On the Automatic Analysis of Recursive Security Protocols with XOR*

Ralf Küsters[1] and Tomasz Truderung[2]

[1] ETH Zurich
ralf.kuesters@inf.ethz.ch
[2] University of Kiel, Wrocław University
tomasz.truderung@ii.uni.wroc.pl

Abstract. In many security protocols, such as group protocols, principals have to perform iterative or recursive computations. We call such protocols *recursive protocols*. Recently, first results on the decidability of the security of such protocols have been obtained. While recursive protocols often employ operators with algebraic, security relevant properties, such as the exclusive OR (XOR), the existing decision procedures, however, cannot deal with such operators and their properties. In this paper, we show that the security of recursive protocols with XOR is decidable (w.r.t. a bounded number of sessions) for a class of protocols in which recursive computations of principals are modeled by certain Horn theories. Interestingly, this result can be obtained by a reduction to the case without XOR. We also show that relaxing certain assumptions of our model lead to undecidability.

1 Introduction

In group protocols and other classes of security protocols a protocol step performed by a principal (i.e., receiving a message and then sending a message) typically involves recursive or iterative computation. We will refer to such protocols by *recursive protocols*, in contrast to *non-recursive protocols* where the computation performed in one protocol step is simple and does not require recursion. Many, in fact, most of the recursive protocols proposed in the literature employ operators, such as Diffie-Hellman exponentiation and exclusive OR (XOR), which have algebraic, security relevant properties (see, e.g., [11, 12, 7]). The present work is concerned with the automatic security analysis of such protocols. While recently first results on the decidability of the security (more precisely, the secrecy property) of recursive protocols have been obtained [8, 9, 14], these results do not take into account operators with algebraic properties (see also the related work).

The attacks on recursive protocols presented in the literature illustrate that dealing with algebraic properties of operators is security relevant (see, e.g., [11, 12, 7]). One example is the Recursive Authentication (RA) protocol proposed by Bull and Otway [1]. In this protocol, a key distribution server receives a list of

* An extended abstract of this work appears in STACS 2007.

(arbitrary many) requests of pairs of principals who want to establish session keys among them. The server processes this list iteratively, generates the session keys, and then distributes them. In [10], Paulson proved that the RA protocol is secure under the assumption that session keys are distributed using (ideally) secure encryption. However, Ryan and Schneider [12] showed that there is an attack on the protocol if XOR is used to distribute keys, which in fact was the original proposition by Bull and Otway: In the attack by Ryan and Schneider, the adversary is given one session key generated by the server and using this key he can obtain all other session keys by chaining messages via XOR.

Contribution of this work. In this paper, we extend the model for recursive protocols proposed in [14], henceforth called the *Horn theory model*, by adding XOR (along with its algebraic properties). In the Horn theory model, recursive/iterative computations performed by principals in one protocol step are modeled by certain Horn theories, hence the name. While in models for non-recursive protocols XOR can be added without losing decidability of security (w.r.t. a bounded number of sessions) [2, 6]—security even remains NP-complete just as in the case without XOR [2]—for recursive protocols things are more involved. We show that a naïve extension of the Horn theory model by XOR leads to undecidability of security. (As a byproduct we also obtain undecidability in case complex keys are used in the Horn theory model, a fact that has not been observed before.) More precisely, we obtain undecidability in case principals may conjoin arbitrary messages received from the network by XOR. Conversely, we show decidability in case principals may only conjoin a fixed message, i.e., a message that does not depend on messages received from the network, with a message that depends on messages received from the network. We call protocols which only contain such principals $\oplus$-linear. From a practical point of view, $\oplus$-linear protocols are sufficient in many cases, e.g., for the RA protocol and other protocols [12, 2, 7]. We emphasize that we do not constrain the intruder in its ability to conjoin messages by XOR.

The technique used to obtained decidability is very different to the one in [14]. In fact, the main part of our proof is to reduce the security problem in the Horn theory model with XOR to the one without XOR. More precisely, we first prove certain properties of attacks involving XOR. Based on these properties we then reduce the security problem to the case without XOR, which by [14] is decidable. In the reduction we use the ability of principals to perform recursive computations in order to mimic applications of the XOR operator.

Further Related Work. For non-recursive protocols decidability of security (w.r.t. a bounded number of sessions) was shown for several operators with algebraic properties, e.g., XOR [2, 6], Diffie-Hellman Exponentiation [3, 13], and commuting public-key encryption [4]. However, the models employed in these works cannot handle recursive computations of principals, such as the computation of the server in the RA protocol. The techniques differ as well: Due to the absence of recursive computations, the reduction technique developed in the present work is not applicable in these models. Conversely, the techniques for

bounding the size of attacks and the constraint solving techniques employed in [2, 6, 3, 13, 4] cannot immediately be applied to recursive protocols.

In [8, 9], transducers were used to model recursive computations of principals. The expressivity of these transducer-based models is orthogonal to the Horn theory model: While the transducer-based models allow to output messages of complex structure, in the Horn theory model only lists (or sets) of messages of a more simple structure can be produced. The main disadvantage of the transducer-based model is that, unlike the Horn theory model, messages cannot be tested for equality without losing decidability. This, as already observed in [9], immediately implies that security is undecidable in the transducer-based model with XOR (or Diffie-Hellman exponentiation) since these operators allow for (implicit) equality tests between arbitrary messages. In the transducer-based model, even one equality test (or alternatively, one application of the XOR operator) suffices for the undecidability.

Horn theories have also been used for the automatic analysis of *non-recursive* protocols (see, e.g., [5, 15] and references therein). The results and techniques employed in these works are very different to the ones presented here: The main goal of these works, which also consider operators with algebraic properties, is automatic protocol analysis w.r.t. an *unbounded* number of sessions, where, however, the intruder knowledge is over-approximated.

STRUCTURE OF THE PAPER. In the following section, we present our protocol and intruder model, with an example presented in Section 3. The undecidability and decidability results are stated in Section 4 and 5, respectively. We conclude in Section 6. The appendix provides further details and proofs.

2 The Protocol and Intruder Model

In this section, we introduce our protocol and intruder model, including messages, principals, protocols, and attacks, along the lines of [14] where, unlike the model in [14], here messages may contain the exclusive OR (XOR).

HORN THEORIES. Let Σ be a finite signature, V be a set of variables, and $\mathcal{T}$ denote the set of terms over Σ and V. *Ground terms* are terms without variables. Substitutions are defined as usual. The application of a substitution σ to a term t is denoted by $t\sigma$. Substitutions are defined on sets of terms and atoms (see below) in the obvious way. For a unary predicate q and a (ground) term $t \in \mathcal{T}$ we call $q(t)$ a *(ground) atom*. For a set S of terms we write $q(S)$ for the set $\{q(s) \mid s \in S\}$ of atoms. Let $\sim$ be a congruence relation over $\mathcal{T}$. We write $q(t) \sim q'(t')$ if $q = q'$ and $t \sim t'$. A *(unary) Horn theory* T is a finite set of Horn clauses of the form $a_1, \ldots, a_n \Rightarrow a_0$ with atoms a_i for every i. Given a set of ground atoms A and a ground atom a, we say that a can be derived from A w.r.t. T (written $A \vdash_T a$) if there exists a *derivation* for a from A using T, i.e., there exists a sequence $b_1, \ldots, b_l$ of ground atoms such that $b_l \sim a$ and for every $i \in \{1, \ldots, l\}$ we either have $b_i \in A$ or there exists a substitution σ and a Horn clause $a_1, \ldots, a_n \Rightarrow a_0$ in T such that $a_0\sigma \sim b_i$ and for every $j \in \{1, \ldots, n\}$ there exists $k \in \{1, \ldots, i-1\}$ with $a_j\sigma \sim b_k$.

MESSAGES. Let $\mathcal{A}$ be a finite set of constants (also called *atomic messages*), such as principal names, nonces, and keys, and let $\mathcal{K}$ be a subset of $\mathcal{A}$ (the set of keys). We assume that $0, \mathsf{Sec} \in \mathcal{A}$ and that there is a bijection $\cdot^{-1}$ on $\mathcal{K}$ which maps every public (private) key k to its corresponding private (public) key k^{-1}. Let $\Sigma_{\mathcal{A}}$ (or simply Σ) be a finite signature consisting of all constants from $\mathcal{A}$, the unary function symbol $\mathsf{hash}(\cdot)$ (*hashing*), and the following binary function symbols: $\langle \cdot, \cdot \rangle$ (*pairing*), $\{\cdot\}.$ (*symmetric encryption*), $\{\!|\cdot|\!\}.$ (*public key encryption*), and $\oplus$ (*exclusive OR*).

The set of terms over Σ and V is defined by the following grammar:

$$\mathcal{T} ::= \mathcal{A} \mid V \mid \mathsf{hash}(\mathcal{T}) \mid \langle \mathcal{T}, \mathcal{T} \rangle \mid \{\mathcal{T}\}_{\mathcal{K}} \mid \{\!|\mathcal{T}|\!\}_{\mathcal{K}} \mid \mathcal{T} \oplus \mathcal{T}.$$

Note that we assume atomic keys, i.e., keys used to encrypt messages are required to be constants. We denote by $\mathit{Var}(t)$ the set of variables occurring in t.

Ground terms, i.e., terms without variables are called *messages*. To model the algebraic properties of XOR, we consider the congruence relation $\sim$ on $\mathcal{T}$ induced by the following equational theory:

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

For example, we have that $a \oplus b \oplus \{\!|0|\!\}_k \oplus b \oplus \{\!|c \oplus c|\!\}_k \sim a$. (Due to the associativity of $\oplus$ we often omit brackets and simply write $a \oplus b \oplus c$ instead of $(a \oplus b) \oplus c$ or $a \oplus (b \oplus c)$.)

PRINCIPALS AND PROTOCOLS. A *protocol step* consists of a protocol rule and a send program. A *protocol rule* is of the form $t \to q(s)$ where $t, s \in \mathcal{T}$ and q is some unary predicate symbol. A *send program* Φ is a unary Horn theory where every Horn clause is of one of the following forms:

$$q'(t) \Rightarrow q''(x) \qquad \text{with } x \in \mathit{Var}(t), \tag{1}$$

$$q'''(s) \Rightarrow I(s') \qquad \text{with } \mathit{Var}(s') \subseteq \mathit{Var}(s), \tag{2}$$

where q', q'', q''' are arbitrary (not necessarily different) unary predicate symbols, t is a linear term (i.e., every variable in t occurs at most once) which does not contain the symbol $\oplus$ and I is a distinguished unary predicate symbol, which will model the network, and hence, the intruder; the terms s and s' may be non-linear and may contain $\oplus$. Intuitively, clauses of the form (1), called *push clauses*, allow a principal to recursively traverse a term from top to bottom (e.g., process a list). Clauses of the form (2), called *send clauses*, are used by a principal to perform checks on messages (by matching them against s) and to output messages on the network, as will be clearer from the following definition:

For a ground atom $q(m)$, we define the set of *terms sent using* Φ by

$$[\![q(m)]\!]_{\Phi} = \{m' : q(m) \vdash_{\Phi} I(m')\}.$$

Now, the intuition behind a protocol step which consists of a protocol rule $t \to q(s)$ and a send program Φ is that a principal, after having received a term $t\theta$, for some ground substitution θ, sends all the terms from the set $[\![q(s\theta)]\!]_\Phi$ on the network, i.e., to the intruder, by running the send program Φ.

The decidability result in [14] in the Horn theory model without XOR works if t in (1) is flat, i.e., is of the form $t = f(x_1, \ldots, x_n)$ where the variables x_i are not required to be different. We could also allow such terms in (1). However, (complex) linear terms are better suited for modeling protocols. It is easy to see that linear terms in (1) can be turned into flat form by using auxiliary predicate symbols.

A *principal* Π is a finite edge-labeled tree where every edge is labeled by a protocol step. If the protocol rule of a protocol step is of the form $t \to q(s)$, we require that every variable occurring in s also occurs in t or the left-hand side of a protocol rule preceding $t \to q(s)$ in the tree Π. We also assume, w.l.o.g., that the set of predicates used in send programs of different protocol steps are pairwise disjoint, except that I may be used in all of the send programs. The intuition is that if a principal waits at a node of the tree and receives a message, then she can apply one of the protocol steps whose left-hand side (i.e., the left-hand side of the corresponding protocol rule) matches with the incoming message, and after having run the corresponding send program, moves to the next node.

For a principal Π we call a sequence π of protocol steps a *run of* Π if π is a sequence of protocol steps obtained when traversing Π from the root to some node of Π (not necessarily a leaf).

A *protocol* P is a tuple $(\Pi_1, \ldots, \Pi_l)$ of principals Π_i. We assume, w.l.o.g., that the set of variables of protocol rules of different principals are disjoint.

Attack. The intruder is the standard Dolev-Yao intruder extended by the ability to apply the XOR operator [6, 2]. Formally, the intruder is modeled by the following Horn theory $T_\oplus$ where $k \in \mathcal{K}$ and $x, y \in V$, $x \neq y$:

$$
\begin{array}{lll}
I(x), I(y) \Rightarrow I(\langle x, y\rangle) & I(x), I(k) \Rightarrow I(\{x\}_k) & I(\{x\}_k), I(k) \Rightarrow I(x) \\
I(\langle x, y\rangle) \Rightarrow I(x) & I(x), I(k) \Rightarrow I(\{\!|x|\!\}_k) & I(\{\!|x|\!\}_k), I(k^{-1}) \Rightarrow I(x) \\
I(\langle x, y\rangle) \Rightarrow I(y) & I(x) \Rightarrow I(\mathsf{hash}(x)) & I(x), I(y) \Rightarrow I(x \oplus y)
\end{array}
$$

Given a protocol $P = (\Pi_1, \ldots, \Pi_l)$, a *protocol execution scheme* of P is a sequence of protocol steps $\pi = \pi_1, \ldots, \pi_n$ such that each π_i can be assigned to one of the principals $\Pi_1, \ldots, \Pi_l$ and such that, for every i, the subsequence of elements of π assigned to Π_i is a run of Π_i, i.e., π is an interleaving of runs of the Π_i.

Now, an *attack* on P is a pair (π, θ) where $\pi = ((t_i \to q_i(s_i), \Phi_i))_{i=1}^n$ is a protocol execution scheme of P and θ is a ground substitution of the variables in $Var(\{t_1, s_1, \ldots, t_n, s_n\})$ such that

$$
I(0), I([\![q_1(s_1\theta)]\!]_\Phi), \ldots, I([\![q_{i-1}(s_{i-1}\theta)]\!]_\Phi) \vdash_{T_\oplus} I(t_i\theta), \quad \text{for } i = 1, \ldots, n \tag{3}
$$

$$
I(0), I([\![q_1(s_1\theta)]\!]_\Phi), \ldots, I([\![q_n(s_n\theta)]\!]_\Phi) \vdash_{T_\oplus} I(\mathsf{Sec}) \tag{4}
$$

where $\Phi = \bigcup_{i=1}^{n} \Phi_i$ (recall that different send programs use disjoint sets of predicates, except that they all may use I). Condition (3) says that in every step of the protocol execution the intruder is able to derive the message expected by the respective principal and (4) says that at the end he is able to derive the secret Sec. Note that, w.l.o.g., initially the intruder only knows the constant 0: One can define a designated principal that expects to receive 0 and in return outputs messages the intruder is allowed to know, e.g., public keys. A protocol is called *insecure* if there exists an attack on it. Let $\textsc{Attack}_{general} = \{P \mid P \text{ is an insecure protocol}\}$ denote the corresponding decision problem.

3 An Example Protocol

To illustrate our model, we present a formal description of the Recursive Authentication (RA) Protocol [1]. In what follows, we abbreviate messages of the form $\langle m_0, \ldots, \langle m_{n-1}, m_n \rangle \cdots \rangle$ by $m_0, \ldots, m_n$ and messages of the form $\langle m, \mathsf{hash}(\langle k, m \rangle) \rangle$, i.e., a message m along with a keyed hash on m, by $\mathtt{hash}_k(m)$.

The key distribution server S of the RA protocol shares a long-term (symmetric) key with every principal and performs only one (recursive) protocol step in a protocol run. In this protocol step, S receives an a priori unbounded sequence of requests of pairs of principals who want to obtain session keys for secure communication and then generates so-called certificates which contain the session keys. An example of the kind of message S receives is

$$\mathtt{hash}_{K_c}(C, S, N_c, \mathtt{hash}_{K_b}(B, C, N_b, \mathtt{hash}_{K_a}(A, B, N_a, -))) \tag{5}$$

where N_c, N_b, and N_a are nonces generated by C, B, and A, respectively, and K_c, K_b, and K_a are the long-term keys shared between the server S and the principals C, B, and A, respectively. Recall that, for instance, $\mathtt{hash}_{K_a}(A, B, N_a, -)$ stands for the message $\langle \langle A, \langle B, \langle N_a, - \rangle \rangle \rangle, \mathsf{hash}(\langle K_a, \langle A, \langle B, \langle N_a, - \rangle \rangle \rangle \rangle \rangle) \rangle$. Message (5) consists of three requests and indicates that C wants to share a session key with S, B with C, and A with B. The constant "$-$" marks the end of the sequence of requests. We emphasize that messages sent to S may contain an arbitrary number of requests—which must be processed by S recursively. Now, given message (5), S processes the requests starting from the outermost. First, S generates two certificates for C, namely, $\langle C, S, K_{cs} \oplus \mathtt{hash}_{K_c}(N_c), \{C, S, N_c\}_{K_{cs}} \rangle$ and $\langle C, B, K_{bc} \oplus \mathtt{hash}_{K_c}(N_c), \{C, B, N_c\}_{K_{bc}} \rangle$ (from these certificates C can easily deduce K_{cs} and K_{bc} and check whether the encrypted messages have the expected form). In the same way, certificates for B and A are generated, where A only obtains one certificate (containing the session key for communication with B).

Formally, the protocol step performed by S is as follows, where we assume that $P_0, \ldots, P_n$ are the principals that may participate in the RA protocol, with $P_n = S$, and every P_i, $i < n$, shares a long-term key K_i with S: The protocol rule of S is simply $x \to q(x)$ and the send program consists of the following Horn

clauses, where $j \leq n$ and $i, i' < n$:

$$\begin{aligned} q(\langle\langle x_1, \langle x_2, \langle x_3, x_4\rangle\rangle\rangle, x_5\rangle) &\Rightarrow q(x_4) \\ q(\mathtt{hash}_{K_i}(P_i, P_j, x, -)) &\Rightarrow I(M_{i,j}) \\ q(\mathtt{hash}_{K_i}(P_i, P_j, x, \mathtt{hash}_{K_{i'}}(P_{i'}, P_i, x_1, x_2))) &\Rightarrow I(M'_{i,i'}) \\ q(\mathtt{hash}_{K_i}(P_i, P_j, x, \mathtt{hash}_{K_{i'}}(P_{i'}, P_i, x_1, x_2))) &\Rightarrow I(M_{i,j}) \end{aligned}$$

where $M_{i,j} = \langle P_i, P_j, K_{ij} \oplus \mathtt{hash}_{K_i}(x), \{P_i, P_j, x\}_{K_{ij}}\rangle$ and $M'_{i,i'} = \langle P_i, P_{i'}, K_{i'i} \oplus \mathtt{hash}_{K_i}(x), \{P_i, P_{i'}, x\}_{K_{i'i}}\rangle$. The server would also check whether the first request is addressed to it. This can easily be captured by using another predicate; however, for simplicity of presentation this is not modeled here. The model of the principals $P_0, \ldots, P_{n-1}$ of the RA protocol is rather standard as they do not need to perform recursive computations. We therefore omit their formal specification here.

4 Undecidability of the General Case

We prove the following theorem:

Theorem 1. *The problem* $\text{ATTACK}_{general}$ *is undecidable.*

Proof. The proof is by reduction from Post Correspondence Problem (PCP): An instance $\mathcal{I}$ of PCP is a non-empty sequence $(u_1, v_1), \ldots, (u_n, v_n)$ of pairs of words $u_i, v_i \in A^*$ over a finite alphabet A. A solution of $\mathcal{I}$ is a non-empty sequence $i_1, \ldots, i_l$ with $i_j \in \{1, \ldots, n\}$ such that $u_{i_1} \cdots u_{i_l} = v_{i_1} \cdots v_{i_l}$.

To encode $\mathcal{I}$, let $\mathcal{A}_\mathcal{I} = A \cup \{\bot, 1, \ldots, n, k, -\}$ denote the set of atomic messages that we use. For a word $u \in \mathcal{A}_\mathcal{I}^*$ and a term t over $\mathcal{A}_\mathcal{I}$, we define $t \cdot u$ inductively by $t \cdot u = t$ if $u = \varepsilon$ and $t \cdot u = \langle t, a\rangle \cdot v$ with $u = av$ for some $a \in \mathcal{A}_\mathcal{I}$ and $v \in \mathcal{A}_\mathcal{I}^*$.

A solution of $\mathcal{I}$ is encoded by what we call a solution sequence of $\mathcal{I}$. A *solution sequence* of $\mathcal{I}$ is a sequence $t_0, \ldots, t_l$ of messages using constants in $\mathcal{A}_\mathcal{I}$ where $t_0 = \langle \bot, \bot\rangle$, $t_l = \langle m, m\rangle$ for some message m over $\mathcal{A}_\mathcal{I}$, and for every $i \in \{0, \ldots, l-1\}$, if $t_i = \langle m, m'\rangle$, then $t_{i+1} = \langle m \cdot u_j, m' \cdot v_j\rangle$ for some $j \in \{1, \ldots, n\}$. It is easy to see that a PCP instance $\mathcal{I}$ has a solution iff there is a solution sequence for $\mathcal{I}$.

Given $\mathcal{I}$, we reduce checking whether $\mathcal{I}$ has a solution to checking whether there exists an attack on the protocol $P_\mathcal{I}$: $P_\mathcal{I}$ consists of one principal only and this principal performs only one protocol step where the protocol rule of this step is simply $x \to q(x)$ and the idea behind the send program is that it expects to receive a solution sequence $\langle t_0, \langle t_1, \ldots, \langle t_{l-1}, \langle t_l, -\rangle\rangle \cdots\rangle\rangle$ and does the following, where $-$ marks the end of the sequence and $M_i = \langle t_i, \ldots, \langle t_{l-1}, \langle t_l, -\rangle\rangle \cdots\rangle$ (see Appendix A for the formal definition of the send program):

1. If $t_0 = \langle \bot, \bot\rangle$, then $\{M_0\}_k$ is sent to the intruder.
2. For every $i \in \{0, \ldots, l-1\}$, if t_i is of the form $\langle x, y\rangle$ and t_{i+1} is of the form $\langle x \cdot u_j, y \cdot v_j\rangle$ for some $j \in \{1, \ldots, n\}$, then $\{M_i\}_k \oplus \{M_{i+1}\}_k$ is sent to the intruder.

3. If t_l is of the form $\langle x, x\rangle$, then $\{M_l\}_k \oplus \mathsf{Sec}$ is sent to the intruder.

Now, it is easy to see that if the intruder is able to send a solution sequence to the principal (which is the case if such a sequence exists), then by chaining all messages received from the principal via XOR, the intruder obtains Sec. Conversely, if the intruder cannot send a solution sequence (since no such sequence exists), he will not obtain Sec since at least one "link" in the chain will be missing. □

A similar reduction as the one presented above also works if XOR is replaced by symmetric encryption where keys may be arbitrary messages (complex keys). Hence, we also obtain the following theorem, where $\text{ATTACK}_{compkey}$ is the security problem in our model with complex keys (and without XOR).

Theorem 2. *The problem* $\text{ATTACK}_{compkey}$ *is undecidable.*

5 Decidability of ⊕-linear Protocols

In the proof of undecidability (Theorem 1) we used that a principal may conjoin two messages by XOR where both messages may depend on messages received from the network. In ⊕-linear protocols, defined next, this is forbidden. In this section, we show that the existence of attacks can be decided for ⊕-linear protocols.

A protocol P is *⊕-linear* if for each subterm of the form $t \oplus s$ occurring in P (both in protocol rules and in send programs), t or s is ground. For example, if the term $(x \oplus a) \oplus y$ with $a \in \mathcal{A}$, $x, y \in V$ occurs in P, then P is not ⊕-linear. The RA protocol (Section 3) is an example of an ⊕-linear protocol; see, e.g., [2] for another example. Let $\text{ATTACK}_{\oplus-linear} = \{P \mid P \text{ is an } \oplus\text{-linear, insecure protocol}\}$.

The main result of this paper is:

Theorem 3. *The problem* $\text{ATTACK}_{\oplus-linear}$ *is decidable.*

Before we provide a sketch of a proof of this theorem, we note that our result extends the decidability results presented in [6, 2] for non-recursive protocols to recursive protocols, in case the protocols are ⊕-linear and restricted to atomic keys.

In the remainder of this section we only sketch main ideas of the proof of Theorem 3, while the full proof is given in Appendix B.

The proof consists of two main steps: First, we prove certain properties of derivations in the Horn theory model with XOR (Section 5.1). Based on these properties we then reduce the security problem to the case without XOR, which by [14] is decidable. In the reduction we use the ability of principals to perform recursive computations in order to mimic operations involving XOR. The reduction is sketched in Section 5.2. An initial (minor) step, not further discussed in this section, is to turn a protocol into simple form. This is used to combine all derivations carried out in (3) and (4) into a single derivation from $I(0)$ to $I(\mathsf{Sec})$. So, we may consider an attack as a single derivation, called *attack derivation* (see Appendix B.1 for details).

$$I(s), I(s') \to I(c) \quad \text{for } s \oplus s' \sim c \tag{6}$$

$$I(c \oplus t), I(c) \to I(t) \tag{7}$$

$$I(c \oplus t), I(c') \to I(c \oplus c' \oplus t) \quad \text{for } c \not\sim 0 \text{ and } c \oplus c' \not\sim 0 \tag{8}$$

$$I(t), I(c) \to I(c \oplus t) \tag{9}$$

$$I(c_0), I(c_1 \oplus t_1), \ldots, I(c_n \oplus t_n) \to I(c \oplus t_1 \oplus \cdots \oplus t_n) \quad \text{for } n > 1, \text{ where the terms } t_1, \ldots, t_n \text{ are pairwise distinct and } c \sim c_0 \oplus \cdots \oplus c_n \tag{10}$$

Fig. 1. $\oplus$-Rules in Modest Derivations

5.1 Good Derivations

In this section, we identify and analyze properties of attack derivations.

First, we need to introduce some notation and terminology. We call a term *standard* if its top-symbol is not $\oplus$; otherwise, it is called *non-standard*. For a protocol P, let $\mathcal{S}_P$ denote the set of all the ground subterms of terms occurring in P and let $\mathcal{C}_P$ be the set consisting of all terms of the form $t_1 \oplus \cdots \oplus t_n$ with $t_i \in \mathcal{S}_P$ modulo $\sim$. Elements of $\mathcal{C}_P$ are referred to by c and decorations thereof. In what follows, non-standard terms will be written as $c \oplus t_1 \oplus \cdots \oplus t_n$ where c stands for a (ground) term in $\mathcal{C}_P$ and $t_1, \ldots, t_n$ are standard terms not in $\mathcal{C}_P$. A term is $\mathcal{C}_P$-*long* (or just *long*, if C_P is known from the context) if it is of the form $c \oplus t_1 \oplus \cdots \oplus t_n$, for $n > 1$. Otherwise it is called $\mathcal{C}_P$-*short* (or just *short*).

We now introduce what we call modest and normal derivations and prove properties about them.

A derivation is *modest* if it uses the rules depicted in Figure 1 instead of $I(x), I(y) \to I(x \oplus y)$ where $c, c_0, \ldots, c_n \in \mathcal{C}_P$ and $t, t', t_1, \ldots, t_n$ are standard ground terms not in $\mathcal{C}_P$; s and s' are arbitrary ground terms. We observe that in a modest derivation long terms may only be used to obtain an element of $\mathcal{C}_P$, by applying rule (6). In all other rules ((7)–(10)), only short terms may be conjoined by XOR. However, (10), which allows to combine an unbounded number of short terms, may produce a long term.

The next lemma states that it is enough to consider modest derivations. Therefore, in the remainder of this section we will assume derivations to be modest.

Lemma 1. *If there exists an attack on P, then this attack can be proven by a modest derivation.*

We now introduce normal derivations. In a normal derivation, applications of certain classes of rules are grouped into segments and segments occur in a certain order. In this extended abstract, we will only define some aspects of normal derivations (see the appendix for a full definition).

If $b_1, \ldots, b_l$ is a derivation, then $b_i, b_{i+1}, \ldots, b_j$ for $i \leq j$ is a subsequence of the derivation. A *segment* of a derivation is a maximal subsequence which does not contain any atom of the form $I(c)$, for some $c \in \mathcal{C}_P$, or any atom obtained by a protocol rule.

Rule (8) is called *variant rule*. A *variant segment* of a derivation is a maximal subsequence of a segment containing only atoms obtained by variant rules. Among others, a *normal* derivation satisfies the following conditions: (i) it does not contain two atoms a, a' such that $a \sim a'$, (ii) each segment contains at most one variant segment, and (iii) variant rules do not use as a premise an atom obtained from a variant rule. We can show the following:

Lemma 2. *If there exists an attack on P, then there exists a modest and normal derivation for this attack.*

Because the cardinality of the set $\mathcal{C}_P$ is exponentially bounded w.r.t. the size of P, the number of segments in an attack derivation is also exponentially bounded in the size of P. Now, as a result of Lemma 2 and the definition of normal derivations we obtain:

Lemma 3. *In a modest and normal attack derivation the number of variant segments is exponentially bounded in the size of the protocol.*

We now show that the number of long terms can be bounded in attack derivations. The key is the notion of a *profile* of a standard term. A profile α is defined w.r.t. an attack derivation δ and consists of an element c in $\mathcal{C}_P$ and a natural number k. Roughly speaking, two standard ground terms satisfy the same profile if they behave similarly w.r.t. c in the k-th segment of δ. In particular, if two terms have the same profile, each of them can be used instead of the other one when long terms are constructed by Rule (10). So, for a given derivation δ and a profile α, we will fix a term t^{δ}_{α} and use it whenever a term of profile α is used to build a long term.

If $c \oplus t_1 \oplus \cdots \oplus t_k$, for $k > 1$, is a long term, the positions where $t_1, \ldots, t_n$ occur are called *unimportant*. We can show the following lemma:

Lemma 4. *If there exists an attack on a protocol, then there exists a normal attack derivation δ for this protocol such that whenever terms t, t' of the same profile α occur in δ at unimportant positions, then $t = t'$.*

We call derivations of the form described in Lemma 4 *good*. Now, from the definition of profiles it immediately follows that the number of different profiles is (exponentially) bounded in the size of the protocol. Together with Lemma 4 we obtain:

Corollary 1. If a term $c \oplus t_1 \oplus \cdots \oplus t_n$ occurs in a good attack derivation, then n is bounded exponentially in the size of the protocol. Furthermore, in a good attack derivation for a protocol, the number of distinct terms of the form $c \oplus t_1 \oplus \cdots \oplus t_n$, for $n > 1$, is bounded by some (computable) number M in the size of the protocol.

5.2 Reduction to the $\oplus$-Free Case

We now show how the security problem can be reduced to the $\oplus$-free case, i.e., given a protocol P we construct a protocol P^+ which does not contain $\oplus$ such

that there exists an attack on P (in the sense defined in Section 2) iff there exists an attack on P^+ in the $\oplus$-free setting. The main steps are i) to represent terms with $\oplus$ by $\oplus$-free terms and ii) to mimic intruder rules involving $\oplus$ in the $\oplus$-free setting.

For i)—representing terms—we use additional constants: a new constant e and, for each equivalence class $[c]_\sim$, $c \in \mathcal{C}_P$, a new constant denoted by $[c]$. Now, for a term t, we obtain its $\oplus$–free representation, denoted by $\ulcorner t \urcorner$, by recursively applying to each non-standard subterm of t the following transformation: a subterm of the form $c \oplus t$ (recall that, according to our convention, $c \in \mathcal{C}_P$ and t is a standard term not in $\mathcal{C}_P$) is transformed into $\{t\}_{[c]}$, and a subterm of the form $c \oplus t_1 \oplus \cdots \oplus t_n$, for $n > 1$, is transformed into $\{\{t_1, \{\ldots, \{\langle t_{n-1}, t_n\rangle\}_e \ldots\}_e\}_e\}_{[c]}$. We also substitute every $c \in \mathcal{C}_P$ by the constant $[c]$.

Now, we turn to intruder rules involving $\oplus$ and show how they can be mimiced in the $\oplus$-free setting. By the results of Section 5.1, we may assume that attack derivations are modest, normal, and good. In particular, by Lemma 1, it suffices to mimic rules (6) to (10):

- *Rule* (7) *and* (9): These rules can easily be mimiced by ordinary intruder rules (decryption and encryption). Consider, for instance, rule (7): In the original attack atoms $I(c \oplus t)$ and $I(c)$ are used to obtain $I(t)$. Now, $I(\ulcorner t \urcorner)$ can be derived from $I(\ulcorner c \oplus t \urcorner) = I(\{\ulcorner t \urcorner\}_{[c]})$ and $I(\ulcorner c \urcorner) = I([c])$ by the standard decryption rule.
- *Rule* (6): The result of this rule is of the form $I(c)$ with $c \in \mathcal{C}_P$. Because there is a bounded number, say L, of elements in $\mathcal{C}_P$, we can mimic this rule by adding L principals to P each with a single protocol step of the form $\langle \{x\}_{[c]}, \{x\}_{[c']}\rangle \to I([c \oplus c'])$.
- *Rule* (10): The result of this rule is a long term and we know, by Corollary 1, that the number of such terms is bounded by a constant M which only depends on the size of the protocol, so, again, we can handle this case by adding a bounded number of principals each with a single protocol step of the form

$$\langle [c_0], \{y_1\}_{[c_1]}, \ldots, \{y_n\}_{[c_n]}\rangle \to I(\{\{y_1, \{\ldots, \{y_{n-1}, y_n\}_e \ldots\}_e\}_e\}_{[c]}).$$

- *Rule* (8): By Lemma 3, we know that the number of variant segments (i.e., blocks of atoms obtained by the variant rule) is bounded by a number N depending only on the protocols size and that no element obtained by a variant rule is necessary as a premise of a variant rule in the same variant segment. Hence, each of these variant segments can be handled by a protocol step of the following form:

$z \to p(z)$ with the following send program:

$$p(\langle x, y\rangle) \Rightarrow p(y)$$
$$p(\langle x, y\rangle) \Rightarrow p'(x)$$
$$p'(\langle \{x\}_{[c]}, [c']\rangle) \Rightarrow I(\{x\}_{[c \oplus c']}) \qquad \text{for } c, c' \in \mathcal{C}_P$$

More details on the construction of P^+ can be found in the appendix. We can show:

Lemma 5. *For an $\oplus$-linear protocol P we have that there exists an attack on P (in the sense of Section 2) if and only if there exists an attack on P^+ in the $\oplus$-free setting.*

Since the security of P^+ is decidable [14] and P^+ can effectively be computed from P, Theorem 3 follows. A more careful analysis of the complexity of our construction reveals that the size of P^+ is double exponential in the size of P. As the secrecy of P^+ can be decide in NexpTime [14], we obtain an 3-NexpTime upper bound. Nonetheless, we believe that this upper bound can be reduced to NexpTime by a more careful construction and a refinement of the proof in [14].

6 Conclusion

In this work, we have proved that security (w.r.t. a bounded number of sessions) is decidable for the class of $\oplus$-linear protocols. This is the first decidability result for recursive protocols involving algebraic properties of operators. We have also shown that relaxing certain assumptions of our model lead to undecidability of security. Our decidability result was obtained in a modular way by first reducing the problem of deciding security in the Horn theory model with XOR to the one without XOR and then using the existing decidability result for the latter model. We expect that the modular proof technique developed in this paper also helps to deal with other operators, such as Diffie-Hellman exponentiation.

A The Send Program in the Proof of Theorem 1

The send program in the proof of Theorem 1 is defined as follows, where j ranges over $\{1, \ldots, n\}$:

$$q(\langle\langle\bot,\bot\rangle,x\rangle) \Rightarrow q'(x) \tag{11}$$
$$q(\langle\langle\bot,\bot\rangle,x\rangle) \Rightarrow I(\{\langle\langle\bot,\bot\rangle,x\rangle\}_k) \tag{12}$$
$$q(M_{0,j}) \Rightarrow I(\{M_{0,j}\}_k \oplus \{M_{1,j}\}_k) \tag{13}$$
$$q'(\langle x,y\rangle) \Rightarrow q'(y) \tag{14}$$
$$q'(M_{0,j}) \Rightarrow I(\{M_{0,j}\}_k \oplus \{M_{1,j}\}_k) \tag{15}$$
$$q'(\langle\langle x,x\rangle,-\rangle) \Rightarrow I(\{\langle\langle x,x\rangle,-\rangle\}_k \oplus \mathsf{Sec}) \tag{16}$$

with

$$M_{0,j} = \langle\langle x_1,x_2\rangle,\langle\langle x_1\cdot u_j, x_2\cdot v_j\rangle,y\rangle\rangle$$
$$M_{1,j} = \langle\langle x_1\cdot u_j, x_2\cdot v_j\rangle,y\rangle$$

It is straightforward to check that Sec can only be derived by the intruder if a solution sequence is given as input to the send program.

B Proof of Theorem 3

While the key ideas of the proof of Theorem 3 were given in Section 5, now we provide the details of the proof. First, in Section B.1, we show how to turn a protocol into simple form. This is used to combine all derivations carried out in (3) and (4) into a single derivation from $I(0)$ to $I(\mathsf{Sec})$. So, we may consider an attack as a single derivation, called *attack derivation*. Second, in Sections B.2 and B.3, we prove certain properties of derivations in the Horn theory model with XOR. Based on these properties we then reduce the security problem to the case without XOR, which by [14] is decidable. In the reduction we use the ability of principals to perform recursive computations in order to mimic operations involving XOR. The reduction is presented in Section B.4.

B.1 Simple Protocols

The goal of this subsection is to give a more compact and convenient characterization of an attack.

In the following, we write a protocol step with the protocol rule of the form $t \to q_1(s_1), \ldots, q_n(s_n)$ and a send program Φ as a shortcut for the protocol step which consists of the protocol rule $t \to q_0(\langle s_1, \ldots, s_n \rangle)$, where q_0 is a distinct predicate symbol, and the send program Φ with the additional rules $q_0(\langle x_1, \ldots, x_n \rangle) \to q_i(x_i)$, for $i = 1, \ldots, n$.

Definition 1. A protocol P is *simple*, if both of the following condition are satisfied:

1. Each principal has exactly one protocol step (thus a protocol can be seen as a set of independent protocol steps).
2. Each protocol rule of R is of the form $x \to r(x)$, for some $r \in R$.

Lemma 6. *For each protocol P there exists a simple protocol P' such that P is secure iff P' is. Moreover, P' can be computed in polynomial time w.r.t. the size of P.*

Proof. First we show how to transform a principal Π into a set of independent protocol steps (more formally, into a set of principals each of who consists of one protocol step). We represent nodes of Π by sequences of natural numbers (elements of $\{0, \ldots, k-1\}^*$, where k is the maximal branching degree of Π), in the usual way. For a node u of Π, let $r_u = \{x_1, \ldots, x_n\}_{l_u}$, where l_u is a fresh key and $x_1, \ldots, x_n$ are the variables occurring in the protocol rules obtained when traversing from the root of Π to u. We transform Π into the following rules.

$x \to I(\{x\}_{k_u})$	for each inner node u of Π
$\{d\}_{k_v},\ r_v,\ t \to q(s),\ I(r_u)$	for each node $u = vd$ of Π, where the label of the edge (v, u) consists of a rule $t \to q(s)$ with a send program Φ

where, for each node u, we have a fresh key k_u. In the case of the latter rule, we equip it with the original send program Φ. One can show that the protocol P'

obtained by replacing each participant Π by the set of protocol steps obtained in this way is secure if and only if P is secure.

Now, to obtain a version of the protocol which meets Item 2 of Definition 1, we replace each protocol rule $t \to I(s)$ by the rule $x \to q(x)$, for some fresh predicate symbol q, and we add the rule $q(t) \Rightarrow I(s)$ to the send program. Similarly, we replace each protocol rule $t \to r(s)$, for $r \in Q$, by rules $x \to q(x)$ and $y \to p(y)$, where q, p are fresh predicate symbols, and we extend the send program by the rules $q(t) \Rightarrow I(\{s\}_{k_r})$ and $p(\{z\}_{k_r}) \Rightarrow r(z)$. One can show that the protocol obtained in this way is secure if and only if P is secure.

One can also observe that the transformations described above are polynomial.

Lemma 6 allows us to consider only simple protocols without loss of generality. Furthermore, if simple protocols are considered, we can reformulate the notion of an attack in a simpler and more uniform way, as we do it below.

Let P be a simple protocol. We can assume that distinct protocol rules do not share variables. For a ground substitution θ, we define the theory $P(\theta)$:

$$P(\theta) \;=\; T_{\oplus} \cup \Phi \cup \{I(x\theta) \Rightarrow q(x\theta) \mid (x \to q(x)) \text{ is a protocol rule of } P\}.$$

where Φ is the union of all send programs occurring in P. Now, one can see that P is insecure if and only if there exists a substitution θ such that

$$I(0) \vdash_{P(\theta)} I(\mathsf{Sec}). \tag{17}$$

In the remainder of this paper we will always assume that protocols are simple and we will use (17) as a characterization of an attack.

B.2 Modest and Normal Derivations

Recall that we call a term *standard* if its top-symbol is not $\oplus$; otherwise, it is called *non-standard*. For a simple protocol P, let $\mathcal{S}_P$ denote the set containing the term 0 and all the ground subterms of terms occurring in P, and let $\mathcal{C}_P$ be the set $\{t_1 \oplus \cdots \oplus t_n \mid t_1, \ldots, t_n \in \mathcal{S}_P\}$. Like in the previous sections, elements of $\mathcal{C}_P$ are referred to by c and decorations thereof. Non-standard terms will be written as $c \oplus t_1 \oplus \cdots \oplus t_n$ where c stands for a (ground) term in $\mathcal{C}_P$ and $t_1, \ldots, t_n$ are standard terms not in $\mathcal{C}_P$. Without loss of generality we will assume that every nonstandard term $c \oplus t_1 \oplus \cdots \oplus t_n$ occurring in a derivation is in a normal form, i.e. the terms $t_1, \ldots, t_n$ are distinct.

For a term t, we define $F(t)$ as follows: $F(c) = F(t) = F(c \oplus t) = \emptyset$, if $c \in \mathcal{C}_P$ and t is a standard term not in $\mathcal{C}_P$, and $F(c \oplus t_1 \oplus \cdots \oplus t_n) = \{t_1, \ldots, t_n\}$, for $n > 1$, where $c \in \mathcal{C}_P$ and $t_1, \ldots, t_n$ are standard terms not in $\mathcal{C}_P$. Now, for a term t, let $G(t)$ be defined by the equation $G(t) = \bigcup_{t' \leq t} F(t')$ (where $\leq$ is the subterm ordering). A term is $\mathcal{C}_P$-*long* (or just *long*, if C_P is known from the context) if it is of the form $c \oplus t_1 \oplus \cdots \oplus t_n$, for $n > 1$. Otherwise it is called $\mathcal{C}_P$-*short* (or just *short*). One can see that a term t is long if and only if the set $F(t)$ is not empty.

If $\delta = a_1, \ldots, a_n$ is a derivation, then $\delta(i)$ stands for a_i. Moreover, $\delta_{<i}$ stands for $a_1, \ldots, a_{i-1}$. For a derivation δ of length n, let $G(\delta) = \bigcup_{i=1}^{n} G(\delta(i))$.

Lemma 7. *Assume that δ is a derivation for* (17)*. If $\delta(k)$ contains a term t as a subterm, then, for each $t' \in F(t)$, there exists $c \in \mathcal{C}_P$ such that $I(c \oplus t')$ occurs in $\delta_{<k}$.*

Proof. If $F(t) = \emptyset$, it is nothing to prove. So, consider the case $F(t) \neq \emptyset$. We proceed by induction on k. Suppose that the lemma holds true for each $i < k$. We will show that it holds for k. If we assume that, for some $i < k$, a term $t' \sim c \oplus t$ occurs in $\delta(i)$, then we are trivially done (note that in this case $F(t) = F(t')$). Otherwise, $\delta(k)$ has to be $I(t)$ and is obtained by the XOR rule, i.e. $t \sim s \oplus r$, for some $\delta(i) = I(r)$ and $\delta(j) = I(s)$, for $i, j < k$ (it is because the protocol is $\oplus$-linear and thus only the XOR rule can introduce a term t with $F(t) \neq \emptyset$). Let $t' \in F(t)$. One can check that one of the following cases holds: (a) s is of the form $c \oplus t'$, (b) r is of the form $c \oplus t'$, (c) $t' \in F(s)$, or (d) $t' \in F(r)$. If (a) or (b) holds, we clearly have $c \oplus t' \in \delta_{<k}$. If (c) or (d) holds, we obtain $c \oplus t' \in \delta_{<k}$ by the inductive hypothesis. □

A derivation is *modest* if it uses the rules depicted in Figure 1 (on Page 9) instead of $I(x), I(y) \rightarrow I(x \oplus y)$ where $c, c_0, \ldots, c_n \in \mathcal{C}_P$ and $t, t', t_1, \ldots, t_n$ are standard ground terms not in $\mathcal{C}_P$; s and s' are arbitrary ground terms. We observe that in a modest derivation long terms may only be used to obtain an element of $\mathcal{C}_P$, by applying rule (6). In all other rules ((7)–(10)), only short terms may be conjoined by XOR. However, (10), which allows to combine an unbounded number of short terms, may produce a long term.

Lemma 8. *If* (17) *holds, for some θ, then it has a modest derivation.*

Proof. It is enough to show how to obtain $z_0 = I(e \oplus f \oplus t_1 \oplus \cdots \oplus t_k \oplus t_{n+1} \oplus \cdots \oplus t_{n+l})$ from $z_1 = I(e \oplus t_1 \oplus \cdots \oplus t_n)$ and $z_2 = I(f \oplus t_{k+1} \oplus \cdots \oplus t_{n+l})$ in a modest derivation. By Lemma 7, we have previously derived $I(c_1 \oplus t_1), \ldots, I(c_{n+l} \oplus t_{n+l})$. Thus, one can obtain $I(c_1 \oplus \cdots \oplus c_n \oplus t_1 \oplus \cdots \oplus t_n)$ and $I(c_{k+1} \oplus \cdots \oplus c_{n+l} \oplus t_{k+1} \oplus \cdots \oplus t_{n+l})$, so, using z_1 and z_2 one can get $I(e \oplus c_1 \oplus \cdots \oplus c_n)$ and $I(f \oplus c_{k+1} \oplus \cdots \oplus c_{n+l})$, and so $z_3 = I(e \oplus f \oplus c_1 \oplus \cdots \oplus c_k \oplus c_{n+1} \oplus \cdots \oplus c_{n+l})$. Using z_3, $I(c_1 \oplus t_1), \ldots, I(c_k \oplus t_k)$, and $I(c_{n+1} \oplus t_{n+1}), \ldots, I(c_{n+l} \oplus t_{n+l})$, we obtain z_0. □

A derivation δ is *minimal*, if $\delta(i) \not\sim \delta(j)$, for $i \neq j$. A derivation δ is said to be *constant–closed*, if for each i such that $\delta(i)$ is not of the form $I(c)$, for $c \in \mathcal{C}_P$, if $I(c), I(c') \in \delta_{<i}$, for $c, c' \in \mathcal{C}_P$, then $I(c \oplus c') \in \delta_{<i}$ (modulo $\sim$). For any derivation δ it is easy to obtain a minimal derivation δ' just by removing repetitions. It is also easy to obtain a derivation which is constant–closed.

Now, we group the rules of $P(\theta)$ into classes. If $x \rightarrow q(x)$ is a protocol rule of P, then a rule of $P(\theta)$ of the form $I(x\theta) \Rightarrow q(x\theta)$ is called a *protocol rule*. We call Rules (9), (10) and the following standard intruder rules *composition rules*:

$$I(x), I(y) \Rightarrow I(\langle x, y \rangle), \qquad I(x), I(k) \Rightarrow I(\{x\}_k),$$
$$I(x), I(k) \Rightarrow I(\{\!|x|\!\}_k), \qquad I(x) \Rightarrow \mathsf{hash}(x).$$

We call Rule (7) and the following standard intruder rules *decomposition rules*:

$$I(\langle x, y\rangle) \Rightarrow I(x), \qquad I(\{x\}_k), I(k) \Rightarrow I(x),$$
$$I(\langle x, y\rangle) \Rightarrow I(y), \qquad I(\{\!|x|\!\}_k), I(k^{-1}) \Rightarrow I(x).$$

Rule (6) is called *generating* rule, and finally, Rule (8) is called *variant* rule.

If $b_1, \ldots, b_l$ is a derivation, then $b_i, b_{i+1}, \ldots, b_j$ for $i \leq j$ is a *subsequence* of the derivation. A *segment* of a derivation is a maximal subsequence which does not contain any element obtained by a protocol rule nor an element of the form $I(c)$, for $c \in \mathcal{C}_P$. A derivation is *normal*, if

(i) it is modest, minimal, and constant–closed,
(ii) each segment consists of (a) push rules followed by (b) send rules followed by (c) decomposition rules followed by (d) variant rules followed by (e) composition rule.

Note that the number of segments in a minimal derivation is at most exponential in the size of the protocol.

Lemma 9. *If δ is a constant–closed derivation, then variant rules do not need to use as a premise an atom obtained from a variant rule or from a composition rule.*

Proof. For the sake of contradiction, suppose that some application of variant rule needs an element obtained by a composition or variant rule as a premise. Let us take the first such application. So, the obtained element is $I(c_0 \oplus t)$ and the premises are $I(c_1 \oplus t)$ and $I(c_1')$, for some $c_1 \not\sim 0$, with $c_0 \sim c_1 \oplus c_1'$. Now, we have two cases:

Case 1: $I(c_1 \oplus t)$ was obtained by a composition rule, which means that $I(t)$ and $I(c_1)$ were used to obtain it. But, since the derivation is constant–closed, we could use $I(c_1 \oplus c_1')$ and $I(t)$ to obtain $I(c_0 \oplus t)$ by a composition rule.

Case 2: $I(c_2 \oplus t)$ and $I(c_2')$ were used to obtain $I(c_1 \oplus t)$, for $c_1 \sim c_2 \oplus c_2'$. By our assumption, $I(c_2 \oplus t)$ is not obtained by the variant rule. But, because the derivation is constant-closed, we could use $I(c_1' \oplus c_2')$ and $I(c_2 \oplus t)$ to obtain $I(c_0 \oplus t)$ directly, which contradicts the assumption (note that $c_2 \oplus c_1' \oplus c_2' \sim c_0$). □

Lemma 10. *If δ_0 is a modest derivation for* (17)*, then there is a normal derivation δ for* (17)*. Moreover $G(\delta_0) = G(\delta)$.*

Proof. Suppose that δ_0 is a modest derivation for (17). As we mentioned earlier, it is easy to obtain a constant–closed and minimal version δ_1 of it. Note that $G(\delta_0) = G(\delta_1)$. We will obtain a normal derivation δ by reordering the elements of each segment of δ_1. One can easily see that this operation preserves the set $G(\delta)$.

First, we can observe that premises of (a) are not of the form $I(t)$, so they cannot be a result of (b), (c), (d), nor (e). So we can place rules (a) in front of the remaining ones. For the similar reason, we can place rules (b) in front of (c), (d) and (e).

In order to show that results of (d) and (e) need not be used as premises of (c), we consider two cases. *Case 1*: a term t is obtained from a term t' by a standard decomposition rule. If t' is obtained by (e) then t has to be known earlier. The term t' cannot be obtained by (d), since (d) produces nonstandard terms. *Case 2*: a term t is obtained from c and $c \oplus t$, for $c \not\sim 0$, and there is no other way to obtain it. If $c \oplus t$ were obtained by a composition rule from c and t, then t would be known earlier, so we can assume that $c \oplus t$ is obtained by variant rule from some $c' \oplus t$ and c'' such that $c \sim c' \oplus c''$. But in this case, by Lemma 9, $c' \oplus t$ is not obtained neither by variant nor the composition rule. So, we can use this term and $c'' \oplus c \sim c'$ (which can be used because the derivation is constant–closed) to obtain t.

To complete the proof, it is enough to note that, by Lemma 9, no conclusion of (e) is necessary as a premise of (d). □

B.3 Profiles of Terms

Let $t \notin \mathcal{C}_P$ be a standard term. Let δ be a derivation for (17). A *composition-segment* in δ is a maximal subsequence of elements obtained by composition rules. A *profile* (w.r.t δ) of a term t is a pair (c, k), where $c \in \mathcal{C}_P$ (note that $0 \in \mathcal{C}_P$) and i is a natural number such that:

(i) $\delta(i) \sim I(c \oplus t)$, for some i, and there is no $j < i$ such that $\delta(j) \sim I(c' \oplus t)$,
(ii) k is the number of composition–segments in $\delta_{<i}$.

We define a set $\alpha(\delta)$ as follows: $t \in \alpha(\delta)$ iff the profile of t w.r.t. δ is α. Note that, if δ is a normal derivation, then the number of (nonempty) profiles is bounded exponentially in the size of the protocol.

Consider a derivation δ and a nonempty profile $\alpha(\delta)$. We fix a term t^δ_α in the following way. If $\alpha = (0, k)$, then $t^\delta_\alpha = 0$. Otherwise let t^δ_α be a term which is minimal (w.r.t. the subterm ordering) in the set $\alpha(\delta)$. Note that elements of $\alpha(\delta)$ are standard terms. So, either $t^\delta_\alpha = 0$ (note that in this case t^δ_α is not necessarily an element of $\alpha(\delta)$) or it is a standard term from the set $\alpha(\delta)$.

Lemma 11. *Assume that δ is a normal derivation for* (17)*. Let $t \in \alpha(\delta)$. If $\delta(i)$ is obtained by a composition rule and $I(c \oplus t) \in \delta_{<i}$, then $I(c \oplus t^\delta_\alpha)$ can be derived from $\delta_{<i}$ in at most two steps.*

Proof. We consider two cases: *Case 1*: $\alpha = (0, k)$ and thus $t^\delta_\alpha = 0$. We have $I(c \oplus t) \in \delta_{<i}$. Since $(0, k)$ is the profile of t, the element $I(t)$ cannot occur after $I(c \oplus t)$, so it occurs in $\delta_{<i}$ too. Hence, from $\delta_{<i}$, we can derive c which is equivalent to $c \oplus t^\delta_\alpha$.

Case 2: $\alpha = (d, k)$, for $d \neq 0$. The first occurrence of $I(d \oplus t)$ and $I(d \oplus t^\delta_\alpha)$ cannot be obtained by a composition rule (if, for instance, $I(d \oplus t)$ were obtained from $I(d)$ and $I(t)$, then (d, k) could not by the profile of t). Furthermore, there is no element obtained by any composition rule between $I(d \oplus t)$ and $I(d \oplus t^\delta_\alpha)$. $I(d \oplus t)$ cannot occur after $I(c \oplus t)$, so $I(d \oplus t) \in \delta_{<i}$ and so $I(d \oplus t^\delta_\alpha) \in \delta_{<i}$. Recall that we have also $I(c \oplus t) \in \delta_{<i}$. Hence, from $\delta_{<i}$, we can derive $I(c \oplus d)$ and then $I(c \oplus t^\delta_\alpha)$. □

Now, for a derivation δ and a profile α, we define a function ∇^δ_α from terms to terms by the following equations.

$$\begin{aligned}
\nabla^\delta_\alpha(x) &= x && \text{for a variable } x\\
\nabla^\delta_\alpha(f(t_1,\ldots,t_n)) &= f(\nabla^\delta_\alpha(t_1),\ldots,\nabla^\delta_\alpha(t_n)) && \text{for } f\in\Sigma,\ f\neq\oplus\\
\nabla^\delta_\alpha(c) &= c\\
\nabla^\delta_\alpha(c\oplus t) &= c\oplus\nabla^\delta_\alpha(t)\\
\nabla^\delta_\alpha(c\oplus t_1\oplus\cdots\oplus t_n) &= c\oplus t'_1\oplus\cdots\oplus t'_n && \text{for } n>1, \text{ where } t'_i = t^\delta_\alpha, \text{ if } t\in\\
& && \alpha(\delta), \text{ and } t'_i=\nabla^\delta_\alpha(t_i), \text{ otherwise.}
\end{aligned}$$

We can note that, because t^δ_α is equal to 0 or it is a minimal element of $\alpha(\delta)$, we have $\nabla^\delta_\alpha(t^\delta_\alpha) = t^\delta_\alpha$. For a substitution θ, let $\nabla^\delta_\alpha(\theta)$ be the substitution defined by the equality $\nabla^\delta_\alpha(\theta)(x) = \nabla^\delta_\alpha(\theta(x))$, for each variable x. For a derivation δ of the length n, let $\nabla^\delta_\alpha(\delta) = \nabla^\delta_\alpha(\delta(1)),\ldots,\nabla^\delta_\alpha(\delta(n))$.

It is easy to prove the following two lemmas.

Lemma 12. *If $s_1\sim s_2$ then $\nabla^\delta_\alpha(s_1)\sim\nabla^\delta_\alpha(s_2)$.*

Lemma 13. *Let t be a term occurring in the protocol P (recall that P is $\oplus$-linear). Then, for each substitution θ, we have $\nabla^\delta_\alpha(t\theta)\sim t(\nabla^\delta_\alpha(\theta))$.*

For a derivation δ and a profile α, we define the function h^δ_α from terms to terms as follows: $h^\delta_\alpha(t) = t^\delta_\alpha$, if $t\in\alpha(\delta)$, and $h^\delta_\alpha(t) = \nabla^\delta_\alpha(t)$, otherwise.

Lemma 14. *Let δ be a normal derivation for $I(0)\vdash_{P(\theta)} I(\mathsf{Sec})$, and α be a profile. Then there exists a normal derivation δ^* for $I(0)\vdash_{P(\nabla^\delta_\alpha(\theta))} I(\mathsf{Sec})$ such that $G(\delta^*)\subseteq\{h^\delta_\alpha(t)\mid t\in G(\delta))$.*

Proof. Let δ_0 be a normal derivation for $I(0)\vdash_{P(\theta)} I(\mathsf{Sec})$. We will iteratively use Lemma 11 to obtain a derivation δ such that if $\delta(i)$ is obtained by a composition rule and $I(c\oplus t)\in\delta_{<i}$, for some $t\in\alpha(\delta)$, then $I(c\oplus t_\alpha)\in\delta_{<i}$: For each composition-segment, let us take the first element of it, denote it by $\delta(i)$. For each profile α such that $I(c\oplus t)\in\delta_{<i}$, for some $t\in\alpha(\delta)$, and $I(c\oplus t_\alpha)\notin\delta_{<i}$ we insert the two elements given by Lemma 11 right before this composition-segment. Note that this operation does not splits any composition-segment, so the profiles of terms remain the same (i.e. for each term t, we have $t\in\alpha(\delta_0)$ iff $t\in\alpha(\delta)$). Hence $\nabla^{\delta_0}_\alpha(t) = \nabla^\delta_\alpha(t)$. Note that $G(\delta_0) = G(\delta)$.

Now, let $\delta' = \nabla^\delta_\alpha(\delta)$. We will show, by induction, that δ' is a derivation for $I(0)\vdash_{P(\nabla^\delta_\alpha(\theta))} I(\mathsf{Sec})$. So, assume that $\nabla^\delta_\alpha(\delta_{<i})$ is a proper derivation. We should show that $\nabla^\delta_\alpha(\delta(i))$ can be obtained from $\nabla^\delta_\alpha(\delta_{<i})$ in one step. We consider a few cases:

1. If $\delta(i)$ is obtained by some standard intruder rule, a protocol rule, or one of the Rules (7), (8), (9), then the proof is straightforward.
2. $\delta(i)$ is obtained by send rule. We have $\delta(i)\sim I(t\theta)$ and $\delta(j)\sim q(s\theta)$, for some $j<i$ and some send rule $q(s)\to I(t)$. One can show that $\nabla^\delta_\alpha(t\theta)\sim t(\nabla^\delta_\alpha\theta)$ and $\nabla^\delta_\alpha(s\theta)\sim s(\nabla^\delta_\alpha(\theta))$. Hence, by Lemma 12, we have $\nabla^\delta_\alpha(\delta(i))\sim I(\nabla^\delta_\alpha(t\theta))\sim I(t(\nabla^\delta_\alpha(\theta)))$ and $\nabla^\delta_\alpha(\delta(j))\sim q(\nabla^\delta_\alpha(s\theta))\sim q(s(\nabla^\delta_\alpha(\theta)))$. Since $\nabla^\delta_\alpha(\delta(j))\in\nabla^\delta_\alpha(\delta_{<i})$, we can obtain $\nabla^\delta_\alpha(\delta(i))$ from $\nabla^\delta_\alpha(\delta_{<i})$ in one step.

3. If $\delta(i)$ is obtained by push rule, then the proof is similar.
4. $\delta(i) = I(c \oplus c')$ is obtained by Rule (6) from $I(c \oplus t_1 \oplus \cdots \oplus t_n)$ and $I(c' \oplus t_1 \oplus \cdots \oplus t_n)$. In this case, we have $I(c \oplus \nabla^\delta_\alpha(t_1 \oplus \cdots \oplus t_n)) \in \nabla^\delta_\alpha(\delta_{<i})$ and $I(c' \oplus \nabla^\delta_\alpha(t_1 \oplus \cdots \oplus t_n)) \in \nabla^\delta_\alpha(\delta_{<i})$, so we can derive $\nabla^\delta_\alpha(\delta(i)) = c \oplus c'$ from $\nabla^\delta_\alpha(\delta_{<i})$ in one step.
5. $\delta(i) = I(c_0 \oplus \cdots \oplus c_n \oplus t_1 \oplus \cdots \oplus t_n)$ is obtained by Rule (10) from $I(c_0)$, $I(c_1 \oplus t_1)$, $\ldots$,$I(c_n \oplus t_n)$. Without lost of generality we can assume that $t_i \in \alpha(\delta)$ iff $i \in \{1, \ldots, k\}$, for some k. We have $I(c_0) \in \nabla^\delta_\alpha(\delta_{<i})$ and $I(c_i \oplus \nabla^\delta_\alpha(t_i)) \in \nabla^\delta_\alpha(\delta_{<i})$, for $i \in \{k+1, \ldots, n\}$.
   Because $\delta(i)$ is obtained by a composition rule and, for $i \in \{1, \ldots, k\}$, we have $t_i \in \alpha(\delta)$ and $I(c_i \oplus t_i) \in \delta_{<i}$, then $I(c_i \oplus t_\alpha) \in \delta_{<i}$. Since $\nabla^\delta_\alpha(t_\alpha) = t_\alpha$, we have also $I(c_i \oplus t_\alpha) \in \nabla^\delta_\alpha(\delta_{<i})$, for $i \in \{1, \ldots, k\}$. Hence, we can derive $\nabla^\delta_\alpha(\delta(i))$ from $\nabla^\delta_\alpha(\delta_{<i})$ in one step.

One can show that $G(\delta') \subseteq \{h^\delta_\alpha(t) \mid t \in G(\delta)\}$ (we do not have equality here, because some terms might have been canceled applying ∇^δ_α). However, δ' may be not normal (it may not be minimal). So, we apply Lemma 10 to obtain a normal derivation δ^* such that $G(\delta^*) = G(\delta') \subseteq \{h^\delta_\alpha(t) \mid t \in G(\delta)\}$. □

A derivation δ is *good*, if it is modest, minimal, normal, and for each $t, t' \in G(\delta)$ of the same profile α, we have $t = t'$.

Lemma 15. *If, for some substitution θ, we have $I(0) \vdash_{P(\theta)} I(\mathsf{Sec})$, then there exists a substitution θ' and a good derivation δ for $I(0) \vdash_{P(\theta')} I(\mathsf{Sec})$.*

Proof. Let δ_0 be a normal derivation for $I(0) \vdash_{P(\theta_0)} I(\mathsf{Sec})$, for some θ_0. We will iteratively apply Lemma 14, obtaining $\delta_1, \delta_2, \ldots$ together with $\theta_1, \theta_2, \ldots$, such that δ_i is a normal derivation for $I(0) \vdash_{P(\theta_i)} I(\mathsf{Sec})$. We will do it as long as there are two distinct terms $t, t' \in G(\delta_i)$ of the same profile α; in this case we obtain δ_{i+1} applying Lemma 14 for α. Since $G(\delta_{i+1}) \subseteq \{h^{\delta_i}_\alpha(t) \mid t \in G(\delta_i)\}$ and $h^{\delta_i}_\alpha(t) = h^{\delta_i}_\alpha(t') = t^{\delta_i}_\alpha$, we have $|G(\delta_{i+1})| < |G(\delta_i)|$. We know that $G(\delta_0)$ is finite. Hence, after a finite number of steps we have to obtain a good derivation. □

Corollary 2. In a good derivation we have:

1. The number of distinct terms of the form $c \oplus t_1 \oplus \cdots \oplus t_n$, for $n > 1$, is bounded by some M dependent on the size of a protocol. Moreover, for each such a term, n is bounded by some constant dependent of the size of the protocol.
2. The number of segments, and thus of blocks of elements obtained by variant rules, is bounded by some N dependent on the size of a protocol.

B.4 The Reduction

For each equivalence class $[c]$ w.r.t. the XOR equality, for $c \in \mathcal{C}_P$, we add a constant denoted by $[c]$ to Σ. Let $\prec$ be a linear ordering relation on terms. As in the previous section, we assume that nonstandard terms are represented by expressions of the form $c \oplus t_1 \oplus \cdots \oplus t_n$ or $t_1 \oplus \cdots \oplus t_n$, if $c \sim 0$, where $c \in \mathcal{C}_P$ and $t_1, \ldots, t_n \notin \mathcal{C}_P$ are standard. Now, we assume also that $t_1 \prec \cdots \prec t_n$.

$$
\begin{aligned}
\ulcorner x \urcorner &= x && \text{for a variable } x\\
\ulcorner c \urcorner &= [c] && \text{for } c \in \mathcal{C}_P\\
\ulcorner \langle t, s\rangle \urcorner &= \langle \ulcorner t \urcorner, \ulcorner s \urcorner\rangle\\
\ulcorner \{t\}_k \urcorner &= \{\ulcorner t \urcorner\}_k\\
\ulcorner \{\!|t|\!\}_k \urcorner &= \{\!|\ulcorner t \urcorner|\!\}_k\\
\ulcorner \mathsf{hash}(t) \urcorner &= \mathsf{hash}(\ulcorner t \urcorner)\\
\ulcorner c \oplus t \urcorner &= \{\ulcorner t \urcorner\}_{[c]},\\
\ulcorner t_1 \oplus \cdots \oplus t_n \urcorner &= \{\ulcorner t_1 \urcorner, \{\ldots, \{\ulcorner t_{n-1} \urcorner, \ulcorner t_n \urcorner\}_e \ldots\}_e\}_e\\
\ulcorner c \oplus t_1 \oplus \cdots \oplus t_n \urcorner &= \{\{\ulcorner t_1 \urcorner, \{\ldots, \{\ulcorner t_{n-1} \urcorner, \ulcorner t_n \urcorner\}_e \ldots\}_e\}_e\}_{[c]}
\end{aligned}
$$

Fig. 2. Definition of $\ulcorner t \urcorner$

For a term t (possibly with variables), we define $\ulcorner t \urcorner$ in Figure 2, where e is a fresh key. The rough idea of the reduction is as follows. For a ground term t, the term $\ulcorner t \urcorner$, which does not contain $\oplus$, will serve as a canonical representation of the equivalence class of t (w.r.t. $\sim$). We will construct a protocol P^+ without $\oplus$ which is secure if and only if the original protocol P is. In particular, we will prove that whenever, in an attack θ on P, a term t can be derived, then it is possible to mimic this step in such a way that the term $\ulcorner t \urcorner$ can be derived in a corresponding attack on P^+.

We denote by Σ the signature of the original protocol P (containing, amongst the others, the $\oplus$ symbol and constants used in P). By Σ' we will denote the signature of P^+ which does not contain $\oplus$, but in addition, contains new constants (the constant e and the constants of the form $[c]$).

For a set of variables V, we define a set $\Theta(V)$ of substitutions as follows: $\sigma \in \Theta_V$, if $\mathrm{dom}(\sigma) = V$ and, for each $x \in V$, we have $\sigma(x) = x$ or $\sigma(x) = c \oplus x$, for some $c \in \mathcal{C}_P$. Let P be a simple protocol. The definition of the protocol P^+ is given on Figure 3. Rules (20) and (21) are intended to simulate Rule (6). Rule (22) is intended to simulate Rule (10) . Note that, by Corollary 2, it suffices to apply Rule (10) M times. Rules (23)–(25) are intended to simulate Rule (8) (note that, by Corollary 2, the number of blocks of element obtained by variant rule is bounded by N). Finally, Rule (26) is intended to simulate send rules of the original protocol. We use here instances of the rules to be sure that terms in the canonical form can be obtained. One can prove the following fact.

Lemma 16. *Let t be a $\oplus$-linear term over Σ and σ be a substitution over Σ such that, for each variable x, $\sigma(x)$ is of the form $s_1 \oplus \cdots \oplus s_n$, for $n \geq 1$, where $s_1, \ldots, s_n$ are standard terms not in $\mathcal{C}_P$. Then we have $\ulcorner t\theta \urcorner = \ulcorner t \urcorner \ulcorner \theta \urcorner$*

Note that the protocol P^+ does not contain $\oplus$ and thus, by [14], existence of an attack on P^+ is decidable. Hence, by the two following lemmas, existence of an attack on $\oplus$-linear protocols is also decidable.

We extend the protocol P by the following protocol steps (we omit send programs, if they are empty):

$$0 \to I([0]) \tag{18}$$

$$[k] \to I(k) \qquad \text{for } k \in \mathcal{K} \tag{19}$$

$$\langle [c], [c'] \rangle \to I([c'']) \qquad \text{for } c, c', c'' \in \mathcal{C}_P \text{ with } c, c' \vdash^{\mathrm{xor}}_{T_\oplus} c'' \tag{20}$$

$$\langle \{x^c\}_{[c']}, \{x^c\}_{[c'']} \rangle \to I([c]) \qquad \text{for each } c', c'' \in \mathcal{C}_P, \text{ where } c \sim c' \oplus c'' \text{ and } x^c \text{ is a fresh variable} \tag{21}$$

$$\langle [c_0], \{y_1^{i,\bar{c}}\}_{[c_1]}, \ldots, \{y_n^{i,\bar{c}}\}_{[c_n]} \rangle \to I(\{\{y_1^{i,\bar{c}}, \{\ldots, \{y_{n-1}^{i,\bar{c}}, y_n^{i,\bar{c}}\}_e \ldots\}_e\}_e\}_{[c]}) \qquad \text{for each } i \in \{1, \ldots, M\} \text{ and for distinct } [c_0], \ldots, [c_n], \text{ where } c \sim c_0 \oplus \cdots \oplus c_n \text{ and } y^{i,\bar{c}} \text{ is a fresh variable, for } \bar{c} = c_0 \ldots c_n \tag{22}$$

$$z_i \to p(z_i) \qquad \text{with the following send program:} \tag{23}$$

$$p(\langle x, y \rangle) \Rightarrow p(y),\ p'(x) \tag{24}$$

$$p'(\langle \{x\}_{[c]}, [c'] \rangle) \Rightarrow I(\{x\}_{[c \oplus c']}) \qquad \text{for } c, c' \in \mathcal{C}_P \tag{25}$$

for $i = 1, \ldots, N$, where z^i is a fresh variable and p is a fresh predicate symbol.

To each send program Φ of P we add the rules

$$q(\ulcorner t\kappa \urcorner) \Rightarrow I(\ulcorner s\kappa \urcorner) \qquad \text{for } (q(t) \to I(s)) \in \Phi,\ \kappa \in \Theta(\mathrm{dom}(t)) \tag{26}$$

Fig. 3. Protocol P^+

A *pre-derivation* is a sequence $a_1, \ldots, a_n$ such that a_i can be derived from $a_1, \ldots, a_{i-1}$ (in some number of steps). A pre-derivation can be seen as an incomplete derivation. One can easily see that if there is a pre-derivation, then there is also a derivation. We say that a (pre-) derivation is $\oplus$–free, if it does not use the XOR operator.

Lemma 17. *If a $\oplus$-linear protocol P is insecure, then P^+ is insecure.*

Proof. Assume that δ is a good derivation for $I(0) \vdash_{P(\theta)} I(\mathsf{Sec})$, for some substitution θ. We split δ into $\delta_1 \bar{\delta}_1 \cdots \delta_N \bar{\delta}_N$ such that, for $1 \leq i \leq N$, $\bar{\delta}_i$ is a (possibly empty) block of elements obtained by the variant rule, and no element in δ_i is obtained by this rule. We define a substitution θ' in the following way:

(i) For $x \in \mathrm{dom}(\theta)$, we put $\theta'(x) = \ulcorner \theta(x) \urcorner$

(ii) If $I(c)$ in δ is obtained by taking exclusive or of $I(c' \oplus t)$ and $I(c'' \oplus t)$, then we put $\theta'(x^c) = \ulcorner t \urcorner$ (see rule (21)).

(iii) If $I(c \oplus t_1 \oplus \cdots \oplus t_m)$ is the k-th long term derived in δ and it is obtained from $I(c_0), I(c_1 \oplus t_1), \ldots, I(c_m \oplus t_m)$, then we put $\theta'(y_i^{k,\bar{c}}) = \ulcorner t_i \urcorner$, for $i = 1, \ldots, m$ and $\bar{c} = c_0 \ldots c_m$ (see rule (22)).

(iv) If $\bar{\delta}_k$ consists of elements $I(c_i \oplus t_i)$, for $i = 1, \ldots, m$, and these elements are obtained from $I(c'_i \oplus t_i)$ and $I(c''_i)$, then we put

$$\theta'(z^k) = \langle \langle \{\ulcorner t_1 \urcorner\}_{[c'_1]}, [c''_1] \rangle, \ldots, \langle \{\ulcorner t_m \urcorner\}_{[c'_m]}, [c''_m] \rangle \rangle.$$

For each $\bar{\delta}_k$ we define $\breve{\delta}_k$ in the following way: $\breve{\delta}_k$ contains the following elements: $I(\theta'(z^k))$, then $p(\theta'(z^k))$, and finally $I(\{\ulcorner t_i \urcorner\}_{[c_i]})$, for $i \in \{1, \ldots, m\}$, where t_i and c_i are terms given by definition of $\theta'(z^k)$.

We will show that $\rho = \ulcorner \delta_1 \urcorner \breve{\delta}_1 \cdots \ulcorner \delta_N \urcorner \breve{\delta}_N$ is a $\oplus$–free pre-derivation for $I(0) \vdash_{P^+(\theta')} I(\mathsf{Sec})$. So, we will show that each element $\rho(i)$ of ρ can be derived in some number of steps from $\rho_{<i}$. In case of the elements of $\breve{\delta}_k$, we note that, because $I(c_i \oplus t_i) \in \bar{\delta}(k)$ is obtained by the variant rule from $I(c_i' \oplus t)$ and $I(c_i'')$, so, by Lemma 9, we have $I(c_i''), I(c_i' \oplus t) \in \delta_1 \cdots \delta_k$ and thus $I([c_i''])$ and $I(\{\ulcorner t \urcorner\}_{[c_i']})$ are in $\ulcorner \delta_1 \urcorner \cdots \ulcorner \delta_k \urcorner$, which suffices to derive $I(\theta'(z^i))$. Further, $p(\theta'(z^k))$ can be derived from $I(\theta'(z^k))$ by (23). Now, $I(\{\ulcorner t_i \urcorner\}_{[c_i]})$, for $i = 1, \ldots, m$, can be derived using (24)–(25).

Now, consider an element $\rho(k) = \ulcorner \delta_i(j) \urcorner$, for some i and j. Note that if $a \in \bar{\delta}_i$, then $\ulcorner a \urcorner \in \breve{\delta}_i$. Note also that if a occurs in δ before $\delta_i(j)$, then $\ulcorner a \urcorner \in \rho_{<k}$. We consider the following cases:

- If we assume that $\delta_i(j)$ was obtained either by a standard intruder rule or by a push rule, then it is easy to check that $\ulcorner \delta_i(j) \urcorner$ can be easily derived (we use here (19) and (20)).
- $\delta_i(j)$ was obtained by a send rule. Hence, $\delta_i(j) = I(s\sigma)$ was obtained from $q(r\sigma)$, for some send rule $q(r) \to I(s)$. We have $q(\ulcorner r\sigma \urcorner) \in \rho_{<k}$. One can check that there exists a substitution σ' and a substitution $\kappa \in \Theta(\mathrm{dom}(r))$ such that $\sigma = \kappa\sigma'$ and, for each variable x, $\sigma'(x)$ is of the form $s_1 \oplus \cdots \oplus s_n$, for $n \geq 1$, where $s_1, \ldots, s_n$ are standard terms not in $\mathcal{C}_P$. Hence, by Lemma 16, we have $q(\ulcorner r\kappa \urcorner \ulcorner \sigma' \urcorner) = q(\ulcorner r\kappa\sigma' \urcorner) = q(\ulcorner r\sigma \urcorner) \in \rho_{<k}$. So, we can derive $I(\ulcorner s\kappa \urcorner \ulcorner \sigma' \urcorner) = I(\ulcorner s\sigma \urcorner)$ from $\rho_{<k}$ in one step by (26).
- If $\delta_i(j)$ was obtained by a protocol rule, i.e. $\delta_i(j) = r(x\theta)$ was obtained from $I(x\theta)$, for a protocol rule $x \to r(x)$, then we have $I(x\theta') = \ulcorner I(x\theta) \urcorner \in \rho_{<k}$, by definition of θ', and so we can derive $r(x\theta') = r(\ulcorner x\theta \urcorner)$.
- If $\delta_i(j)$ was obtained by (9) or (7), then $\rho(k)$ can be obtained by the standard push or pop rules.
- $\delta_i(j) = I(c \oplus c')$ was obtained from $I(c \oplus t_1 \oplus \cdots \oplus t_n)$ and $I(c' \oplus t_1 \oplus \cdots \oplus t_n)$ using (6). So, we can derive $I(\{\ulcorner t_1 \oplus \cdots \oplus t_n \urcorner\}_{[c]})$ and $I(\{\ulcorner t_1 \oplus \cdots \oplus t_n \urcorner\}_{[c']})$ from $\rho_{<k}$ (if $c \not\sim 0$ then we have $a = I(\{\ulcorner t_1 \oplus \cdots \oplus t_n \urcorner\}_{[c]})$ in $\rho_{<k}$; otherwise, $I(\ulcorner t_1 \oplus \cdots \oplus t_n \urcorner) \in \rho_{<k}$, and we can derive a by the encryption rule). Hence $I([c' \oplus c])$ can be derived from $\rho_{<k}$ by Rule (21) or (20) (if $n = 0$).
- $\delta_i(j) = I(c_0 \oplus c_1 \oplus \cdots \oplus c_n \oplus t_1 \oplus \cdots \oplus t_n)$ is the k-th long term derived in δ and is obtained from $I(c_0)$ and $I(c_1 \oplus t_1), \ldots, I(c_n \oplus t_n)$, for some $n > 1$, using (10). Then we can obtain the element $I(\{\ulcorner t_1, \ldots, t_n \urcorner\}_{[c_0 \oplus \cdots \oplus c_n]})$ by Rule (22). If $c_0 \oplus \cdots \oplus c_n \sim 0$, then in an additional step we can obtain $I(\ulcorner t_1, \ldots, t_n \urcorner)$. □

We complete the proof of Theorem 3 by the following lemma which says, roughly speaking, that P^+ does not allow to derive more attacks than P does.

Lemma 18. *Let P be a $\oplus$-linear protocol. If P^+ is insecure, then P is insecure.*

Proof. For a term t we define $\llcorner t \lrcorner$ as follows.

$$\begin{aligned} \llcorner [c] \lrcorner &= c & \llcorner k \lrcorner &= k \\ \llcorner \langle t, s \rangle \lrcorner &= \langle \llcorner t \lrcorner, \llcorner s \lrcorner \rangle & \llcorner \{\!|t|\!\}_k \lrcorner &= \{\!|\llcorner t \lrcorner|\!\}_k \\ \llcorner \{t\}_k \lrcorner &= \{\llcorner t \lrcorner\}_k & \llcorner \{t\}_{[c]} \lrcorner &= c \oplus \llcorner t \lrcorner \\ \llcorner \{t, s\}_e \lrcorner &= \llcorner t \lrcorner \oplus \llcorner s \lrcorner & \llcorner \mathsf{hash}(t) \lrcorner &= \mathsf{hash}(\llcorner t \lrcorner) \end{aligned}$$

First, one can prove, by induction on the structure of terms, that, for each term t over Σ and each substitution σ over Σ', it holds

$$\llcorner \ulcorner t \urcorner \phi \lrcorner \sim t \llcorner \phi \lrcorner, \tag{27}$$

Now, let $\delta = a_1, \ldots, a_n$ be a $\oplus$–free derivation for $I(0) \vdash_{P^+(\theta)} I(\mathsf{Sec})$. Let $\delta' = a'_1, \ldots, a'_n$, where $a'_i = I(0)$, if a_i is of the form $p(t)$ or $p'(t)$ (recall that p and p' are fresh predicate symbols introduced in P^+) and $a'_i = \llcorner a_i \lrcorner$, otherwise. We will show that δ' is a pre-derivation for $I(0) \vdash_{P(\llcorner \theta \lrcorner)} I(\mathsf{Sec})$. In order to do it, we show that $\delta'(i)$ can be derived from $\delta'_{<i}$ in some number of steps. We consider a number of cases. If $\delta(i)$ is obtained either by some intruder rule, a protocol rule of P, a protocol rule introduced in P^+, a push clause of P, or a push rule introduced in P^+, then the proof is straightforward, so we omit it. Let us consider the remaining two cases:

- $\delta(i)$ is obtained by a send clause (26). Hence, $\delta(i)$ is of the form $I(\ulcorner s\kappa \urcorner \sigma)$ and is obtained from $q(\ulcorner r\kappa \urcorner \sigma)$, for some $q(r) \to I(s) \in \Phi$. So, we have $\delta'_{<i} \ni q(\llcorner \ulcorner r\kappa \urcorner \sigma \lrcorner) \sim q(r\kappa \llcorner \sigma \lrcorner)$ (the last equivalence is by (27)). Thus, we can derive $I(s\kappa \llcorner \sigma \lrcorner)$ which, by (27), is equivalent to $I(\ulcorner s\kappa \urcorner \sigma)$.
- $\delta(i)$ is obtained by a send clause (25). So, $\delta(i) = I(\{t\}_{[c \oplus c']})$ is obtained from $p'(\langle \{t\}_{[c]}, [c'] \rangle)$. One can check that $I(\{t\}_{[c]}) \in \delta_{<i}$ and $I([c']) \in \delta_{<i}$. Hence $I(\llcorner \{t\}_{[c]} \lrcorner) = I(c \oplus \llcorner t \lrcorner) \in \delta'_{<i}$ and $I(\llcorner [c'] \lrcorner) = c' \in \delta'_{<i}$. So, we can derive $I(c \oplus c' \oplus \llcorner t \lrcorner) = \delta'(i)$. □

References

1. J.A. Bull and D.J. Otway. The authentication protocol. Technical Report DRA/CIS3/PROJ/CORBA/SC/1/CSM/436-04/03, Defence Research Agency, Malvern, UK, 1997.
2. Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. An NP Decision Procedure for Protocol Insecurity with XOR. In *LICS 2003*, pages 261–270. IEEE, Computer Society Press, 2003.
3. Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents. In *FSTTCS 2003*, volume 2914 of *LNCS*, pages 124–135. Springer, 2003.
4. Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. Deciding the Security of Protocols with Commuting Public Key Encryption. *ENTCS*, 125(1):55–66, 2005.
5. H. Comon-Lundh and V. Cortier. New Decidability Results for Fragments of First-order Logic and Application to Cryptographic Protocols. In *RTA 2003*, volume 2706 of *LNCS*, pages 148–164. Springer, 2003.

6. H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *LICS 2003*, pages 271–280. IEEE, Computer Society Press, 2003.
7. V. Cortier, S. Delaune, and P. Lafourcade. A survey of algebraic properties used in cryptographic protocols. *Journal of Computer Security*, 14(1):1–43, 2006.
8. R. Küsters. On the Decidability of Cryptographic Protocols with Open-ended Data Structures. *International Journal of Information Security*, 4(1–2):49–70, 2005.
9. R. Küsters and T. Wilke. Automata-based Analysis of Recursive Cryptographic Protocols. In *STACS 2004*, volume 2996 of *LNCS*, pages 382–393. Springer, 2004.
10. L.C. Paulson. Mechanized Proofs for a Recursive Authentication Protocol. In *CSFW-10*, pages 84–95. IEEE Computer Society Press, 1997.
11. O. Pereira and J.-J. Quisquater. A Security Analysis of the Cliques Protocols Suites. In *CSFW-14*, pages 73–81, IEEE Computer Society Press, 2001.
12. P.Y.A. Ryan and S.A. Schneider. An Attack on a Recursive Authentication Protocol. *Information Processing Letters*, 65(1):7–10, 1998.
13. V. Shmatikov. Decidable Analysis of Cryptographic Protocols with Products and Modular Exponentiation. In *(ESOP 2004)*, volume 2986 of *LNCS*, pages 355–369. Springer, 2004.
14. T. Truderung. Selecting theories and recursive protocols. In *CONCUR 2005*, volume 3653 of *LNCS*, pages 217–232. Springer, 2005.
15. K.N. Verma, H. Seidl, and T. Schwentick. On the complexity of equational horn clauses. In *CADE 2005*, volume 3328 of *LNCS*, pages 337–352. Springer, 2005.