory, set constraints, and ping-pong protocols. In *Proc. ICALP '01*, pages 682–693, 2001.

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

[7] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov. Undecidability of bounded security protocols. In *Proc. Workshop on formal methods in security protocols*, Trento, Italy, 1999.

[8] M. Fiore and M. Abadi. Computing symbolic models for verifying cryptographic protocols. In *Proc. 14th IEEE Computer Security Foundations Workshop*, pages 160–173, 2001.

[9] D. McAllester. Automatic recognition of tractability in inference relations. *J. ACM*, 40(2):284–303, 1993.

[10] C. Meaddows and P. Narendran. A unification algorithm for the group diffie-hellman protocol. In *Proc. Workshop on Issues in Theory of Security*, 2002.

[11] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[12] J. Millen and V. Shmatikov. Constraint solving for bounded process cryptographic protocol analysis. In *Proc. 8th ACM Conference on Computer and Communications Security (CCS '01)*, pages 166–175, 2001.

[13] P. Narendran, Q. Guo, and D. Wolfram. Unification and matching modulo nilpotence. In *Proc. CADE-13*, volume 1104 of *LNCS*, pages 261–274, 1996.

[14] L. Paulson. Mechanized proofs for a recursive authentication protocol. In *Proc. 10th IEEE Computer Security Foundations Workshop*, pages 84–95, 1997.

[15] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *Proc. 14th IEEE Computer Security Foundations Workshop*, pages 174–190, 2001.

[16] P. Ryan and S. Schneider. An attack on a recursive authentication protocol: A cautionary tale. *Information Processing Letters*, 65(1):7–10, 1998.

[17] M. Schmidt-Schauß. Unification in a combination of arbitrary disjoint equational theories. *J. Symbolic Computation*, 8(1/2):51–99, 1989.

[18] M. Steiner, G. Tsudik, and M. Waidner. Diffie-Hellman key distribution extended to group communication. In *Proc. 3rd ACM Conference on Computer and Communications Security (CCS '96)*, pages 31–37, 1996.

A Proof of lemma 2: the auxiliary proof transformation rules

The auxiliary proof transformation rules are displayed in figure 8.

Note that the transformation rules preserve conclusions of the proofs since $(I((u \oplus v)\downarrow))\downarrow = (I(u) \oplus I(v))\downarrow$ and $(u\downarrow \oplus v)\downarrow = (u \oplus v)\downarrow$.

The rules terminate (consider, *e.g.*, a path ordering) and yield a proof $\mathcal{P}' = C'[\mathcal{P}'_1, \ldots \mathcal{P}'_n]$ of $T \vdash u$, such that C' consists in a single application of (GX) and $\mathcal{P}'_i$ is either $\mathcal{P}_i$ itself or a single (I) rule applied to $\mathcal{P}_i$. $\mathcal{P}'$ may be larger than $\mathcal{P}$.

B Proof of theorem 3

Theorem 3 *Given a finite set of (ground) terms T and a ground term u, the derivability of $T \vdash u$ is NP-complete, both in the* `xor` *case and in the case of Abelian groups.*

Proof: Let us define *immediate derivability* as follows: $T \vdash u$ is immediately derivable if there are terms $u_1, \ldots, u_n \in T$ such that $T \vdash u$ is obtained from $T \vdash u_1, \ldots, T \vdash u_n$ by a single application of an inference rule from (E,P,I,GX).

First, to prove the NP-membership, we proceed as follows:

1. Guess a subset S of $\mathsf{St}(T \cup \{u\})$ containing u. Guess an ordering $s_1 > \ldots > s_n$ on $S \cap \mathsf{St}(T)$ and an ordering $u_1 > \ldots > u_m$ on the remaining part of S. Informally, we guess the order in which the subterms are derived.

2. For every $i = 1, ..., n$, check that $T \cup \{s_1, \ldots, s_{i-1}\} \vdash s_i$ is immediately derivable.

3. For every $i = 1, ..., m$, check that $T \cup \{s_1, \ldots, s_n\} \cup \{u_1, \ldots, u_{i-1}\} \vdash u$ is immediately derivable using rules E,P,I,GX only.

First, this algorithm is in NP. There are at most $O(|\mathsf{St}(T \cup \{u\})|)$ steps (*i.e.*, polynomially many) and each step can be completed in non-deterministic polynomial time: checking that a term t is a sum of terms in T is a particular case of integer programming.

Now, let us show that this algorithm succeeds if and only if $T \vdash u$ is derivable. It is straightforward to see that if the algorithm succeeds, then $T \vdash u$ is derivable. Now, consider the converse. By lemma 2, we only have to consider a normal proof $\mathcal{P}$ of $T \vdash u$. Let then S be the terms v in $\mathsf{St}(T \cup \{u\})$ such that $T \vdash v$ occurs as a label of $\mathcal{P}$. We define the relation R on S as follows (informally, vRw means that the proof of $T \vdash v$ does not depend on the proof of $T \vdash w$):

> vRw *iff* there exists a subterm P_v of $\mathcal{P}$, whose root is labeled with $T \vdash v$ and such that $T \vdash w$ does not label any node of P_v.

Observe that, for every $v, w \in S$ such that $v \neq w$, either vRw or wRv since $T \vdash v$ cannot label two distinct nodes on the same path of $\mathcal{P}$. This means that $\overline{R} \stackrel{\text{def}}{=} S \times S \setminus R$ is antisymmetric. $\overline{R}$ is also reflexive since R is anti-reflexive by definition. Next, if $v_1 \overline{R} v_2$ and $v_2 \overline{R} v_3$, then in all subproofs P_{v_1} there is a subproof P_{v_2} and in all proofs P_{v_2} there is a subproof P_{v_3}, hence in all subproofs of $T \vdash v_1$, there is a subproof of $T \vdash v_3$. It follows that $\overline{R}$ is transitive, hence an ordering.

Let us show now that for every terms $u_1 \in S \setminus \mathsf{St}(T)$ and $u_2 \in S \cap \mathsf{St}(T)$, $u_2 R u_1$. In such a situation, $u_1 \in \mathsf{St}(u) \setminus \mathsf{St}(T)$. Hence, by definition of a normal proof, $T \vdash u_1$ does not label a normal proof of $T \vdash u_2$.

We choose now for $\geq$ any total ordering compatible with $\overline{R}$ and such that terms in $S \setminus \mathsf{St}(T)$ are smaller than terms in $S \cap \mathsf{St}(T)$. By definition of a normal proof, for every $v \in S$, there is a subterm P_v of $\mathcal{P}$ such that $T \vdash v$ is obtained from $T \vdash v_1, \ldots, T \vdash v_k$ either by applying one inference rule or by summing the v_i's and $v_1, \ldots, v_k \in \mathsf{St}(T)$. By definition, $v_i > v$ for every i, hence $T \cup \{v_1, \ldots, v_k\} \vdash v$ is immediately derivable.

Now consider the NP-hardness. We reduce 3-SAT as follows: assume a set of propositional variables $\mathcal{P}$ and a set of clauses $\mathcal{C}$. Let us defined the following terms:

- $t_P \stackrel{\text{def}}{=} e_P \oplus a_{P,C_1} \oplus \ldots \oplus a_{P,C_k}$ if P occurs positively in the clauses $C_1, \ldots, C_k$ of $\mathcal{C}$
- $u_P \stackrel{\text{def}}{=} e_P \oplus b_{P,C_1} \oplus \ldots \oplus b_{P,C_k}$ if P occurs negatively in the clauses $C_1, \ldots, C_k$ of $\mathcal{C}$
- $S_C \stackrel{\text{def}}{=} \{d_C \oplus w_\sigma \mid \sigma \in \Sigma_C\}$ for every clause $C \in \mathcal{C}$, where Σ_C is the set of non-empty subsets of literals occurring in C (there are 7 such subsets) and w_σ is the sum of $a_{P,C}$ for $P \in \sigma$, and of $b_{P,C}$ for $\neg P \in \sigma$. (For example, if $C = P \vee Q \vee \neg R$, $S_C = \{d_C \oplus a_{P,C}, d_C \oplus a_{Q,C}, d_C \oplus b_{R,C}, d_C \oplus a_{P,C} \oplus a_{Q,C}, d_C \oplus a_{P,C} \oplus b_{R,C}, d_C \oplus a_{Q,C} \oplus b_{R,C}, d_C \oplus a_{P,C} \oplus a_{Q,C} \oplus b_{R,C}\}$).
- $u \stackrel{\text{def}}{=} d_{C_1} \oplus \ldots \oplus d_{C_m} \oplus e_{P_1} \oplus \ldots \oplus e_{P_n}$ if $\mathcal{C} = \{C_1, \ldots, C_m\}$ and $\mathcal{P} = \{P_1, \ldots, P_n\}$

The expressions $a_{P,C}, e_P, b_{P,C}, d_C$ are constants. They can also be encoded using a single constant and pairing.

Let $T \stackrel{\text{def}}{=} \{t_P \mid P \in \mathcal{P}\} \cup \{u_P \mid P \in \mathcal{P}\} \cup \bigcup_{C \in \mathcal{C}} S_C$. The size of T, u is polynomial in the size of $\mathcal{P}, \mathcal{C}$.Then we prove that $T \vdash u$ *iff* $\mathcal{C}$ is satisfiable.

If $\mathcal{C}$ is satisfiable, variables in $\mathcal{P}_1$ being assigned true and variables in $\mathcal{P}_2$ being assigned false, then we consider

$$\text{(GX)}\ \frac{T\vdash u_1\ \ldots\ T\vdash u_n}{\text{(I)}\ \dfrac{T\vdash (u_1\oplus\ldots\oplus u_n)\downarrow}{T\vdash I((u_1\oplus\ldots\oplus u_n)\downarrow)}}\quad\Rightarrow\quad\text{(GX)}\ \frac{\text{(I)}\ \dfrac{T\vdash u_1}{T\vdash I(u_1)}\quad\cdots\quad\text{(I)}\ \dfrac{T\vdash u_n}{T\vdash I(u_n)}}{T\vdash (I(u_1)\oplus\ldots\oplus I(u_n))\downarrow}$$

$$\text{(GX)}\ \frac{\text{(GX)}\ \dfrac{T\vdash u_1\quad\cdots\quad T\vdash u_n}{T\vdash (u_1\oplus\ldots\oplus u_n)\downarrow}\qquad T\vdash v_1\qquad\cdots\qquad T\vdash v_m}{T\vdash ((u_1\oplus\ldots\oplus u_n)\downarrow\oplus v_1\oplus\ldots\oplus v_m)\downarrow}$$

$$\Rightarrow\quad\text{(GX)}\ \frac{T\vdash u_1\ \cdots\ T\vdash u_n\ T\vdash v_1\ \cdots\ T\vdash v_m}{T\vdash (u_1\oplus\ldots\oplus u_n\oplus v_1\oplus\ldots\oplus v_m)\downarrow}$$

$$\text{(I)}\ \frac{\text{(I)}\ \dfrac{T\vdash u}{T\vdash I(u)}}{T\vdash u}\quad\Rightarrow\quad T\vdash u$$

Figure 8. Proof transformation rules

the proof consisting of $\oplus$-rule applications only and whose leaves are labeled t_P for $P\in\mathcal{P}_1$, u_P for $P\in\mathcal{P}_2$, $d_C\oplus w_\sigma$ if σ is the set of literals occurring in C and which are satisfied by the assignment. The sum of these terms contains every constant d_C exactly once, every constant e_P exactly once, every constant $a_{P,C}$ with $P\in\mathcal{P}_1$ exactly twice, every constant $b_{P,C}$ with $P\in\mathcal{P}_2$ exactly twice and no other expression. Hence thi sum normalizes to u.

Conversely, if there is a proof of $T\vdash u$, then there is a normal proof by lemma 2, hence there is a proof using the $\oplus$-rule only. Moreover, each term of T can be assumed to occur only once. Then, for every P, either t_P occurs or u_P and not both. For every clause C, there is an odd number terms $d_C\oplus w_\sigma$ occurring in the proof, hence their sum is again of the form $d_C\oplus w_\sigma$ and we can assume w.l.o.g that only one of these terms occurs in the proof. Then, for every P, C, either $a_{P,C}$ occurs exactly twice in the leaves or $b_{P,C}$ occurs exactly twice in the leaves and not both. Then consider the assignment mapping P to true *iff* t_P is a leaf of the proof. This assignment satisfies $\mathcal{C}$.

□

C Proof of lemma 5

Lemma 5 *If σ is a normalised solution of C, then there is a substitution θ, which is an output of the above procedure, there is a collapse-free solution σ' of $C\theta$ such that, for every $x\in Var(C)$, $x\sigma = x\theta\sigma'$.*

Proof: Given σ, we first construct $=_S, \theta$ (and σ'). We let $x_t =_S x_u$ whenever $t,u\in\mathsf{St}(C)$ and $t\sigma\downarrow =_{AC} u\sigma\downarrow$. Choose arbitrarily one representative for each equivalence class.

Let x be a variable of C and $x\sigma=\sum_{i\in I_x}u_i$ where u_i is not headed with $\oplus$ and the u_i's are distinct. If $t\sigma\downarrow =_{AC} u_i$ for some $t\in\mathsf{St}(C)$ and $i\in I_x$, we let $\epsilon_{x,\widetilde{x_t}}=1$ and 0 otherwise. Note that, by definition of $=_S$, this is well-defined. Then let

$$x\sigma=\sum_{\widetilde{x_t}\in G_0, t\sigma\downarrow=u_i} u_i\oplus\sum_{i\in J_x}u_i$$

and for every $i\in J_x$, for every $t\in\mathsf{St}(C)$, $u_i\neq_{AC} t\sigma\downarrow$.

Now, if $S\subseteq Var(C)$, $S\neq\emptyset$, we let $E_S=\{u\mid x\in S\Leftrightarrow\exists i\in J_x, u=_{AC}u_i\}$. We let $\epsilon_{x,S}=1$ *iff* $E_S\neq\emptyset$.

This defines the substitution σ_0 on $Var(C)$, as in step 3 of the procedure:

$$x\sigma_0=\sum_{y\in G_0}\epsilon_{x,y}y\oplus\sum_{x\in S\subseteq Var(C)}\epsilon_{x,S}y_S$$

Consider now the occurrence relation $>$ as defined in steps 4 and 5 of the procedure. We claim that $\overline{\ }, =_S$ and $>$ pass the consistency check of figure 5:

- if $x_{f(t_1,\ldots,t_n)}=f(x_{t_1},\ldots,x_{t_n})\in E$, $\widetilde{x_{f(t_1,\ldots,t_n)}}=x_u$, then, by definition, $f(t_1,\ldots,t_n)\sigma\downarrow = u\sigma\downarrow$. Then

either $u = f(u_1, \ldots, u_n)$ and, for every i, $u_i\sigma \downarrow = t_i\sigma \downarrow$, which implies $\widetilde{x_{t_i}} = \widetilde{x_{u_i}}$, hence

$$\begin{array}{rcl} \overline{x_{f(t_1,\ldots,t_n)}} & = & \overline{x_{f(u_1,\ldots,u_n)}} \\ & = & f(\widetilde{x_{u_1}}, \ldots, \widetilde{x_{u_n}}) \\ & = & f(\widetilde{x_{t_1}}, \ldots, \widetilde{x_{t_n}}) \end{array}$$

Or else $u =_{AC} u_0 \oplus u_1 \oplus \ldots \oplus u_{2n} \oplus x_1 \oplus \ldots \oplus x_m$, where every $u_i, i \geq 0$ is not headed with $\oplus$, every $u_i, i \geq 1$ is not a variable, $x_1, \ldots, x_m$ are variables, for every $i \geq 1$, $u_{2i-1}\sigma \downarrow = u_{2i}\sigma \downarrow$, and either u_0 is not a variable and $u\sigma \downarrow = u_0\sigma \downarrow$ or u_0 is a variable x_0. In such a case, by definition $\widetilde{x_{u_{2i-1}}} = \widetilde{x_{u_{2i}}}$. Remains to verify that, with our definitions, $(x_1 \oplus \ldots \oplus x_m)\sigma_0 \downarrow = 0$ (and $u_0\sigma \downarrow = u\sigma \downarrow$) or $(x_0 \oplus \ldots \oplus x_m)\sigma_0 \downarrow = f(\widetilde{x_{t_1}}, \ldots, \widetilde{x_{t_n}})$. Consider for instance the first case. $(x_1 \oplus \ldots \oplus x_m)\sigma \downarrow = 0$ and

$$(\sum_{i=1}^{n} x_i)\sigma_0 = \sum_{i=1}^{m}(\sum_{y \in G_0} \epsilon_{x_i,y} y \oplus \sum_{x_i \in S \subseteq Var(C)} \epsilon_{x,S} y_S)$$

$\sum_{i=1}^{n} \epsilon_{x_i,\widetilde{x_t}}$ is the number of occurrences of $t\sigma \downarrow$ in $(x_1 \oplus \ldots \oplus x_m)\sigma$, hence it is an even number. Similarly, if we fix $S \subseteq Var(C)$, $\sum_{i=1}^{m} \epsilon_{x_i,S}$ is even. Hence the normal form of the above expression is 0.

- if $x_{t_1 \oplus \ldots \oplus t_n} = x_{t_1} \oplus \ldots \oplus x_{t_n} \in E$, we proceed essentially in the same way as in the the above case: $\overline{x_{t_1 \oplus \ldots \oplus t_n}} = \widetilde{x_{t_1 \oplus \ldots \oplus t_n}} = \overline{x_u}$ and $u\sigma \downarrow = (t_1 \oplus \ldots \oplus t_n)\sigma \downarrow$. Let $u = u_1 \oplus \ldots \oplus u_m$. By definition, $\overline{x_u} = (\widetilde{x_{u_1}} \oplus \ldots \oplus \widetilde{x_{u_m}})\sigma_0 \downarrow$. Remains to show that

$$(\widetilde{x_{t_1}} \oplus \ldots \oplus \widetilde{x_{t_n}})\sigma_0 \downarrow = (\widetilde{x_{u_1}} \oplus \ldots \oplus \widetilde{x_{u_m}})\sigma_0 \downarrow$$

which is a consequence of the definition of σ_0.

- $>$ does not contain cycles since $x > y$ only if there is a term u such that $y\sigma \oplus u \in \mathsf{St}(x\sigma)$, by construction.

Then θ is recursively defined as in the step 4 of the procedure: $x\theta \stackrel{\text{def}}{=} \widetilde{x}\theta$ and, if $\overline{x} = f(x_1, \ldots, x_n)$, then $x\theta \stackrel{\text{def}}{=} f(x_1\theta, \ldots, x_n\theta)$, if $\overline{x} = x_1 \oplus \ldots \oplus x_n$, then $x\theta \stackrel{\text{def}}{=} x_1\theta \oplus \ldots \oplus x_n\theta$. This definition is well-founded, as $>$ does not contain cycles.

Let σ' be the substitution mapping every y_S to $\sum_{u \in E_S} u$. We have

$$x\sigma =_{AC} x\theta\sigma'$$

Now, it remains to show that, for every $u, v \in \mathsf{SS}(C\theta)$, either $u \downarrow =_{AC} v \downarrow$ or else $u\sigma' \downarrow \neq_{AC} v\sigma' \downarrow$ (which implies that σ' is collapse-free).

First consider the case where $u, v \in \mathsf{St}(C\theta)$. Every term in $\mathsf{St}(C\theta)$ is either equal to some $\overline{x}\theta$ for $x \in \{x_t, t \in \mathsf{St}(C)\} \cup Var(C)$ or else is some variable y_S, by construction of θ. If u is a variable y_S and v is a variable $y_{S'}$, then $\sum_{w \in E'_S} w =_{AC} \sum_{w \in E'_{S'}} w$. This can occur only if $E_S = E_{S'}$. Then, if $u \in E_{S'}$ and $x \in S'$, by definition, there is an index $i \in J_x$ such that $u =_{AC} u_i$. But $u \in E_S$, hence $x \in S$ (by the converse implication i the definition of E_S). It follows that $S = S'$. Now, if $u = \overline{x}\theta$ and v is some variable y_S, by construction, $E_S \cap \{t\sigma \downarrow, t \in \mathsf{St}(C)\} = \emptyset$ since $\forall i \in J_x, \forall t \in \mathsf{St}(C), u_i \neq_{AC} t\sigma \downarrow$. Then $u\sigma' \downarrow =_{AC} y_S\sigma' \downarrow$ implies

$$\sum_{y \in G', y = \widetilde{x_t}} \epsilon_{x,y}(t\sigma) \downarrow \oplus \sum_{i \in J_x} u_i =_{AC} \sum_{w \in E_S} w$$

which in turn implies that $\epsilon_{x,y} = 0$ for all $y \in G'$, hence $J_x = I_x$. Finally, from $E_S \cap E_{S'} = \emptyset$ for $S \neq S'$, we deduce $x = y_S$. Similarly, when $u = \overline{x}\sigma_0$ and $v = \overline{x'}\sigma_0$, we can conclude that $u =_{AC} v$. When $u = \overline{x_t}\theta$, and $v = \overline{x}\sigma_0$, we have also $\epsilon_{x,y} =$, except when $y = \widetilde{x_t}$ and $\epsilon_{x,S} = 0$ whenever $x \in S$. At last, when $u = \overline{x_t}$ and $v = \overline{x_{t'}}$, we must have $t =_S t'$, hence $x_t\theta =_{AC} x_{t'}\theta$.

Consider now two terms in $\mathsf{SS}(C\theta)$: $u \downarrow = u_1 \oplus \ldots \oplus u_n$ and $v \downarrow = v_1 \oplus \ldots \oplus v_m$ with $u_i, v_i \in \mathsf{St}(C\theta)$. We may assume that we removed on both sides identical terms so that $u_i \neq v_j$ for all i, j. Assume by contradiction that there is at least one term left. We may assume w.l.o.g that $m = 0$ (put all terms on the left). We want to show that $u\sigma' \downarrow = u_1\sigma' \downarrow \oplus \ldots \oplus u_n\sigma' \downarrow$ (i.e. there is no top redex), which will yield a contradiction. We have already seen that, for $u_i, u_j \in \mathsf{St}(C\theta)$, $u_i\sigma' \downarrow \neq_{AC} u_j\sigma' \downarrow$ whenever $u_i \neq_{AC} u_j$. So, the only remaining potential top redex comes from a variable instance: u_i is a variable y_S, $y_S\sigma' = \sum_{w \in E_S} w$ and, for some $w_0 \in E_S$, either $w_0 \in E_{S'}$ or $w_0 = u_j\sigma' \downarrow$. Both cases are impossible by construction. □

D Proof of lemma 6

Lemma 6

- *Every σ which is a collapse-free solution of C, is also a collapse free solution of some C' such that $C \rightsquigarrow_2 C'$.*
- *for every collapse free solution σ of C, there is some C' such that $C \rightsquigarrow_2 C'$ and, for every $T \Vdash u \in C'$, for every $v \in \mathsf{St}(T)$ such that $T\sigma \vdash v\sigma$ is derivable, then either $v \in T$ or $T \vdash v$ is not derivable.*
- *If $C \rightsquigarrow_2 C'$, then $\mathsf{St}(C') \subseteq \mathsf{St}(C)$*
- *If $C \rightsquigarrow_2 C'$, then every solution of C' is a solution of C*

Proof: If σ is a collapse-free solution of C, choose for S_i the set of terms t in $\mathsf{St}(T_i)$ such that $T_i\sigma \vdash t\sigma$ is derivable. We have $S_i \subseteq S_{i+1}$ since $T_i \subseteq T_{i+1}$. Choose for $>_i$ the

derivability ordering: $t >_i t'$ if $T_i\sigma \vdash t'\sigma$ occurs in the proof of $T_i\sigma \vdash t'\sigma$. Then, by construction, σ is also a solution of $T_i \cup \{v \in S_i \mid v <_i u\} \Vdash u$. Closure by $\oplus$ does not change the derivability and all substitutions are solutions of $T \Vdash u$ if $T \vdash u$ is derivable. □

E Proof of lemma 7

We start with a few auxilliary lemmas:

Lemma 11 *If P is a normal proof of $T \vdash u$ and (either u is not headed with $\oplus$ or u is not obtained by (GX)) and (u is not obtained by Encryption or Pairing), then there is a branch of P on which only Decryption, Unpairing and ((GX) yielding a term not headed with $\oplus$) are applied. In particular, $u \in \mathsf{St}(v)$ for every $T \vdash v$ of this branch.*

Proof: We prove the result by induction on the size of P. Consider the last rule which is applied. If it is a decryption (the reasoning is similar for unpairing):

$$\frac{T \vdash [u]_v \quad T \vdash v}{T \vdash u}$$

By definition of a normal proof, $[u]_v$ is not obtained by an encryption (and of course not by a pairing) and is not headed with $\oplus$. We can simply apply the induction hypothesis.

If the last rule is a (GX), yielding a term not headed with $\oplus$:

$$\frac{T \vdash u_1 \ldots T \vdash u_n}{T \vdash u}$$

$u \in \mathsf{St}(u_i)$ for some i and $T \vdash u_i$ is not obtained by (GX). Moreover, since $u_i \neq u$, u_i must be headed with $\oplus$, hence not obtained by encryption or pairing. Then, we apply the induction hypothesis. □

Lemma 12 *Let C be a constraint obtained after the second transformation step, T be a left member of the constraint, $u \in \mathsf{St}(C)$, and σ be a collapse-free solution of C.*

If $u\sigma \oplus w \notin T\sigma$ and P is normal proof of $T\sigma \vdash u\sigma \oplus w$ (w is possibly 0) without internal nodes belonging to $T\sigma$ and (either $u\sigma \oplus w$ is not headed with $\oplus$ or $u\sigma \oplus w$ is not obtained by (GX)) and ($u\sigma \oplus w$ is not obtained by Encryption or Pairing), then there is a term $y \oplus v \in T$ such that y is a variable and

$$\begin{cases} y\sigma = v_1 \oplus v_2 \\ u\sigma \in \mathsf{St}(v_1) \\ u\sigma \notin \mathsf{St}(v_2 \oplus v\sigma) \\ T\sigma \vdash v_2 \oplus v\sigma \end{cases}$$

with v_2, v possibly 0.

Proof: We apply lemma 11 and consider the longest branch such that $u\sigma \in \mathsf{St}(t)$ for every $T\sigma \vdash t$ on the branch, We consider the first rule applied on this branch: its a decryption, an unpairing or (GX) yielding a term not headed with $\oplus$. If it is a decryption (unpairing is similar):

$$\frac{T\sigma \vdash [u_1]_{u_2} \quad T\sigma \vdash u_2}{T\sigma \vdash u_1}$$

and $[u_1]_{u_2} \in T\sigma$, $u\sigma \in \mathsf{St}(u_1)$.

$[u_1]_{u_2}$ cannot be an instance of some term $[u_1']_{u_2'} \in T$: if it was the case, we would have $u_1' \in T$ by lemma 6 and there would be an internal node of P labeled with a term in $T\sigma$. It follows that $[u_1]_{u_2} = x\sigma$ for some $x \in T$. Then the lemma is proved, choosing $v = v_2 = 0$.

Consider now that the first rule is (GX) yielding a term not headed with $\oplus$:

$$\frac{T\sigma \vdash u_0 \oplus v_1 \quad T\sigma \vdash u_1 \ \ldots \ T\sigma \vdash u_n}{T\sigma \vdash v_1}$$

and $u_0 \oplus v_1 \in T\sigma, u\sigma \in \mathsf{St}(v_1)$, v_1 not headed with $\oplus$, $u_1, \ldots, u_k \in T\sigma$ and $u_{k+1}, \ldots, u_n \notin T\sigma$. By maximality of the branch, for every $i > k$, $u\sigma \notin \mathsf{St}(u_i)$. Moreover, if $u_i = u_i'\sigma$ for every i, by closure of T under $\oplus$, we may assume that, for every pair of distinct indices $i, j \leq k$, $(u_i' \oplus u_j') \downarrow = u_i' \oplus u_j'$. Now, since $u\sigma \notin \mathsf{St}((u_0' \oplus \ldots \oplus u_k')\sigma \downarrow) = \mathsf{St}((u_{k+1} \oplus \ldots \oplus u_n) \downarrow)$, and since (by collapse-freeness and construction) $(u_0' \oplus \ldots \oplus u_k')\sigma \downarrow = u_0'\sigma \oplus \ldots \oplus u_k'\sigma$, $u\sigma \notin \mathsf{St}(u_i)$ for every i.

Then, as before, since no internal node is an instance of some term in T, in particular $v_1 \notin T\sigma$. This implies that there is a term $y \oplus v \in T$ such that $(y \oplus v)\sigma = u_0 \oplus v_1$ and $y\sigma = v_1 \oplus v_2$. Finally, $u\sigma \notin \mathsf{St}(u_0) = \mathsf{St}(v_2 \oplus v\sigma)$.

Also, $(u_0 \oplus u_1 \oplus \ldots \oplus u_n) \downarrow = 0$, hence $v\sigma \oplus v_2 = u_0 = (u_1 \oplus \ldots \oplus u_n) \downarrow$, which implies that $T\sigma \vdash v\sigma \oplus v_2$. □

Lemma 7 *If σ is a collapse free solution of C (and C is obtained by the previous transformation steps), then there is a C' such that $C \rightsquigarrow C'$ by the rules of figure 6 and C' is a solved form.*

Proof: We prove actually a slightly stronger property, strengthening the definition of solved forms, imposing that $\succeq_{occ}$ is the least relation $\succeq$ satisfying the following properties:

1. $u \succeq v$ whenever $v\sigma \in \mathsf{St}(u\sigma)$
2. $x_u \succeq u$
3. $x_u \succeq y$ if $y \in Var(v_u)$

4. $u \succeq u$

5. $u \succeq v$ and $v \succeq w$ implies $u \succeq w$

Note that, if we have such an ordering then condition 3 of solved forms is automatically satisfied thanks to 1 above. We only have to focus on the two other properties of solved forms.

Let σ be a collapse-free solution of C. For every $t \in \mathsf{SS}(C)$ such that, for some left member T of C, $T\sigma \vdash t\sigma$ is derivable, we fix a minimal size normal proof $P_{T,t}$ of $T\sigma \vdash t\sigma$. Consider then the rules of figure 6; we apply these rules only when σ is a solution of the right sides of the rule and, moreover, $P_{T,v}$ is strictly smaller (in size) than $P_{T,u}$ for any $T \Vdash u$ on the left and any $T \Vdash v$ on the corresponding right side. Then the rules of figure 6 are terminating since the multiset of the sizes of proofs $P_{T,t}$ for constraints $T \Vdash t \in C$ is strictly decreasing. Hence it suffices to prove the lemma for constraints C to which it is impossible to apply any rule without loosing the solution σ or non-decreasing some proof $P_{T,t}$. Then we prove the result by induction on the number of elementary constraints in C.

In the base case, C is empty, hence in solved form.

Consider an elementary constraint $T \Vdash u$ in C such that T is maximal (w.r.t. inclusion) and, among these constraints, one for which u is not a variable, if it exists. Then T contains all left hand sides of elementary constraints in C since the left hand sides are totally ordered by inclusion. We apply the induction hypothesis on $C \setminus \{T \Vdash u\}$, which we assume in solved form, with an ordering $\succeq$ on $\mathsf{St}(C)$ defined as above.

If u is a variable, we only have to check property 2 of solved forms. By choice of T, if $T, T' \Vdash u_1 \oplus \ldots \oplus u_n \in C$, then T' must be empty, $n = 1$ and u_1 is a variable y. Then, by lemma 6, either $u \notin Var(T)$ or $y \notin Var(T)$ (if they are both in $Var(T)$ we would have guessed an ordering between u and y and get either $T \cup \{u\} \Vdash y$ or $T \cup \{y\} \Vdash u$). Then, by an appropriate choice of u, we may assume that $u \notin Var(T)$, hence is distinct from x_y (if the latter is defined).

Assume now that u is not a variable. Consider a normal proof of $T\sigma \vdash u\sigma$. We may assume w.l.o.g. that this proof does not contain any internal node labeled with a term in $T\sigma$. Consider now the last rule applied in this proof.

If it is a pairing or an encryption, then $T \Vdash u$ can be transformed by the corresponding Pairing or Encryption rule of figure 6, and σ is a solution of the new constraint. We discarded this possibility.

If it is a decryption or an unpairing, either x_u is already defined in $C \setminus \{T \Vdash u\}$ and, by maximality of T, there is nothing to do: the same ordering satisfies the desired conditions for C. Otherwise, we use lemma 12: there is a term $x \oplus v \in T$ such that $x\sigma = v_1 \oplus v_2, u\sigma \in \mathsf{St}(v_1)$ (hence in $\mathsf{St}(x\sigma)$, which implies $x \succ u$ by the rule 1 above), $u\sigma \notin \mathsf{St}(v_2 \oplus v\sigma)$ (hence $x\sigma \notin \mathsf{St}(y\sigma)$ for any $y \in Var(v)$) and $T\sigma \vdash v_2 \oplus v\sigma$.

We choose for x the maximal variable (w.r.t. $\succeq$) such that $x \succ u$, $x \oplus v \in T$ and, for every variable $y \in Var(v)$,$x\sigma \notin \mathsf{St}(y\sigma)$.

Assume by contradiction that $y \succ x$ for some variable $y \in Var(v)$. Since $\succeq$ is the minimal relation satisfying properties 1-5 there must be a secquence

$$y \succ t_n \succ \ldots \succ t_1 \succ x$$

such that each of the above inequalities follows from 1, 2 or 3 in the definition of the ordering. Moreover, since $x\sigma \notin \mathsf{St}(y\sigma)$, at least one inequality does not follow from 1. This implies that there is a variable x_w such that $y \succeq x_w \succ c$. But, in that case, by definition and by maximality of T, there is a term $x_w \oplus v_w \in T$ such that $x_w \succ z$ for every variable $z \in Var(v_w)$. In particular, for any such variable, $x_w\sigma \notin \mathsf{St}(z\sigma)$. Since $x_w \succ x \succ u$, this contradicts the maximality of x.

So, in any case, $y \not\succ x$. Moreover, $y \neq x$ since $x\sigma \notin \mathsf{St}(y\sigma)$. Then $y \not\succeq x$ and setting $x_u = x$, $x_u \succ u$, $x_u \succ y$ for $y \in Var(v)$, we do not create any cycle in the ordering.

If it is (GX),

$$\frac{T\sigma \vdash v_1 \quad \ldots \quad T\sigma \vdash v_n}{T\sigma \vdash u\sigma}$$

and $u\sigma = w_1 \oplus \ldots \oplus w_m$ where each w_i is not headed with $\oplus$. The terms w_i can be split into three categories:

1. $w_i = u'\sigma$, $u = u' \oplus u''$ and $w_i = v_j$ for some j
2. $w_i = u'\sigma$, $u = u' \oplus u''$ and $v_j = w_i \oplus v'_j$ for some j
3. There is a variable z such that $u = z \oplus u'$ and $z\sigma = w_i \oplus w'_i$.

In the first case, we can simplify the constraint using the **(S)** rule of figure 6: σ is still a solution of $T \Vdash u'$ and of $T \Vdash u''$.

Consider now the second case. If $x_{u'}$ is already defined, there is nothing to do. Otherwise, if $v_j \notin T\sigma$, we use lemma 12, there is a term $x \oplus v \in T$ such that $v_j \in \mathsf{St}(x\sigma)$ (hence $u'\sigma \in \mathsf{St}(x\sigma)$ and therefore $x \succ u'$) and $v_j \notin \mathsf{St}(v\sigma)$ (hence $u'\sigma \notin \mathsf{St}(v\sigma)$). Then, we proceed exactly as in the case of decryption or unpairing, replacing u with u'. If $v_j \in T\sigma$, then we can apply the (**X**) rule: the proof of $T\sigma \vdash v_1 \oplus \ldots \oplus v_{j-1} \oplus v_{j+1} \oplus \ldots \oplus v_n$ has a strictly smaller size.

Consider finally the third case. If x_z is already defined, there is nothing to do. If there is already a term $x_u \oplus v \in T$ such that $x_u \succ z$, it is sufficient to set $x_z = x_u$ and the

ordering remains unchanged. We discard these cases in the following. Let $z\sigma = t_1 \oplus \ldots \oplus t_p$. If every t_i is equal to some v_j (and if u is not the variable z itself), then we can use the **(S)** rule of figure 6, since $T\sigma \vdash z\sigma$. Otherwise, consider a maximal t_j such that t_j is not some v_p. By lemma 12, for every t_j such that t_j is not any v_p, there is a term w_j, there is a term $x_j \oplus v'_j \in T$ such that $x_j\sigma = w'_1 \oplus w'_2$, $(v_k =)t_j \oplus w_j \in \mathsf{St}(w'_1)$ and $t_j \notin \mathsf{St}(w'_2 \oplus v\sigma)$.

Now, let x_0 be the maximal variable (w.r.t. $\succeq$) such that ($x_0 \succ t_j \oplus w_j$ or $x_0\sigma = t_j \oplus w_j$), $x_0 \oplus v_0 \in T$ and for every variable $y \in Var(v_0)$, $x_0\sigma \notin \mathsf{St}(y\sigma)$.

Let us show first that $x_0\sigma \notin \mathsf{St}(z\sigma)$. Assume by contradiction that this is the case. Then there are two cases:

- either $x_0\sigma \in \mathsf{St}(t_k)$ for some k, in which case $t_k \succ x_0$ since $x_0\sigma \neq t_k$ unless $t_k = z\sigma$ (by collapse-freeness). Now, $x_0\sigma \succeq t_j$, which contradicts the maximality of t_j
- or else $x_0\sigma = z\sigma$. In that case, we must have $x_0 = z$ since σ is collapse-free. Then, by definition of $z\sigma$ and since $t_j \in \mathsf{St}(w'_1)$, $t_j \notin \mathsf{St}(w'_2)$, we must have $w'_1 = t_j$. Then, by normality of the proof, it must have the form:

$$(GX)\ \frac{T\sigma \vdash t_j \quad \vdots}{T\sigma \vdash t_j \oplus \ldots}$$

  In particular, thanks to lemma 12, $z\sigma = t_j$, and therefore $z \in T$. In this case, we can apply **(X)** since σ is a solution of $T \Vdash u'$.

We conclude that $x_0\sigma \notin \mathsf{St}(z\sigma)$.

Assume now, by contradiction, that $z \succ x_0$. Since $x_0\sigma \notin \mathsf{St}(z\sigma)$, by a reasoning similar to the decryption case, ther must be a variable x_w such that

$$z \succeq x_w \succ x_0 \succ t_j \oplus w_j$$

This contradicts the maximality of x_0. Hence $z \not\succeq x_0$.

Finally, if we had $y \in Var(v_0)$ such that $y \succeq z$, we would again contradict the maximality of x_0, thanks to $x_0\sigma \notin \mathsf{St}(y\sigma)$.

Since $z \not\succeq x_0$, $y \not\succeq x_0$ for $y \in Var(v_0)$, we can set $x_z = x_0$ and $v_z = v_0$, and $x_z \succ y$ for $y \in Var(v_z)$; the new least relation $\succeq$ is still an ordering and the first condition of solved forms is satisfied.

□

F Proof of lemma 8

Lemma 8 *Every solved form has a solution.*

Proof: We prove the lemma by induction on the number of variables. If there is no variable, C must be empty, hence valid.

Now, if C contains at least one variable, consider a minimal variable x w.r.t. $\geq_{occ}$. We are going to construct a term t, which will be assigned to x. If $x \neq x_u$ for every u, then we let $t = 0$. Otherwise, consider a sequent $T \Vdash u_1 \oplus \ldots \oplus u_n$ such that $x_{u_i} = x$ and such that T is minimal (w.r.t. inclusion). Let u_0 be v_{u_i} (or 0 if $x \in T$). Consider finally all terms $v_1, \ldots, v_m$ such that $x_{v_i} = x$. We claim that $u_0, v_1, \ldots, v_m$ are ground terms. For u_0, by property 1b of solved forms, $x >_{occ} y$ whenever $y \in Var(u_0)$, which implies that u_0 does not contain any variable, by minimality of x. For $v_1, \ldots, v_n$, by property (1a) of solved forms, $x >_{occ} v_i$ for every i and, by property (3) of solved forms, $v_i >_{occ} y$ for every variable $y \in Var(v_i)$. Again, by minimality of x, this implies that every v_i is ground.

We let $t = u_0 \oplus \langle u_0, < u_0, v_1, \ldots, v_m > \rangle$ where $< v_1, \ldots, v_m >$ is defined by induction on m: $< v_1 > \stackrel{\text{def}}{=} v_1$, $< v_1, \ldots, v_m > \stackrel{\text{def}}{=} \langle v_1, < v_2, \ldots, v_m > \rangle$.

For every $T_1 \Vdash w_1 \oplus \ldots \oplus w_k$ in C, $T_1\{x \mapsto t\} \vdash w_j$ is derivable whenever $x_{w_j} = x$: by minimality of T and since the left hand sides of sequents are totally ordered by inclusion, we must have $T \subseteq T_1$, hence $x \oplus u_0 \in T_1$, which implies that $T_1\{x \mapsto t\} \vdash v_k$ is derivable for every k.

On the other hand, by the second property of solved forms, if $T_1 \Vdash x \oplus u \in C$ for some T_1, then every T' such that $T' \Vdash w_1 \oplus \ldots \oplus w_k \in C$ and $x_{w_i} = x$ for some i is contained in T_1. In particular, $x \oplus u_0 \in T_1$ or else $t = 0$. It follows that $T_1\{x \mapsto t\} \vdash u_0$ is derivable or $t = 0$. In both cases, $T_1\{x \mapsto t\} \vdash t$ is derivable.

Now, apply the substitution $\{x \mapsto t\}$ to C, remove from the right sides all w_j such that $x_{w_j} = x$, remove sequents with empty right hand sides and remove x from sequents $T \Vdash x \oplus u$. We get a constraint C' which is again in solved form: we let $\geq'_{occ}$ be defined by $u' \geq'_{occ} v'$ *iff* $u' = u\{x \mapsto t\}\downarrow$, $v' = v\{x \mapsto t\}\downarrow$ and $u \geq_{occ} v$. Consider now a sequent $T\{x \mapsto t\} \Vdash u'$ in C'. Either u' is a variable or a sum $u'_1 \oplus \ldots \oplus u'_m$. In the latter case, each $u'_i = u_i\sigma \downarrow$ with $u_i \neq x$. Then there is a term $x_{u_i} \in Var(T)$ such that $x_{u_i} >_{occ} u_i$, hence $x_{u_i}\{x \mapsto t\} >'_{occ} u'_i$. But $x_{u_i} \neq x$, hence $x_{u_i} >'_{occ} u'_i$. Moreover, either $x_{u_i} \in T'$ or else $x_{u_i} \oplus v'_{u_i} \in T'$ with $v'_{u_i} = v_{u_i}\{x \mapsto t\}$. Since t is ground, for every variable $y \in Var(v'_{u_i})$, $y \in Var(v_{u_i})$ and $y \neq x$. It follows from $x_{u_i} >_{occ} y$ that $x_{u_i} >'_{occ} y$. Then it suffices to choose $x_{u'_i} = x_{u_i}$ and $v_{u'_i} = v'_{u_i}$: the first condition of solved forms is met. As for the second condition: if $T\{x \mapsto t\} \Vdash y \oplus u\{x \mapsto t\} \in C'$, and $T\{x \mapsto t\}, T'\{x \mapsto t\} \Vdash w' \oplus v'$ then, by definition, $x_{v'} = x_v \neq y$. The third condition follows from the fact that t is ground.

Then, we apply the induction hypothesis to C': there is a solution σ' to C'. We let $\sigma = \sigma' \circ \{x \mapsto t\}$. σ is a solution of C. □

Note however that, in the proof, we did not construct a collapse-free solution. We know however by the other lemmas, that, in this case, some other constraint obtained by the previous transformation steps does have a collapse-free solution.

G Example of transformation step 1 in the case of Abelian groups

Example 7
Consider the simple term $2x+3y$ and assume that free symbols consist only of constants. The equivalent of step 3 should guess possible interactions between the three copies of y and the two copies of x, yielding a reduction. This may occur only if $x\sigma = \alpha a + u$ and $y\sigma = \beta I(a) + u$, with $\alpha, \beta > 1$. First, we have to guess whether there are remaining copies of a after reduction and whether they come from $x\sigma$ or from $y\sigma$: split first x into four parts $x = x_0 + x_1 + x_2 + x_3 + x_4$ (a part containing constants occurring in $x\sigma$ and not in $y\sigma$, a part containing constants which yield a simplification, but only part of them disappear by simplification, a part containing constants which disappear after reduction, a part which disappears by simplicication, though the opposite constants remain in the normal form). Consider for instance the second part (x_1). $x_1\sigma = \alpha_1 a_1 + \ldots + \alpha_n a_n$ (for simplicity, assume $\alpha_i > 0$ for every i and $y_1\sigma = \beta_1 a_1 + \ldots + \beta_n a_n$ with $\beta_i < 0$ for every i. Also, $2\alpha_i + 3\beta_i > 0$ for every i by construction. We may assume that we put in x_1 the minimal amount of each constants which was sufficient to capture all negative occurrences: $2(\alpha_i - 1) + 3\beta_i < 0$. It follows that $2\alpha_i + 3\beta_i = 1$ for every i: β_i is odd. Let $\beta_i = -2\gamma_i - 1$. $2\alpha_i = 1 + 6\gamma_i + 3$, hence $\alpha_i = 2 + 3\gamma_i$. So, we can split x_1: $x_1 = 3x_{11} + 2x_{12}$ and $y_1 = -2x_{11} - y_{12}$.