Steve Kremer, Antoine Mercier,
Ralf Treinen

Proving Group Protocols Secure Against Eavesdroppers

Research Report LSV-08-24

July 2008

Laboratoire
Spécification
et
Vérification

CENTRE NATIONAL
DE LA RECHERCHE
SCIENTIFIQUE

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

Proving Group Protocols Secure Against Eavesdroppers

Steve Kremer[1], Antoine Mercier[1], and Ralf Treinen[2]

[1] LSV, ENS Cachan, CNRS, INRIA, France
[2] PPS, Université Paris Diderot, CNRS, France

Abstract. Security protocols are small programs designed to ensure properties such as secrecy of messages or authentication of parties in a hostile environment. In this paper we investigate automated verification of a particular type of security protocols, called *group protocols*, in the presence of an eavesdropper, i.e., a passive attacker. The specificity of group protocols is that the number of participants is not bounded.
Our approach consists in representing an infinite set of messages exchanged during an unbounded number of sessions, one session for each possible number of participants, as well as the infinite set of associated secrets. We use so-called visibly tree automata with memory and structural constraints (introduced recently by Comon-Lundh et al.) to represent over-approximations of these two sets. We identify restrictions on the specification of protocols which allow us to reduce the attacker capabilities guaranteeing that the above mentioned class of automata is closed under the application of the remaining attacker rules. The class of protocols respecting these restrictions is large enough to cover several existing protocols, such as the GDH family, GKE, and others.

1 Introduction

Many modern computing environments, on wired or wireless networks, involve groups of users of variable size, and hence raise the need for secure communication protocols designed for an unbounded number of participants. This situation can be encountered in multi-user games, conferencing applications, or when securing an ad-hoc wireless network. In this paper we investigate the formal analysis of such protocols whose specification is parameterized by the number of participants. Proving protocols by hand is cumbersome and error-prone. Therefore we aim at automated proof methods. The variable number of protocol participants makes this task particularly difficult.

Related works. Several works already attempted to analyze these protocols. Steel developed the CORAL tool [14] which aims at searching for attacks. Pereira and Quisquater [13, 12] analyzed specific types of protocols and proved the impossibility of building secure protocols using only some kinds of cryptographic primitives such as modular exponentiation in the presence of an active adversary. Küsters and Truderung [11, 16] studied the automatic analysis of group protocols in case of a bounded number of sessions for an active intruder. They showed that the secrecy property is decidable for a given number of participants (without bound on this number) and for a group protocol that may be encoded in their model. As far as we know there is no complete and generic method to automatically prove the security of protocols for an unbounded number of participants. We aim to establish such a method.

Contribution of this paper. We consider a passive intruder, that is an intruder who only eavesdrops all the messages emitted during any sessions. This setting is of course restrictive in comparison to an active adversary. However, Katz and Yung [10] have shown that any group key agreement protocol which is secure against a passive adversary can be transformed into a protocol which resists against an active adversary. The security property that we are interested in here is the secrecy of some set of messages. In a group key exchange protocol, for example, the set of messages we hope remain secret is the set of session keys established during several sessions of the protocol.

Vulnerability of protocols designed for a *fixed* number of participants to eavesdropping attacks is quite straightforward to analyze in the symbolic model of Dolev and Yao. This is due to the fact that the set of messages exchanged during one session is a finite set, and that one can construct easily [9] a tree automaton describing precisely the set of messages an attacker can deduce from these using the deduction capabilities of the Dolev/Yao model. This is much more difficult in case of group protocols: not only does the number of messages exchanged during a protocol session grow with the number of participants, one also has to take into consideration parameters like the total number of participants of a session, and the position of a participant among the other protocol participants.

We specify a protocol as a function that assigns to any number of participants two sets of terms. The first set represents the set of messages emitted by the participants of the protocol, and the second set represents the set of terms that we want to remain secret. We suppose that the attacker has access to the set of messages exchanged by the legitimate protocol participants of the sessions for *all* possible numbers of participants combined, and that he attempts to deduce from this a supposed secret of *one* of the sessions. In other words, we are interested in the situation where an attacker could listen to several protocol sessions, say for 3, 4, and 5 participants, and then use this combined knowledge in order to deduce a supposed secret for one of these sessions.

As a first step we give sufficient restrictions on protocol specifications which allow us to reduce the intruder capabilities: we show that operations typically used in group protocols such as modular exponentiation and *exclusive or* are not necessary for the intruder to discover a secret term. As a consequence, the deduction capabilities of an intruder can be reduced to the classical so-called passive *Dolev-Yao* intruder. These restrictions are met by the protocols we have studied ($[15, 2, 1], \ldots$). In contrast to classical protocols between a fixed number of participants, however, we now have to deal with infinite sets of terms that are in general inductively defined.

The second step is to represent an over-approximation of the set of emitted messages and the set of supposed secrets in a formalism that has the following features:

- The set of messages that the intruder can deduce from the set of emitted messages can again be described by the same formalism.
- Disjointness of sets described in this formalism is decidable.

The formalism of classical tree automata enjoys these two properties; unfortunately it is not expressive enough to specify the inductively defined sets of messages that occur in group protocols. For this reason we employ here the recently [3] proposed class of so-called *visibly tree automata with memory, visibility and structural constraints*. Additionally, this constitutes a new unexpected application of this class of automata.

A further difficulty is that the messages may still use associative and commutative (AC) operators. The class of automata used here does not take into account AC operators. We will explain how to cope with this difficulty by working with certain representatives of equivalence classes modulo AC.

Structure of the paper. Section 2 presents the running example used throughout the paper. In Section 3 we introduce our attacker model. In Section 4 we explain the result on the reduction of intruder capabilities and its proof. The details of the proof are in the appendix. In Section 5 we exhibit how to represent our sets of terms using the formalism of [3]. We illustrate our technique with the representation of the running example in this formalism in Section 6. We conclude in Section 7.

2 Running Example

Our running example is the group Diffie-Hellman key agreement protocol (GDH-2) [15]. Every participant in a session between n participants generates a nonce N_i (a secret fresh value). The first participant sends out $[\alpha, \alpha^{N_1}]$ where α is a publicly known constant. The i-th participant (for $1 < i < n$) expects from its predecessor a list of messages $[\alpha^{x_1}, \ldots, \alpha^{x_i}]$, and he sends out $[\alpha^{x_i}, \alpha^{x_1 \cdot N_i}, \ldots, \alpha^{x_i \cdot N_i}]$. The last participant, on reception of $[\alpha^{x_1}, \ldots, \alpha^{x_n}]$, sends to all other participants $[\alpha^{x_1 \cdot N_n}, \ldots, \alpha^{x_{n-1} \cdot N_n}]$. For instance in case of 4 participants the following sequence of messages is sent (we use here a list notation that will later be formalised by a binary pairing operator) :

Sender	Message
1	$[\alpha, \alpha^{N_1}]$
2	$[\alpha^{N_1}, \alpha^{N_2}, \alpha^{N_1 \cdot N_2}]$
3	$[\alpha^{N_1 \cdot N_2}, \alpha^{N_1 \cdot N_3}, \alpha^{N_2 \cdot N_3}, \alpha^{N_1 \cdot N_2 \cdot N_3}]$
4	$[\alpha^{N_1 \cdot N_2 \cdot N_4}, \alpha^{N_1 \cdot N_3 \cdot N_4}, \alpha^{N_2 \cdot N_3 \cdot N_4}]$

The common key is $\alpha^{N_1 \cdot N_2 \cdot N_3 \cdot N_4}$. Note that each of the participants i for $i < n$ can calculate that key from the message sent out by the last participant since

$$x \oplus 0 \to x$$
$$x \oplus x \to 0$$
$$((x)^y)^z \to x^{y \cdot z}$$
$$\langle x, y \rangle^z \to \langle x^z, y^z \rangle$$

Fig. 1. Rewrite System R

he knows the missing part N_i, and that the participant n can calculate that key using the last element of the sequence he received form the participant $n-1$.

3 Model

We present here an extension of the model of a passive Dolev-Yao intruder [8]. This intruder is represented by a deduction system which defines his capabilities to obtain new terms from terms he already knows.

Messages. Messages exchanged during executions of the protocol are represented by terms over the following signature Σ:

$$\Sigma = \{\mathsf{pair}/2, \mathsf{enc}/2, \mathsf{exp}/2, \mathsf{mult}/2, \mathsf{xor}/2, H/1\} \uplus \Sigma_0$$

A pair $\mathsf{pair}(u, v)$ is usually written $\langle u, v \rangle$, the encryption of a message u by the key v, $\mathsf{enc}(u, v)$, is written $\{u\}_v$, and the exponentiation of u by v, $\mathsf{exp}(u, v)$, is written u^v. Multiplication mult and exclusive or xor are denoted by the infix operators $\cdot$ and $\oplus$. The symbol H denotes a unary hash function. Σ_0 is an infinite set of constant symbols, including nonces and encryption keys, and possibly elements of an algebraic structure such as the generator of some group. $\mathcal{T}(\Sigma)$ denotes the set of all terms build over Σ. We write $St(t)$ for the set of subterms of the term t, defined as usual, and extend this notation to sets of terms. We say that a function symbol f, where $f/n \in \Sigma$, *occurs in* a term t if $f(t_1, \ldots, t_n) \in St(t)$ for some $t_1, \ldots, t_n$.

Equational theory. We extend this model by an equational theory represented by the rewrite system R, which is given in Figure 1, modulo AC. The associative and commutative operators are *exclusive or* $\oplus$ and multiplication $\cdot$. Normalization by R modulo AC is denoted $\downarrow_{R/AC}$. The first two rules express the neutral element and the nilpotency law of *exclusive or*. The third rewrite rule allows for a normalization of nested exponentiations. Note that we do not consider a neutral element for multiplication, and that we do not have laws for multiplicative inverse, or for distribution of multiplication over other operations such as addition. The last rule allows one to normalize a list of terms exponentiated with the same term. It will be useful in order to model some protocols such as GKE [2]. Confluence and termination of this rewrite system have been proven

$$\frac{}{S \vdash t}\ axiom \text{ if } t \in S \qquad \frac{S \vdash t_1 \quad S \vdash t_2}{S \vdash \langle t_1, t_2 \rangle}\ pair$$

$$\frac{S \vdash \langle t_1, t_2 \rangle}{S \vdash t_1}\ proj_1 \qquad \frac{S \vdash \langle t_1, t_2 \rangle}{S \vdash t_2}\ proj_2$$

$$\frac{S \vdash t_1 \quad S \vdash t_2}{S \vdash \{t_1\}_{t_2}}\ enc \qquad \frac{S \vdash \{t_1\}_{t_2} \quad S \vdash t_2}{S \vdash t_1}\ dec$$

$$\frac{S \vdash t}{S \vdash H(t)}\ hash$$

Fig. 2. The Dolev-Yao Deduction System *DY*

$$\frac{S \vdash t_1 \quad S \vdash t_2 \text{ and } t_2 \in \Sigma_0}{S \vdash t_1^{t_2} \downarrow_{R/AC}}\ exp \qquad \frac{S \vdash t_1 \ \cdots \ S \vdash t_n}{S \vdash t_1 \oplus \cdots \oplus t_n \downarrow_{R/AC}}\ Gxor$$

Fig. 3. Extension of the Dolev-Yao Deduction System

using the tool C*i*ME [6]. In the rest of this paper we will only consider terms in normal form modulo R/AC. Any operation on terms, such as the intruder deduction system presented below, has to yield normal forms modulo R/AC.

The Intruder. The deduction capabilities of the intruder are described as the union of the deduction system *DY* of Figure 2 and the system of Figure 3. The complete system is called I. A sequent $S \vdash t$, where S is a finite set of terms and t a term, expresses the fact that the intruder can deduce t from S.

The system *DY* represents the classical Dolev-Yao intruder capacities to encrypt (*enc*) or decrypt (*dec*) a message with keys he knows, to build pairs (*pair*) of two messages, to extract one of the messages from a pair ($proj_1$, $proj_2$), and finally to apply the hash function (*hash*). Note that the term in the conclusion is in normal form w.r.t. R/AC when its hypotheses are, we hence do not have to normalize the term in the conclusion.

The system I extends the capabilities of *DY* by additional rules that allow the intruder to apply functions that are subject to the equational theory. We have to normalize the terms in the conclusions in these rules since applications of exponentiation or $\oplus$ may create new redexes.

As usually *pair*, *enc*, *hash*, *exp* and *Gxor* will be called *construction* rules, and $proj_1$, $proj_2$, and *dec*, will be called *deconstruction* rules. Note that the *Gxor* rule may also be used to deduce a subterm of a term thanks to the nilpotency of $\oplus$. For instance, let a and b be two constants of Σ_0. Applying a *Gxor* rule to sequents $S \vdash a \oplus b$ and $S \vdash b$ allows to deduce a, a subterm of $a \oplus b$. We implicitly assume that the constants 0 and α are always included in S.

For a set of ground terms S and a ground term t we write $S \vdash_D t$ if there exists a deduction of the term t from the set S in the deduction system D.

Definition 1 (Deduction). *A deduction of t from S in a system D is a tree where every node is labelled with a sequent. A node labelled with $S \vdash_D u$ has n sons $S \vdash_D v_1, \ldots, S \vdash_D v_n$ such that $\frac{S \vdash_D v_1, \ldots, S \vdash_D v_n}{S \vdash_D u}$ is an instance of one of the rules of D. The root is labelled by $S \vdash_D t$. The size of the deduction is the number of nodes of the tree.*

When the deduction system D is clear from the context we may sometimes omit the subscript D and just write $S \vdash t$. In the following we consider deductions in the systems DY and I. The *deductive closure* by a system D of a set of terms E is the set of all the terms the intruder can deduce from E using D.

Definition 2 (Deductive closure). *Let D be a deduction system and T a set of ground terms. The* deductive closure *by D of T is*

$$D(T) = \{t \mid T \vdash_D t\}$$

Protocol Specification and Secrecy Property. We suppose that a protocol is described by two functions $e : \mathbb{N} \to 2^{\mathcal{T}(\Sigma)}$ and $k : \mathbb{N} \to 2^{\mathcal{T}(\Sigma)}$. Given a number of participants n, $e(n)$ yields the set of terms that are emitted during a protocol execution with n participants and $k(n)$ is the set of secrets of this execution, typically the singleton set consisting of the constructed key.

Example 1. Consider our running example introduced Section 2. We have that

$$e_{\mathsf{GDH}}(n) = \begin{cases} \emptyset & \text{if } n < 2 \\ \{t_1^n, \ldots, t_n^n\} & \text{else} \end{cases} \quad \text{and} \quad k_{\mathsf{GDH}}(n) = \begin{cases} \emptyset & \text{if } n < 2 \\ \{\alpha^{N_1^n \cdot \ldots \cdot N_n^n}\} & \text{else} \end{cases}$$

where

$$\begin{aligned} t_1^n &= \langle \alpha, \alpha^{N_1^n} \rangle \\ t_i^n &= \langle \alpha^{N_1^n \cdot \ldots \cdot N_{i-1}^n}, t_{i-1}^{N_i^n} \rangle \qquad (1 < i < n) \\ t_n^n &= \langle \alpha^{N_2^n \cdot \ldots \cdot N_n^n}, \langle \ldots, \langle \alpha^{N_1^n \cdot \ldots \cdot N_{j-1}^n \cdot N_{j+1}^n \cdot \ldots \cdot N_n^n}, \langle \ldots, \alpha^{N_1^n \cdot \ldots \cdot N_{n-2}^n \cdot N_n^n} \rangle \rangle \rangle \rangle \end{aligned}$$

In the following we call a protocol specification the pair of (infinite) sets of terms (E, K) where $E = \bigcup_{n \in \mathbb{N}} e(n)$ and $K = \bigcup_{n \in \mathbb{N}} k(n)$. Given a protocol specification (E, K) we are interested in the question whether a supposed secret in K is deducible from the set of messages emitted in *any* of the sessions, i.e., $I(E) \cap K \stackrel{?}{=} \emptyset$. It is understood that $e(n)$ and $k(n)$, and hence E and K, are closed under associativity and commutativity of exclusive or and multiplication.

4 Reducing I to DY

In this section we show that, under carefully chosen restrictions on the sets E and K, we can consider an intruder who is weaker than expected. We will define *well-formed* protocols that will allow us to reduce a strong intruder (using the system I) to a weaker intruder (using only the system DY). This class is general enough to cover existing group protocols.

We first define a slightly stronger deduction system which will be more convenient for the proofs. Let R' be the rewrite system $R \setminus \{\langle x, y\rangle^z \to \langle x^z, y^z\rangle\}$, and I' the deduction system I where the rewrite system used in the rules *exp* and *Gxor* is R'. We show that any term which can be deduced in I can also be deduced in I'. This allows us to use the system I' which is more convenient for our proofs.

Lemma 1. *Let E be a set of terms. If $E \vdash_I t$ then $E \vdash_{I'} t$, and if $E \vdash_{I \setminus Gxor} t$ then $E \vdash_{I' \setminus Gxor} t$.*

To prove this result it is sufficient to note that each time an exponent is applied to a pair, one can obtain both elements of the pair by projection and apply the exponent to each of the elements before recomposing the pair.

We may however note that in general it is not the case that $E \vdash_{I'} t$ implies $E \vdash_I t$ as it is possible in I' to deduce a term of the form $\langle u, v\rangle^c$ (which would not be in normal form with respect to R).

Well formation. To state our well-formation condition, we define a closure function C on terms that computes an over-approximation of the constants that are deducible in a given term.

Definition 3 (closure). *Let $C : \mathcal{T}(\Sigma) \to 2^{\Sigma_0}$ be the function defined inductively as follows*

$$\begin{aligned} C(c) &= \{c\} \qquad \text{if } c \in \Sigma_0 \\ C(\langle u, v\rangle) &= C(u) \cup C(v) \\ C(\{u\}_v) &= C(u) \\ C(u_1 \oplus \cdots \oplus u_n) &= \textstyle\bigcup_{i=1} C(u_i) \\ C(f(u_1, \ldots, u_n)) &= \emptyset \qquad \text{if } f \neq \langle ., .\rangle, \{.\}. \end{aligned}$$

We extend this definition to sets of terms in the natural way, i.e., $c \in C(S)$ if there exists $u \in S$ such that $c \in C(u)$.

The following lemma states that if a constant c is in the closure of a term t which can be deduced from a set E, then c is also in the closure of E. A direct consequence is that if a constant c can be deduced from E then c is in the closure of E.

Lemma 2. *Let $c \in \Sigma_0$ and t be a term such that $c \in C(t)$. If $E \vdash_{I'} t$ then $c \in C(E)$.*

Corollary 1. *If $c \in \Sigma_0$ and $E \vdash_{I'} c$ then $c \in C(E)$.*

We impose restrictions on the protocols. These restrictions concern the usage of modular exponentiation and the usage of $\oplus$ during the execution of the protocol. We will also restrict the set of supposed secrets.

Definition 4 (Well formation). *A protocol specification* (E, K) *is said to be* well-formed *if it satisfies the following constraints.*

1. *If* $t \in E$ *and if* $\oplus$ *occurs in* t*, then* $t = u \oplus v$ *for some* u *and* v *and*
   - $\oplus$ *does not occur neither in* u *nor in* v
   - $u, v \notin DY(E)$.
2. *Let* $t = u^{c_1 \cdot \ldots \cdot c_n}$ *and* $c_i \in \Sigma_0$ *for all* $1 \leq i \leq n$*. If* $t \in St(E \cup K)$ *then* $c_1, \ldots, c_n \notin C(E)$
3. *For any* $t \in K$*, we have that* $\oplus$ *does not occur in* t*.*

Constraint 1 implies that $\oplus$ only occurs at the root position in terms of E and moreover $\oplus$ is of arity 2. Note that this constraint does not prevent the intruder from constructing terms where $\oplus$ has an arity greater than 2. Constraint 1 additionally requires that u and v cannot be deduced by a Dolev-Yao adversary. Constraint 2 requires that any constant occuring as an exponent in some term of $E \cup S$ cannot be accessed "easily", i.e. is not in the closure C, representing an over-approximation of accessible terms. Adding this constraint seems quite natural, since modular exponentiation is generally used to hide the exponent. This is for instance the case in the Diffie-Hellman protocol which serves as our running example: each participant of the protocol generates a nonce N, exponentiates some of the received values with N which are then send out. If the access to such a nonce would be "easy" then the computation of the established key would also be possible. Finally, constraint 3 requires that the secrets do not contain any xored terms.

These constraints allow us to derive the main theorem of this section and will be useful for automating the analysis of group protocols. While these constraints are obviously restrictive, they are nevertheless verified for several protocols, including GDH and GKE. In particular, the last constraint imposes that the specification of sets of keys must not involve any $\oplus$. This may seem rather arbitrary but the protocols we looked at fulfill this requirement.

The main theorem of this section states that, for well-formed protocols, if there is an attack for the attacker I then there is an attack for the attacker DY.

Theorem 1. *For all well-formed* (E, K) *we have* $I(E) \cap K = DY(E) \cap K$*.*

The proof of this result relies on several additional lemmas and is postponed to the end of this section. We only present some of the key lemmas. Remaining details and full proofs are given in Appendix A .

The following lemma is similar to Lemma 1 of [5], but adapted to our setting.

Lemma 3. *Let* E *be a set of terms. If* π *is a minimal deduction in* I' *of one of the following forms :*

$$\dfrac{\dfrac{\vdots}{E \vdash \langle u, v\rangle}}{E \vdash u}\, proj_1 \qquad \dfrac{\dfrac{\vdots}{E \vdash \langle u, v\rangle}}{E \vdash v}\, proj_2 \qquad \dfrac{\dfrac{\vdots}{E \vdash \{u\}_v} \quad \dfrac{\vdots}{E \vdash v}}{E \vdash u}\, dec$$

Then $\langle u, v\rangle \in St(E)$ *(resp.* $\langle u, v\rangle \in St(E)$*, resp.* $\{u\}_v \in St(E)$*).*

We now show that in the case of well-formed protocol specifications, if a deduction does not apply the *Gxor* rule, then it does not need to apply the *exp* rule neither.

Lemma 4. *Let E be a set of terms and t a term. If for every $u^{c_1 \cdot \ldots \cdot c_n} \in St(E, t)$ such that $c_i \in \Sigma_0$ one has $c_i \notin C(E)$ and if $E \vdash_{I' \setminus Gxor} t$ then $E \vdash_{DY} t$.*

The proof is done by induction on the length of the deduction tree showing that $E \vdash_{I' \setminus Gxor} t$. The delicate case occurs when the last rule application is either projection or decryption. In these cases we rely on Lemma 3.

The next lemma states that whenever a *Gxor* rule is applied then a $\oplus$ occurs in the conclusion of the deduction. A direct corollary allows us to get rid of any application of the *Gxor* rule in well-formed protocol specifications.

Lemma 5. *Let E be a set of terms satisfying constraints 1 and 2 of Definition 4. Let π be a minimal deduction of $E \vdash_{I'} t$. If π involves an application of the Gxor rule, then $\oplus$ occurs in t.*

Corollary 2. *Let (E, K) be a well-formed protocol. Every minimal deduction of $E \vdash_{I'} t$ such that $t \in K$ does not involve an application of the Gxor rule.*

Proof. By contradiction. Let π be a minimal deduction of $E \vdash_{I'} t$ that involves a *Gxor* rule. As (E, K) is a well-formed protocol, by Lemma 5 $\oplus$ occurs in t. However, as $t \in K$ (E, K) is a well-formed protocol, by constraint 3, $\oplus$ does not occur in t.

We are now ready to prove the main theorem of this section.

Proof (of Theorem 1). We obviously have that $DY(E) \cap K \subseteq I(E) \cap K$ since DY is a subsystem of I. To prove the other direction, let t be a term of K and suppose that $E \vdash_I t$. By Lemma 1, there is a deduction of $E \vdash_{I'} t$. As (E, K) is well-formed and $t \in K$, by Corollary 2, there exists a deduction of $E \vdash_{I' \setminus Gxor} t$. Hence by Lemma 4, we have a deduction of $E \vdash_{DY} t$.

5 Representing Group Protocols by Automata

5.1 The Automaton Model

We first recall the definition of *Visibly Tree Automata with Memory and Structural Constraints* introduced first in [3] and later refined in [4]. Let $\mathcal{X}$ be an infinite set of variables. The set of terms built over Σ and $\mathcal{X}$ is denoted $T(\Sigma, \mathcal{X})$.

Definition 5 ([3]). *A* bottom-up tree automaton with memory *on a finite input signature Σ is a tuple (Γ, Q, Q_f, Δ) where Γ is a memory signature, Q is a finite set of unary state symbols, disjoint from $\Sigma \cup \Gamma$, $Q_f \subseteq Q$ is the subset of final states and Δ is a set of rewrite rules of the form $f(q_1(m_1), \ldots, q_n(m_n)) \to q(m)$ where $f \in \Sigma$ of arity n, $q_1, \ldots, q_n, q \in Q$ and $m_1, \ldots m_n, m \in T(\Sigma, \mathcal{X})$.*

For the next definition we will have to consider a partition of the signature, and we will also (as in [3]) require that all symbols of Σ and Γ have either arity 0 or 2. We also assume that Γ contains the constant $\perp$.

$$\begin{aligned}\Sigma = \Sigma_{\textsc{Push}} \uplus \Sigma_{\textsc{Pop}_{11}} \uplus \Sigma_{\textsc{Pop}_{12}} \uplus \Sigma_{\textsc{Pop}_{21}} \uplus \Sigma_{\textsc{Pop}_{22}} \\ \uplus \Sigma_{\textsc{Int}_0} \uplus \Sigma_{\textsc{Int}_1} \uplus \Sigma_{\textsc{Int}_2} \uplus \Sigma^{\equiv}_{\textsc{Int}_1} \uplus \Sigma^{\equiv}_{\textsc{Int}_2}\end{aligned}$$

Two terms t_1 and t_2 are equivalent, written $t_1 \equiv t_2$, if they are equal when identifying all symbols of the same arity, that is $\equiv$ is the smallest equivalence on ground terms satisfying

- $a \equiv b$ for all a, b of arity 0,
- $f(s_1, s_2) \equiv g(t_1, t_2)$ if $s_1 \equiv t_1$ and $s_2 \equiv t_2$, for all f and g of arity 2.

Definition 6 ([4]). *A visibly tree automaton with memory and constraints (short* $\textsc{Vtam}^{\equiv}_{\not\equiv}$*) on a finite input signature* Σ *is a tuple* $(\Gamma, \equiv, Q, Q_f, \Delta)$ *where* Γ,Q,Q_f *are as in Definition 5,* $\equiv$ *is the relation on* $\mathcal{T}(\Gamma)$ *defined above and* Δ *is the set of rewrite rules of one of the following forms:*

Push	a	$\to$	$q(c)$	$a \in \Sigma_{\textsc{Push}}$
Push	$f(q_1(y_1), q_2(y_2))$	$\to$	$q(h(y_1, y_2))$	$f \in \Sigma_{\textsc{Push}}$
$\textsc{Pop}_{1i}$	$f(q_1(h(y_{11}, y_{12}), q_2(y_2)))$	$\to$	$q(y_{1i})$	$f \in \Sigma_{\textsc{Pop}_{1i}}, 1 \leq i \leq 2$
$\textsc{Pop}_{1i}$	$f(q_1(\perp), q_2(y_2))$	$\to$	$q(\perp)$	$f \in \Sigma_{\textsc{Pop}_{1i}}, 1 \leq i \leq 2$
$\textsc{Pop}_{2i}$	$f(q_1(y_1), q_2(h(y_{21}, y_{22})))$	$\to$	$q(y_{2i})$	$f \in \Sigma_{\textsc{Pop}_{2i}}, 1 \leq i \leq 2$
$\textsc{Pop}_{2i}$	$f(q_1(y_1), q_2(\perp))$	$\to$	$q(\perp)$	$f \in \Sigma_{\textsc{Pop}_{2i}}, 1 \leq i \leq 2$
$\textsc{Int}_0$	a	$\to$	$q(\perp)$	$a \in \Sigma_{\textsc{Int}_0}$
$\textsc{Int}_i$	$f(q_1(y_1), q_2(y_2))$	$\to$	$q(y_i)$	$f \in \Sigma_{\textsc{Int}_i}, 1 \leq i \leq 2$
$\textsc{Int}^{\equiv}_i$	$f(q_1(y_1), q_2(y_2))$	$\xrightarrow{y_1 \equiv y_2}$	$q(y_i)$	$f \in \Sigma_{\textsc{Int}^{\equiv}_i}, 1 \leq i \leq 2$
$\textsc{Int}^{\not\equiv}_i$	$f(q_1(y_1), q_2(y_2))$	$\xrightarrow{y_1 \not\equiv y_2}$	$q(y_i)$	$f \in \Sigma_{\textsc{Int}^{\equiv}_i}, 1 \leq i \leq 2$

where $q_1, q_2, q \in Q$*,* y_1*,* y_2 *are distinct variables of* $\mathcal{X}, c, h \in \Gamma$*.*

A $\textsc{Vtam}^{\equiv}_{\not\equiv}$ can apply a transition of type $\textsc{Int}^{\equiv}_i$ (resp. $\textsc{Int}^{\not\equiv}_i$) to a term $f(q_1(m_1), q_2(m_2))$ only when $m_1 \equiv m_2$ (resp. $m_1 \not\equiv m_2$). A term t is accepted by a $\textsc{Vtam}^{\equiv}_{\not\equiv}$ $\mathcal{A}$ in state $q \in Q$ and with memory $m \in T(\Gamma)$ iff $t \to^* q(m)$. The language $L(\mathcal{A}, q)$ and memory language $M(\mathcal{A}, q)$ of $\mathcal{A}$ in state q are respectively defined by:

$$L(\mathcal{A}, q) = \{t | \exists m \in T(\Gamma), t \to^* q(m)\} \qquad M(\mathcal{A}, q) = \{m | \exists t \in T(\Sigma), t \to^* q(m)\}$$

Theorem 2 ([3],[4]). *The class of languages recognizable by* $\textsc{Vtam}^{\equiv}_{\not\equiv}$ *is closed under Boolean operations, and emptiness of* $\textsc{Vtam}^{\equiv}_{\not\equiv}$ *is decidable.*

Note that the closure under $\cup$, $\cap$ supposes the same partition of the input signature Σ into $\Sigma_{\textsc{Push}}, \Sigma_{\textsc{Pop}_{11}}$ etc.

5.2 Encoding Infinite Signatures

The signature Σ used in the specification of the protocol may be infinite, in particular due to constants that are indexed by the number of a participants of a session. In order to be able to define $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ that recognize E and K we have to find an appropriate finite signature Σ' that contains only constants and binary symbols, and an appropriate function $\rho : T(\Sigma) \to T(\Sigma')$. The function ρ extends in a natural way to sets of terms. We will then use $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ constructions in order to show that $\rho(DY(E)) \cap \rho(K) = \emptyset$. Note that this implies $DY(E) \cap K = \emptyset$ independent of the choice of ρ, though in practice we will define ρ as an injective homomorphism. If ρ is injective then we have that disjointness of $DY(E)$ and K is equivalent to disjointness of $\rho(DY(E))$ and $\rho(K)$.

Example 2. The signature of our running example contains constants N_i^j, denoting the nonce of participant i in session j (where $i \leq j$). To make this example more interesting we could also consider constants K_i^j for symmetric keys between participants i and j (where $i < j$), and K_i^- (resp. K_i^+) for asymmetric decryption (resp. encryption) keys of the participant i.

We choose the finite signature Σ' consisting of the set of constants $\Sigma'_0 = \{0, \alpha\}$, and the set of binary function symbols

$$\Sigma'_2 = \{\mathsf{pair}, \mathsf{enc}, \mathsf{exp}, \mathsf{mult}, \mathsf{xor}, t, H, N, K, K^+, K^-, s, s'\}$$

The function $\rho\colon T(\Sigma) \to T(\Sigma')$ for the running example is defined as follows (using auxiliary functions $\rho_1\colon \mathbb{N} \to T(\Sigma')$ and $\rho_2\colon \{(i,j) \mid i \leq j\} \to T(\Sigma')$):

$$\begin{array}{rlrl}
\alpha & \to \alpha & \mathsf{pair}(u,v) & \to \mathsf{pair}(\rho(u),\rho(v)) \\
0 & \to 0 & \mathsf{enc}(u,v) & \to \mathsf{enc}(\rho(u),\rho(v)) \\
K_i^+ & \to K^+(0,\rho_1(i)) & \mathsf{exp}(u,v) & \to \mathsf{exp}(\rho(u),\rho(v)) \\
K_i^- & \to K^-(0,\rho_1(i)) & \mathsf{mult}(u,v) & \to \mathsf{mult}(\rho(u),t(0,\rho(v))) \\
K_i^j & \to K(0,\rho_2(i,j)) & \mathsf{xor}(u,v) & \to \mathsf{xor}(\rho(u),\rho(v)) \\
N_i^j & \to N(0,\rho_2(i,j)) & H(u) & \to H(0,\rho(u))
\end{array}$$

where we define

$$\begin{array}{ll}
\rho_1(i) = s'(0,\rho_1(i-1)) \text{ if } i > 0 & \rho_2(i,j) = s'(0,\rho_2(i-1,j-1)) \text{ if } i > 0 \\
\rho_1(0) = 0 & \rho_2(0,j) = s(0,\rho_2(0,j-1)) \quad\quad \text{if } j > 0 \\
 & \rho_2(0,0) = 0
\end{array}$$

For instance, $\rho_1(2) = s'(0, s'(0,0))$, and $\rho_2(1,3) = s'(0, s(0, s(0,0)))$. This encoding of pairs has been choosen in order to facilitate the automaton construction in Section 6.

Finally, we have to adapt the deduction system DY to the translation of the signature, yielding a modified deduction system DY' such that $\rho(DY(S)) = DY'(\rho(S))$ for any $S \subseteq T(\Sigma)$.

Example 3. (continued) In our running example we just have to adapt the rule *hash* and replace it by the following variant:

$$\frac{S \vdash t}{S \vdash H(0,t)}\ hash'$$

The other rules remain unchanged.

Lemma 6. *$\rho(DY(S)) = DY'(\rho(S))$ for ρ defined as in Example 2.*

The proof of this lemma can be found in Appendix B .

5.3 Coping with Associativity and Commutativity of xor and mult.

As for classical tree automata, the languages recognized by $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ are in general not closed under associativity and commutativity. In order to cope with this difficulty we define a witness function W on $T(\Sigma')$ which associates to any term t the minimal element of the equivalence class $[t]_{AC}$ w.r.t. the order $\prec_{\Sigma'}$, the lexicographic path order [7] for the following precedence $<_{\Sigma'}$ on Σ':

$$\begin{gathered}0 <_{\Sigma'} \alpha <_{\Sigma'} s <_{\Sigma'} s' <_{\Sigma'} N <_{\Sigma'} K <_{\Sigma'} K^+ <_{\Sigma'}\\ K^- <_{\Sigma'} H <_{\Sigma'} t <_{\Sigma'} \mathsf{xor} <_{\Sigma'} \mathsf{mult} <_{\Sigma'} \mathsf{exp} <_{\Sigma'} \mathsf{enc} <_{\Sigma'} \mathsf{pair}\end{gathered}$$

One verifies easily that $\rho(N_i^j) \prec_{\Sigma'} \rho(N_{i'}^{j'})$ if and only if either $i <_{\mathbb{N}} i'$, or $i = i'$ and $j <_{\mathbb{N}} j'$. We can now easily define the witness function:

Definition 7. *The function $W : T(\Sigma') \mapsto T(\Sigma')$ assigns to any $t' \in T(\Sigma')$ such that $t' = \rho(t)$ the minimal element of $\rho([t]_{AC})$.*

This function extends in a natural way to sets of terms. Now, the disjointness of two sets of terms S_1 and S_2 that are closed under congruence modulo AC is equivalent to the disjointness of $W(S_1)$ and $W(S_2)$.

Theorem 3. *If S is closed under AC then $W(DY'(S)) = DY'(W(S))$.*

5.4 Closure under DY and Compatibility with the Closure under AC

Theorem 4. *For every $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ $\mathcal{A}$, such that $\mathsf{pair}, \mathsf{enc} \notin \{\Sigma'_{\mathrm{INT}^{\equiv}_{1}} \cup \Sigma'_{\mathrm{INT}^{\equiv}_{2}}\}$ and the only constant symbol of Γ is $\bot$, there exists a $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ $\mathcal{A}_{DY}$ such that $L(\mathcal{A}_{DY}) = DY'(L(\mathcal{A}))$.*

The proof is based on the classical technique of completion of the automaton (see [9]), with special care taken to the extension to memory and constraints. The complete construction is given in Appendix B and depends on the partition of the input signature, we illustrate it here for the case where $\mathsf{pair}, \mathsf{enc}, H \in \Sigma_{\mathrm{PUSH}}$. The automaton extends $\mathcal{A}$ by new final states q_{pair}, q_{enc}, and q_H. We also add some new transitions and promote some states to final states:

$$\frac{q_1, q_2 \in Q_f}{\mathsf{pair}(q_1(x), q_2(y)) \to q_{pair}(h(x,y))}\ Pair$$

$$\frac{\mathsf{pair}(q_1(x), q_2(y)) \to q(h(x,y)) \qquad q \in Q_f \qquad L(\mathcal{A}, q_{3-i}) \neq \emptyset}{q_i \in Q_f} \; Proj_i,\ 1 \le i \le 2$$

$$\frac{q_1, q_2 \in Q_f}{\mathsf{enc}(q_1(x), q_2(y)) \to q_{enc}(h(x,y))} \; Enc$$

$$\frac{\mathsf{enc}(q_1(x), q_2(y)) \to q(h(x,y)) \qquad q \in Q_f \qquad L(\mathcal{A}, q_2) \cap L(\mathcal{A}) \neq \emptyset}{q_1 \in Q_f} \; Dec$$

$$\frac{q_1 \in Q_f}{H(q_0(x), q_1(y)) \to q_H(h(x,y))} \; Hash$$

6 Example

Here we propose an over-approximation of the set of computed keys during an unbounded number of sessions of the protocol (one session for each number of participants). An over-approximation of the set of emitted messages and its representation by automata is given in Appendix C .

The approximation we propose to represent is the following:

$$K = \{\alpha^{N_{j_1}^{j_1} \cdot N_{(j_1-1)}^{j_2} \cdots N_1^{j_{j_1}}}\}$$

Here we only give the construction of the automaton $\mathcal{A}_K$ recognizing the set K. K is the set of symbols of the form α^p where p is a product of nonces N_i^j :

- $i = j$ for the maximal nonce N_i^j in p,
- the number of nonces is j, where N_i^j is the maximal nonce,
- for every i such that $1 \le i \le j$, N_i^k belongs to p for some k.

We use the following partition of the signature Σ' in the automata:

$$\Sigma_{PUSH} = \{s', \mathsf{exp}, 0, \alpha\} \qquad \Sigma_{POP_{22}} = \{t\}$$
$$\Sigma_{INT_2^{\equiv}} = \{\mathsf{mult}\} \qquad \Sigma_{INT_2} = \{s, N\}$$

The other symbols can be put into any part of the signature. We define $\Gamma = \{S, S', h, \perp\}$. The automaton $\mathcal{A}_K$ is defined as follows (q_{acc} is the final state):

$$0 \to q_d(\perp) \qquad \alpha \to q_\alpha(\perp)$$

The following transitions check that if a term $t \to^* q_{nent}(m)$ then t is of the form $N(0, s'(0, \ldots s'(0,0)\ldots))$ and $m = S'(0, \ldots S'(0,0)\ldots)$ and the number of S' equals the number of s'. Hence t represents a nonce such that $i = j$.

$$\begin{aligned} s'(q_d(m), q_d(m')) &\to q_{s'ent}(S'(m,m')) \\ s'(q_d(m), q_{s'ent}(m')) &\to q_{s'ent}(S'(m,m')) \\ N(q_d(m), q_{s'ent}(m')) &\to q_{nent}(m') \end{aligned}$$

The following transitions are similar but also allow several S between the S' and the constant 0. We count in the memory only the number of S'. We also check that terms leading to $q_{nonly1s'}$ involve at most one occurrence of the symbol s'.

$$\begin{aligned}
s(q_d(m), q_d(m')) &\rightarrow q_d(m')\\
s'(q_d(m), q_d(m')) &\rightarrow q_{only1s'}(S'(m, m'))\\
s'(q_d(m), q_{only1s'}(m')) &\rightarrow q_{s'}(S'(m, m'))\\
s'(q_d(m), q_{s'}(m')) &\rightarrow q_{s'}(S'(m, m'))\\
N(q_d(m), q_{s'}(m')) &\rightarrow q_n(m')\\
N(q_d(m), q_{only1s'}(m')) &\rightarrow q_{nonly1s'}(m')
\end{aligned}$$

The following transitions remove an S' symbol from the memory.

$$\begin{aligned}
t(q_d(m), q_{nent}(S'(m', m'')) &\rightarrow q_{nt}(m'')\\
t(q_d(m), q_{narg}(S'(m', m'')) &\rightarrow q_{nt}(m'')
\end{aligned}$$

The following transition can be applied between a term that represents a nonce and a term that represents either a product or a nonce n_{jj} on which we have applied one af the above transitions.

$$\mathsf{mult}(q_n(m), q_{nt}(m')) \overset{m \equiv m'}{\rightarrow} q_{narg}(m)$$

The following transition applies (by the memory language of $q_{nonly1s'}$) only if m' is $S'(\perp, \perp)$. In this case the term is considered a possible product of K.

$$\mathsf{mult}(q_{nonly1s'}(m), q_{nt}(m')) \overset{m \equiv m'}{\rightarrow} q_{exp}(m)$$

In this case it is possible to apply this last transition.

$$\mathsf{exp}(q_\alpha(m), q_{exp}(m')) \rightarrow q_{acc}(h(m, m'))$$

The following lemma states that our automaton recognizes in fact a slight over-appoximation of $W(\rho(K))$ as it recognizes also some terms that are not witnesses (but that are still in $\rho(K)$).

Lemma 7. $W(\rho(K)) \subseteq L(\mathcal{A}_K) \subseteq \rho(K)$.

Lemma 8. *$(L(\mathcal{A}_{E_1}) \cup L(\mathcal{A}_{E_2}), L(\mathcal{A}_K))$ is well-formed.*

Proof. As no transitions has a left hand side headed by an xor, constraints (1) and (3) of Definition 4 are satisfied. We can check on the construction of the automata $\mathcal{A}_{E_1}$ and $\mathcal{A}_{E_2}$ (given in Appendix C) that every term t accepted by these automata is of the form $\mathsf{exp}(u, v)$ for some u and v. By Definition 3, this implies that $C(L(\mathcal{A}_{E_1}) \cup L(\mathcal{A}_{E_2})) = \emptyset$.

7 Conclusion

We have shown that for a class of well-formed protocols, a general model of intruder capabilities including applications of modular exponentiation and exclusive or is equivalent to a weaker model which can be seen as the classical Dolev-Yao model modulo associativity and commutativity of some operators. We have then shown, by a series of reductions and over-approximations, that the secrecy problem for group protocols in presence of a passive attacker can be shown by using advanced tree automata techniques. We have shown how to check (over-approximations of) conditions on the indexes of constants appearing in a term by a $\mathrm{V}\textsc{tam}^{\equiv}_{\not\equiv}$ automaton, how to cope with congruence classes modulo associativity and commutativity in this automata model, and finally that recognizability by this class of automata is preserved by construction of the Dolev-Yao closure.

While our approach applies to several examples of group protocols there is still room for improvements. The first possible generalization concerns our definition of well-formation of a group protocol. Some of the clauses of our definition seem to be rather natural, whereas some others are more arbitrary. A possible continuation of this work is to relax or to modify some of these restrictions, keeping in mind that it must still be possible to prove a reduction result to the classical Dolev-Yao intruder model.

Another restriction of our approach consists in the hypothesis that the only possible exponents are products of constants. This is not the case in general. Group protocols involving exponents different from a simple product exist. An exponent could be represented by a sum, or by an exponentiation itself. The theory of modular exponentiation seems to remain hard to manage in its full generality.

An important avenue of future research is the automatisation of the construction of the automaton recognizing the set of emitted messages, resp. of supposed secrets. This includes the definition of a specification language proper to group protocols.

References

1. Collin Boyd and Juan-Manuel González Nieto. Round-optimal ciontributory conference key agreement. In *Proceedings of the 6th International Workshop on Theory and Practice in Public Key Cryptography (PKC'03)*, volume 2567 of *LNCS*, pages 161–174. Springer, 2003.
2. Emmanuel Bresson, Olivier Chevassut, Abdelilah Essiari, and David Pointcheval. Mutual authentication and group key agreement for low-power mobile devices. *Computer Communications*, 27(17):1730–1737, 2004.
3. Hubert Comon-Lundh, Florent Jacquemard, and Nicolas Perrin. Tree automata with memory, visibility and structural constraints. In *Proceedings of the 10th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'07)*, volume 4423 of *LNCS*, pages 168–182. Springer, 2007.

4. Hubert Comon-Lundh, Florent Jacquemard, and Nicolas Perrin. Visibly tree automata with memory and constraints. Research Report LSV-07-30, Laboratoire Spécification et Vérification, ENS Cachan, France, September 2007. To appear in Logical Methods in Computer Science.
5. Hubert Comon-Lundh and Vitaly Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *Proceedings of the 18th IEEE Symposium on Logic in Computer Science (LICS'03)*, volume 171, pages 271–280. IEEE Computer Society Press, 2003.
6. Evelyne Contejean, Claude Marché, Benjamin Monate, and Xavier Urbain. *The CiME Rewrite Tool*, 2000. http://cime.lri.fr.
7. Nachum Dershowitz. Termination of rewriting. *J. Symb. Comput.*, 3(1-2):69–116, 1987.
8. Danny Dolev and Andrew C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
9. Jean Goubault-Larrecq. A method for automatic cryptographic protocol verification (extended abstract). In *Parallel and Distributed Processing Symposium (IPDPS '00)*, volume 1800 of *LNCS*, pages 977–984. Springer, 2000.
10. Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. In *Advances in Cryptology – CRYPTO'03*, volume 2729 of *LNCS*, pages 110–125. Springer, 2003.
11. Ralf Küsters and Tomasz Truderung. On the Automatic Analysis of Recursive Security Protocols with XOR. In *Proceedings of the 24th Symposium on Theoretical Aspects of Computer Science (STACS 2007)*, volume 4393 of *LNCS*. Springer, 2007.
12. Olivier Pereira and Jean-Jacques Quisquater. Some attacks upon authenticated group key agreement protocols. *Journal of Computer Security*, 11(4):555–580, 2003.
13. Olivier Pereira and Jean-Jacques Quisquater. On the impossibility of building secure cliques-type authenticated group key agreement protocols. *Journal of Computer Security*, 14(2):197–246, 2006.
14. Graham Steel and Alan Bundy. Attacking group protocols by refuting incorrect inductive conjectures. *Journal of Automated Reasoning*, 36(1-2):149–176, 2006.
15. Michael Steiner, Gene Tsudik, and Michael Waidner. Diffie-Hellman key distribution extended to group communication. In *ACM Conference on Computer and Communications Security*, pages 31–37, 1996.
16. Tomasz Truderung. Selecting theories and recursive protocols. In *Proceedings of the 16th International Conference on Concurrency Theory (CONCUR'05)*, volume 3653 of *LNCS*, pages 217–232. Springer, 2005.

A Proofs of Section 4

Lemma 1. *Let E be a set of terms. If $E \vdash_I t$ then $E \vdash_{I'} t$, and if $E \vdash_{I \setminus Gxor} t$ then $E \vdash_{I' \setminus Gxor} t$.*

Proof. The proof is the same for both deduction systems so we only present the proof for the system I. Let π be a deduction of $E \vdash_I t$.

We iteratively replace any part of π of the form

$$\frac{S \vdash \langle u, v\rangle \quad S \vdash c}{S \vdash \langle u', v'\rangle}exp$$

by

$$\dfrac{\dfrac{\dfrac{E \vdash \langle u, v\rangle}{E \vdash u}proj_1 \qquad E \vdash c}{E \vdash u^c}exp \qquad \dfrac{\dfrac{E \vdash \langle u, v\rangle}{E \vdash v}proj_2 \qquad E \vdash c}{E \vdash v^c}exp}{E \vdash \langle u^c, v^c\rangle \downarrow_{R/AC}}pair$$

It is easy to see that this transformation terminates and yields a deduction of $\langle u', v'\rangle$. Moreover, the obtained deduction of $\langle u^c, v^c\rangle \downarrow_{R/AC}$ is such that u and v are not headed by the pair symbol. Otherwise the transformation would have been applied again. Hence, the normalization step in R/AC coincides with the normalization step in R'/AC and we have a deduction in I'.

Lemma 2. *Let $c \in \Sigma_0$ and t be a term such that $c \in C(t)$. If $E \vdash_{I'} t$ then $c \in C(E)$.*

Proof. Let π be a deduction of $E \vdash_{I'} t$. We prove the result by induction on the size of the deduction. Let r be the last rule of π.

Base case. $|\pi| = 1$. Hence, $r = axiom$. As $t \in E$ and $c \in C(t)$ we have that $c \in C(E)$.

Inductive case. We consider all possibilities for r.

- $r = proj_1$. It must be that the root of π has a unique successor, labelled by $E \vdash \langle t, u\rangle$ for some u. As $c \in C(t)$, by Definition 3, $c \in C(\langle t, u\rangle)$. We apply the induction hypothesis on the deduction π' of $E \vdash \langle t, u\rangle$ and conclude that $c \in C(E)$.
- $r = proj_2$ or $r = dec$. These cases are similar to the case where $r = proj_1$.
- $r = pair$. It must be that $t = \langle u, v\rangle$ for some u and v. By Definition 3 $C(\langle u, v\rangle) = C(u) \cup C(v)$. As $c \in C(\langle u, v\rangle)$ we have that $c \in C(u)$ or $c \in C(v)$. Let π_1 and π_2 be the deductions of $E \vdash u$ and $E \vdash v$. By induction hypothesis on either π_1 or π_2 we conclude that $c \in C(E)$.

- $r = enc$. This case is similar to the case where $r = pair$.
- $r = hash$. It must be that $t = H(u)$. By Definition 3, $C(t) = \emptyset$ which contradicts that $c \in C(t)$. So $r \neq hash$.
- $r = exp$. $t = u^v$ for some u and v. By definition 3, $C(t) = \emptyset$. We get a contradiction with the hypothesis that $c \in C(t)$. So $r \neq exp$.
- $r = Gxor$. By Lemma 10, one of the premises of r is either $E \vdash_I t$ or $E \vdash_{I'} t \oplus u_{1'} \oplus \cdots \oplus u_n$, for some u_i. In the first case we immediately conclude by induction hypothesis. Let us consider the case where $E \vdash_{I'} t \oplus u_1 \oplus \cdots \oplus u_n$. By definition 3, $C(t \oplus u_1 \oplus \cdots \oplus u_n) = C(t) \cup \bigcup_{i=1} C(u_i)$. So $C(t) \subseteq C(t \oplus u_1 \oplus \cdots \oplus u_n)$ and $c \in C(t \oplus u_1 \oplus \cdots \oplus u_n)$. By induction hypothesis we have that $c \in C(E)$.

Lemma 9. *Let $t = u_1 \oplus \cdots \oplus u_n$ be a term in normal form such that for every i we have that u_i is not headed by $\oplus$. For every $v_1, \ldots, v_m$ in which $\oplus$ does not occur, if $v_1 \oplus \ldots \oplus v_m \rightarrow^* t$ then for all i there exists j such that $v_j \rightarrow^* u_i$.*

Proof. By induction on the number l of rewriting steps.

Base case. $l = 0$. $u_1 \oplus \cdots \oplus u_n = v_1 \oplus \cdots \oplus v_m$. So we trivially have that for all i there exists j such that $v_j \rightarrow^* u_i$.

Inductive case. $l = i$. Let $v'_1, \ldots, v'_{n'}$ be terms not headed by $\oplus$, such that $v'_1 \oplus \cdots \oplus v'_{n'} \rightarrow^* t$ in i steps. By induction hypothesis for all i there exists j such that $v'_j \rightarrow u_i$. Let us show that for any term $v_1, \ldots, v_m$ such that $v_1 \oplus \ldots \oplus v_m \rightarrow v'_1 \oplus \ldots \oplus v'_{n'}$, for all i there exists j such that $v'_j \rightarrow u_i$. The result is straightforward by considering the different possibilities of application of rewrite rules.

Lemma 10. *For every instance $\frac{S \vdash v_1 \cdots S \vdash v_n}{S \vdash u_1 \oplus \ldots \oplus u_m}$ of the Gxor rule such that u_i is not headed with $\oplus$ and $u_i \neq 0$, there is a j such that $v_j = u_i$ or $v_j = u_i \oplus u'$ for some term u'.*

Proof. The result follows directly from Lemma 9.

Lemma 3. *Let E be a set of terms. If π is a minimal deduction in I' of one of the following forms :*

$$\dfrac{\dfrac{\vdots}{E \vdash \langle u, v \rangle}}{E \vdash u}\, proj_1 \qquad \dfrac{\dfrac{\vdots}{E \vdash \langle u, v \rangle}}{E \vdash v}\, proj_2 \qquad \dfrac{\dfrac{\vdots}{E \vdash \{u\}_v} \quad \dfrac{\vdots}{E \vdash v}}{E \vdash u}\, dec$$

Then $\langle u, v \rangle \in St(E)$ (resp. $\langle u, v \rangle \in St(E)$, resp. $\{u\}_v \in St(E)$).

Proof. Assume that the last rule is $proj_1$. The other cases are similar. We claim that for every $n \in \mathbb{N}^*$,

– either in π there exists a branch, i.e., a sequence of nodes of length n, starting from $E \vdash \langle u, v\rangle$ such that for every node $E \vdash t$ on this branch we have, $\langle u, v\rangle \in St(t)$.
– or there exists such a branch of length $l \leq n$ which is maximal, i.e. whose last node is obtained by the rule *axiom*.

As π is finite there exists always an integer m such that if the claim is true, we are in the second case, and so we conclude that $\langle u, v\rangle \in St(E)$.

Let us prove the claim by induction on n.

Base case. If $n = 1$, we immediate conclude because $\langle u, v\rangle \in St(\langle u, v\rangle)$.

Inductive case. We suppose that the claim is true for n and show that it also holds for $n+1$. By induction, either there is a maximal branch of length $l \leq n$ such that for every node $E \vdash t$ on this branch we have that $\langle u, v\rangle \in St(t)$. Then this is also the case for $n+1$. Or we have that there is a sequence of rules of length n such that for every node $E \vdash t$ we have that $\langle u, v\rangle \in St(t)$. Let $E \vdash t_f$ be the last node of this sequence. We consider all the possible rules r' introducing $E \vdash t_f$. If r' is an axiom, the sequence of length n $(\leq n+1)$ was a maximal branch. Else we have to show that there is a premise $E \vdash t'_f$ of r' such that $\langle u, v\rangle \in St(t'_f)$. If r' is a deconstruction rule, as by induction $\langle u, v\rangle \in St(t_f)$, $\langle u, v\rangle$ is trivially a subterm of one of the premises. Else we are in one of the two following cases.

– $t_f = \langle u, v\rangle$. r' cannot be a *pair* by minimality of π. r' cannot be an *exp*, an *enc* or a *hash* because of the form of t_f. So it is a deconstruction rule.
– $t_f \neq \langle u, v\rangle$. By induction hypothesis we have that $\langle u, v\rangle \in St(t_f)$.
  - If r' is a *pair* then $t_f = \langle u', v'\rangle$, and either $\langle u, v\rangle \in St(u')$ or $\langle u, v\rangle \in St(v')$. As $E \vdash u'$ and $E \vdash v'$ are the premises of r' we can conclude. We do a similar reasoning if $r' = enc$ or $r' = hash$.
  - If r' is a $Gxor$, if $t_f = u_1 \oplus \cdots \oplus u_n$ then there is an i such that $\langle u, v\rangle \in St(u_i)$. By Lemma 10, there is a premise $E \vdash t'_f$ of r' such that $\langle u, v\rangle \in St(t'_f)$. If $t_f \neq u_1 \oplus \cdots \oplus u_n$ we straightforwardly conclude by Lemma 10.
  - If r' is an *exp*, as the deduction belongs to the system I' $t_f = u'^{c_1 \cdot \cdots \cdot c_n}$ for some u'. If $\langle u, v\rangle \in St(t_f)$, then $\langle u, v\rangle \in St(u')$.

Lemma 4. *Let E be a set of terms and t a term. If for every $u^{c_1 \cdot \cdots \cdot c_n} \in St(E, t)$ such that $c_i \in \Sigma_0$ one has $c_i \notin C(E)$ and if $E \vdash_{I' \setminus Gxor} t$ then $E \vdash_{DY} t$.*

Proof. Let π be a deduction of $E \vdash_{I' \setminus Gxor} t$. By induction on the size $|\pi|$ of π. Let r be the last rule of π.

Base case. $|\pi| = 1$. We must have that $r = axiom$ and hence π is a deduction in DY.

Inductive case. $|\pi| = n$. We consider the different possibilities for the rule r.

- $r = pair$. We have that $t = \langle u, v\rangle$ for some u and v. There exist two deductions π_1 of $E \vdash_{I'} u$ and π_2 of $E \vdash_{I'} v$ such that $|\pi_1|, |\pi_2| \leq n$. As $u, v \in St(t)$, we have that for every $w^{c_1 \cdot \dots \cdot c_n} \in St(E, u)$, $c_i \notin C(E)$, and for every $w^{c_1 \cdot \dots \cdot c_n} \in St(E, v)$, $c_i \notin C(E)$. So we can apply the induction hypothesis and we obtain that $E \vdash_{DY} u$ and $E \vdash_{DY} v$. Hence we can build a deduction π' of $E \vdash t$ in DY.
- $r = hash$ or $r = enc$. These cases are similar to $r = pair$.
- $r = proj_1$. Let $E \vdash \langle t, u\rangle$ be the premise of r. By hypothesis we have that for every $u^{c_1 \cdot \dots \cdot c_n} \in St(E, t)$, $c_i \notin C(E)$. By Lemma 3 $\langle t, u\rangle \in St(E)$. It follows that for every $u^{c_1 \cdot \dots \cdot c_n} \in St(E, \langle t, u\rangle)$, $c_i \notin C(E)$. We also have a deduction of $E \vdash \langle t, u\rangle$ in I'. Hence we can apply our induction hypothesis and conclude that we have a deduction of $E \vdash \langle t, u\rangle$ in DY. We conclude that we have a deduction of $E \vdash t$, in DY.
- $r = proj_2$ or $r = dec$. These cases are analogous to $r = proj_1$.
- $r = exp$. We have that $t = v^{c_1 \cdot \dots \cdot c_n}$. Moreover, one of the premises of exp is $c \in \{c_1, \cdots, c_n\}$. On the one hand, by hypothesis, $c \notin C(E)$. On the other hand, we have a deduction π_1 of $E \vdash_{I'} c$. By Corollary 1, we have that $c \in C(E)$. So we get a contradiction, and we conclude that $r \neq exp$.

Lemma 11. *Let E be a set of terms satisfying constraint 1 of Definition 4 and π be a minimal deduction of $E \vdash_{I'} u \oplus v$. π is of one of the following forms :*

$$\frac{}{E \vdash u \oplus v}\, axiom \qquad\qquad \frac{\begin{array}{c}\pi_1\\ \vdots\\ E \vdash t_1\end{array} \quad \dots \quad \begin{array}{c}\pi_n\\ \vdots\\ E \vdash t_n\end{array}}{E \vdash u \oplus v}\, Gxor$$

Proof. Let r be the last rule of π. We consider all possibilities for the rule r

- r is a construction rule (*pair*, *enc*, *hash*, *exp*). This is excluded because of the form of $u \oplus v$.
- r is a deconstruction rule ($proj_1$, $proj_2$, *dec*). Assume the last rule is $proj_1$. The other cases are similar. As π is minimal and ends by $proj_1$, by Lemma 3, for some t, $\langle u \oplus v, t\rangle \in St(E)$. Moreover E satisfies constraint 1 of Definition 4. It follows that for every term $t' \in E$ such that $\oplus$ occurs in t', $t' = u' \oplus v'$ for some u' and v' and $\oplus$ does not occur neither in u' nor in v'. Hence we have a contradiction with $\langle u \oplus v, t\rangle \in St(E)$. So r is not a deconstruction rule.

The only remaining possibilities for r are *axiom* and *Gxor*.

Lemma 12. *Let E be a set of terms satisfying constraint 1 of Definition 4, and let π be a minimal deduction in I' of the following form :*

$$\frac{\dfrac{\begin{array}{c}\pi_1\\ \vdots\end{array}}{E \vdash t_1}\, r_1 \quad \dots \quad \dfrac{\begin{array}{c}\pi_n\\ \vdots\end{array}}{E \vdash t_n}\, r_n}{E \vdash t_1 \oplus \dots \oplus t_n \downarrow_{R/AC}}\, Gxor$$

such that π_i is a deduction of $E \vdash_{I' \setminus Gxor} t_i$. Then for every t_i either $t_i = u \oplus v$ or $t_i = u$ for some u and v not headed by $\oplus$.

Proof. By minimality of π, for every i, $r_i \neq Gxor$. By Lemma 11, if $t_i = u \oplus v$, $E \vdash t_i$ is introduced by an *axiom*. Hence, $t_i \in E$. As E satisfies constraint 1 of Definition 4, if $t_i = u \oplus v$, $\oplus$ does not occur in u nor in v.

Lemma 13. *Let $u_1, \ldots, u_n$ and t be terms in normal form such that $t \neq 0$ and $u_i, t \neq u \oplus v$. If $u_1 \oplus \ldots \oplus u_n \rightarrow^* t$ in R', then there exists an odd $m \leq n$ such that there are exactly m u_i such that $u_i \neq 0$.*

Proof. By induction on the number l of rewriting steps.

Base case. $l = 0$. $u_1 \oplus \ldots \oplus u_n = t$, as $t \neq u \oplus v$, $n = 1$ so n is odd and $u_1 = t \neq 0$.

Inductive case. $l = i + 1$. We have that $u_1 \oplus \ldots \oplus u_n \rightarrow^{i+1} t$. Let $v_1, \ldots, v_{n'}$ be such that $u_1 \oplus \ldots \oplus u_n \rightarrow v_1 \oplus \ldots \oplus v_{n'} \rightarrow^i t$. We have to consider the two following possibilitites of applications of a rewrting rule for $u_1 \oplus \ldots \oplus u_n \rightarrow v_1 \oplus \ldots \oplus v_{n'}$.

- Either the used rewrite rule is $x \oplus 0 \rightarrow x$. Then, each of the v_i is in normal form and not headed by $\oplus$. By induction hypothesis there exists an odd $m' \leq n'$ such that there are exactly m' v_i such that $v_i \neq 0$. Let $m = m'$. We have that m is odd, $m \leq n$ and there are exactly $m \leq n$ u_i such that $u_i \neq 0$, which correspond to the m' v_j.
- Or the used rewrite rule is $x \oplus x \rightarrow 0$ and $x \neq 0$ (if $x = 0$ we suppose that we are in the first case). Again, each of the v_i is in normal form and not headed by $\oplus$. By induction hypothesis there exists an odd $m' \leq n'$ such that there are exactly m' v_i such that $v_i \neq 0$. Let $m = m' + 2$. As, at least one of the v_i is 0, we have that $m' < n'$. As moreover, $n = n' + 1$ we have that $m \leq n$, m is odd and there are exactly m u_i such that $u_i \neq 0$, which correspond to the m' v_j plus two u_i which are equal to the value of x in the application of the rewrite rule.

Lemma 14. *Let E be a set of terms satisfying constraints 1 and 2 of Definition 4, and let π be a minimal deduction in I' of the following form :*

$$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ E \vdash t_1\end{array} \quad \ldots \quad \begin{array}{c}\pi_n\\ \vdots\\ E \vdash t_n\end{array}}{E \vdash t_1 \oplus \ldots \oplus t_n \downarrow_{R/AC}}\ Gxor$$

such that every π_i is a deduction of $E \vdash t_i$ in $I' \setminus Gxor$, then $t_1 \oplus \ldots \oplus t_n \downarrow_{R/AC} = u_1 \oplus \ldots \oplus u_m$ for some terms u_i, with $m \geq 2$.

Proof. By contradiction. Suppose that $t = t_1 \oplus \ldots \oplus t_n \downarrow_{R/AC}$ is not headed by $\oplus$. As π is minimal and $0 \in E$, $t \neq 0$. By Lemma 10 there is an i such that either

$t_i = t$ or $t_i = t \oplus u$ for some term u. By minimality of π, there is no t_i such that $t_i = t$. Hence, there is a t_i such that $t_i = t \oplus u$.

Let $t_1 \oplus \ldots \oplus t_n$ be equal to $v_1 \oplus \ldots \oplus v_m$ such that $v_i \neq u \oplus v$. By minimality of π, every $t_i \neq 0$. As the t_i are in normal form by definition of I', $t_i \neq u \oplus 0$. It follows that every $v_i \neq 0$. As $t \neq 0$ and $t \neq u_1 \oplus \ldots \oplus u_m$ for any $m \geq 2$ and any u_i, and as $t_1 \oplus \ldots \oplus t_n \rightarrow^* t$, by Lemma 13, m is odd.

By Lemma 12, this implies that there is an i such that t_i is not headed by $\oplus$. By minimality it also implies that there exists j such that $t_j = t_i \oplus v$ for some v.

By Lemma 11, $t_i \oplus v \in E$, and so $t_i \in St(E)$. As E satisfies constraint 2, we have that for every $u^{c_1,\ldots,c_n} \in St(E, t_i)$ such that $c_i \in \Sigma_0$, $c_i \notin C(E)$. As π_i is a deduction of $E \vdash_{I' \setminus Gxor} t_i$, by Lemma 4, $E \vdash_{DY} u$. As $t_i \oplus v \in E$ we have a contradiction with the fact that E satisfies constraint 1 of Definition 4.

Lemma 15. *Let E be a set satisfying constraints 1 and 2 of Definition 4. Let π be a minimal deduction of $E \vdash_{I'} t$. Let π' be the subtree of π constructed as follows: starting from the root $E \vdash_{I'} t$ follow each branch until reaching either a leaf or a node obtained by the application of a Gxor rule. In the second case, cut the branch such that the leaf of this branch is the node corresponding to the conclusion of the Gxor rule. Then if there exists a leaf of π', which is labeled $E \vdash_{I'} u \oplus v$, $u \oplus v \in St(t)$*

Proof. By induction on the size $|\pi'|$ of π'.

Base case. If $|\pi'| = 1$, the only leaf is $E \vdash_{I'} t$. So if $t = u \oplus v$, trivially $u \oplus v \in St(t)$.

Inductive case. $|\pi'| = n$. Let r be the last rule of π'. We consider the different possibilities for the last rule r.

- $r = pair$. $t = \langle t_1, t_2 \rangle$ for some t_1 and t_2 and $E \vdash_{I'} t_1$ and $E \vdash_{I'} t_2$ are the premises of r. Let π'_1 and π'_2 be the sub-trees identically defined as π' but whose roots are $E \vdash_{I'} t_1$ and $E \vdash_{I'} t_2$. If there is a leaf of π', which is labelled $E \vdash_{I'} u \oplus v$, then it is also the case for π'_1 or π'_2. As $|\pi'_1| < n$ and $|\pi'_2| < n$, we can apply the induction hypothesis. We have that $u \oplus v \in St(t_1)$ or $u \oplus v \in St(t_2)$. As $t = \langle t_1, t_2 \rangle$, $u \oplus v \in St(t)$.
- $r = enc$, $r = exp$, $r = hash$. These cases are similar to the case $r = pair$.
- $r = proj_1$. We show that it would imply a contradiction with the fact that E is a set satisfying constraints 1 and 2 of Definition 4. Let $E \vdash_{I'} \langle t, u \rangle$ be the premise of r for some u. Let π'^- be the sub-tree identically defined as π' but whose root is $E \vdash_{I'} \langle t, u \rangle$. If there is a leaf of π', which is labeled $E \vdash_{I'} u \oplus v$, then it is also the case for π'^-. As $|\pi'^-| < n$, we can apply the induction hypothesis. We have that $u \oplus v \in St(\langle t, u \rangle)$. Moreover, as π is a minimal deduction of $E \vdash_{I'} t$, by Lemma 3 $\langle t, u \rangle \in St(E)$. As $u \oplus v \in St(\langle t, u \rangle)$ and $u \oplus v \neq \langle t, u \rangle$, this is a contradiction with constraint 1 of Definition 4. Hence $r \neq proj_1$.
- $r = proj_2$, $r = dec$. These cases are similar to $r = proj_1$.

– $r \neq Gxor$ by the construction of π'.

Lemma 5. *Let E be a set of terms satisfying constraints 1 and 2 of Definition 4. Let π be a minimal deduction of $E \vdash_{I'} t$. If π involves an application of the Gxor rule, then $\oplus$ occurs in t.*

Proof. By induction on the number $|\pi|_{Gxor}$ of applications of the $Gxor$ rule in π.

Base case. $|\pi|_{Gxor} = 0$. π does not involve any instance of $Gxor$ rule. Hence the result trivially holds.

Inductive case. $|\pi|_{Gxor} = n$ We distinguish two cases.

– Every instance of a $Gxor$ rule appears on a different branch of π. Let π' be the maximal sub-tree of π not containing any instance of the $Gxor$ rule whose root is $E \vdash_{I'} t$, defined as in Lemma15. Let $E \vdash_{I'} t_f$ be a leaf of π' introduced in π by an application of the $Gxor$ rule r. As the deductions of the premises of r do not involve any application of $Gxor$ and as π is minimal, by Lemma 14, $t_f = u \oplus v$ for some u and v. By Lemma 15, $u \oplus v \in St(t)$.
– There exists a branch involving at least 2 $Gxor$ rules. Let us consider such a branch. Let r be the lowest, i.e., closest to the root, instance of the $Gxor$ rule on this branch. Let $E \vdash_{I'} t_f$ be the node introduced by r.
  - If $t_f = u \oplus v$ for some u and v, we do a similar reasoning as in the case where every instance of the $Gxor$ rule is on a different branch of π.
  - If $t_f \neq u \oplus v$ for any u and v then we show a contradiction with constraint 1 of Definition 4. Suppose that $t_f \neq u \oplus v$ for any u and v. r has the following form :

$$\dfrac{\begin{array}{c}\pi_1\\ \vdots\\ E \vdash t_1\end{array} \quad \ldots \quad \begin{array}{c}\pi_n\\ \vdots\\ E \vdash t_n\end{array}}{E \vdash t_f}\ Gxor$$

    As π is minimal and $0 \in E$, $t \neq 0$. By Lemma 10 there exists i such that either $t_i = t_f$ or $t_i = t_f \oplus u$ for some term u. By minimality of π, there is no t_i such that $t_i = t_f$. Hence, there is a t_i such that $t_i = t_f \oplus u$.
    Let $t_1 \oplus \ldots \oplus t_n$ be equal to $v_1 \oplus \ldots \oplus v_m$ such that $v_i \neq u \oplus v$. By minimality of π, every $t_i \neq 0$. As the t_i are in normal form by definition of I', $t_i \neq u \oplus 0$. It follows that every $v_i \neq 0$. As $t_f \neq 0$ and $t_f \neq u_1 \oplus \ldots \oplus u_m$ for some $m \geq 2$ and some u_i, and as $t_1 \oplus \ldots \oplus t_n \rightarrow^* t_f$, by Lemma 13, m is odd.
    By Lemma 12, this implies that there exists i such that t_i is not headed by $\oplus$. By minimality it also implies that there exists j such that $t_j = t_i \oplus v$ for some v. For any i if $t_i = u \oplus v$ for some u and v then no application of the $Gxor$ occurs in π_i as by Lemma 11 and minimality of π, $\pi_i = \frac{}{E \vdash t_i}$.

So the application of the $Gxor$ rule appears in a deduction π_i of $E \vdash_{I'} t_i$ such that t_i is not headed by $\oplus$. As $|\pi_i|_{Gxor} < n$, by induction hypothesis, $\oplus$ occurs in t_i. As $t_j = t_i \oplus v$ has been introduced by an axiom, we have a contradiction with the constraint 1 of the Definition 4.

B Proofs of Section 5

Lemma 16. *$\rho(DY(S)) = DY'(\rho(S))$ for ρ defined as in Example 2.*

Proof. Let us first prove that $\rho(DY(S)) \subseteq DY'(\rho(S))$. Let t' be in $\rho(DY(S))$. By definition there exists $t \in DY(S))$ such that $t' = \rho(t)$. By Definition 2 $S \vdash_{DY} t$. We claim that if $S \vdash_{DY} t$, then $\rho(S) \vdash_{DY'} \rho(t)$, so $\rho(S) \vdash_{DY'} t'$, and finally $t' \in DY'(\rho(S))$.

Let us show that if there exists a deduction π of $S \vdash_{DY} t$, then there also exists a deduction π' of $\rho(S) \vdash_{DY'} \rho(t)$. By induction on the size of π.

Base case. $|\pi| = 0$. Then the deduction just consists of one aplication of *axioms*, and hence $t \in S$ and $\rho(t) \in \rho(S)$ and straightforwardly $\rho(S) \vdash_{DY'} \rho(t)$.

Inductive case. $|\pi| = n$. We consider the different cases for the last rule r.

- $r = pair$. It must be that $t = \mathsf{pair}(u, v)$ for some u and v. As π is a deduction of $S \vdash_{DY} \mathsf{pair}(u, v)$, there exists two deductions π_1 of $S \vdash_{DY} u$ and π_2 of $S \vdash_{DY} v$ such that $|\pi_1|, |\pi_2| \leq n$. By induction there exists π_1' a deduction of $\rho(S) \vdash_{DY'} \rho(u)$ and π_2' a deduction of $\rho(S) \vdash_{DY'} \rho(v)$. By an application of the *pair* rule we obtain a deduction π' of $\rho(S) \vdash_{DY'} \mathsf{pair}(\rho(u), \rho(v))$. By Definition, $\rho(\mathsf{pair}(u, v)) = \mathsf{pair}(\rho(u), \rho(v))$, and so $\rho(S) \vdash_{DY'} \rho(\mathsf{pair}(u, v))$.
- $r = proj_1$, $r = proj_2$, $r = dec$, $r = enc$. These cases are similar to $r = pair$.
- $r = hash$. It must be that $t = H(u)$ for some u. As π is a deduction of $S \vdash_{DY} H(u)$, there exists a deduction π^- of $S \vdash_{DY} u$ such that $|\pi^-| \leq n$. By induction there exists $\pi^{-'}$ a deduction of $\rho(S) \vdash_{DY'} \rho(u)$. By an application of the $hash'$ rule we obtain a deduction π' of $\rho(S) \vdash_{DY'} H(0, \rho(u))$. By Definition, $\rho(H(u)) = H(0, \rho(u))$, and so $\rho(S) \vdash_{DY'} \rho(H(u))$.

Let us now prove that $DY'(\rho(S)) \subseteq \rho(DY(S))$. Let t' be a term in $DY'(\rho(S))$. By Definition, there exists a deduction π' of $\rho(S) \vdash_{DY'} t'$. We claim that if $\rho(S) \vdash_{DY'} t'$ then there exists a $t \in DY(S)$ such that $t' = \rho(t)$. Hence $t' \in \rho(DY(S))$.

Let us show that if there exists a deduction π' of $\rho(S) \vdash_{DY'} t'$ then there exists a $t \in DY(S)$ such that $t' = \rho(t)$. By induction on the size of π. Let r be the last rule of π'.

Base case. $|\pi| = 0$. We must have that $r = axiom$ and hence $t' \in \rho(S)$. So there exists $t \in S$ such that $t' = \rho(t)$ and trivially $t \in DY(S)$.

Inductive case. $|\pi| = n$. We consider the different cases for the last rule r.

- $r = pair$. It must be that $t' = \mathsf{pair}(u', v')$ for some u' and v'. As π' is a deduction of $\rho(S) \vdash_{DY'} \mathsf{pair}(u', v')$, there exists two deductions π_1' of $\rho(S) \vdash_{DY'} u'$ and π_2' of $\rho(S) \vdash_{DY'} v'$ such that $|\pi_1'|, |\pi_2'| \leq n$. By induction hypothesis there exists $u \in DY(S)$ such that $u' = \rho(u)$ and $v \in DY(S)$ such that $v' = \rho(v)$. By an application of *pair* we have a deduction π of $S \vdash_{DY} \mathsf{pair}(u, v)$, so $\mathsf{pair}(u, v) \in DY(S)$. By Definition, $\rho(\mathsf{pair}(u, v)) = \mathsf{pair}(\rho(u), \rho(v))$ and hence $\rho(\mathsf{pair}(u, v)) = \mathsf{pair}(u', v')$.

- $r = proj_1$, $r = proj_2$, $r = dec$, $r = enc$. These cases are similar to $r = pair$, $r = hash$.

Lemma 17. *Let S_1 and S_2 be sets of terms closed under associativity and commutativity of* xor *and* mult*. If for every t, $t \in S_1 \cap S_2$ if and only if $W(t) \in W(S_1) \cap W(S_2)$.*

Proof. Let us show that if $t \in S_1 \cap S_2$ then $W(t) \in W(S_1) \cap W(S_2)$. Let t be a term of $S_1 \cap S_2$, then $t \in S_1$ and $t \in S_2$. By the extension of Definition 7, $W(t) \in W(S_1)$ and $W(t) \in W(S_2)$, so $W(t) \in W(S_1) \cap W(S_2)$.

Let us show that if $W(t) \in W(S_1) \cap W(S_2)$ then $t \in S_1 \cap S_2$. Let $W(t) \in W(S_1) \cap W(S_2)$, we hence have $W(t) \in W(S_1)$, that is there is an $t_1 \in S_1$ with $W(t) = W(t_1)$. By definition of W this implies that $t \equiv_{AC} t_1$, and hence that $t \in S_1$ since S_1 is closed under AC. The argument for $t \in S_2$ is analogous.

Lemma 18. *For any symbol f of $\{\mathsf{pair}, \mathsf{enc}, H\}$ and for any term u and v, we have $W(f(u, v)) = f(W(u), W(v))$.*

Proof. First, as f is neither xor nor mult, $W(f(u, v)) = f(u', v')$ for some u' and v', and $u \equiv_{AC} u'$ and $v \equiv_{AC} v'$. By minimality of $W(f(u, v))$ in the congruence class of $f(u, v)$, u' is minimal in the congruence class of u and v' is minimal in the congruence class of v. It follows that $u' = W(u)$ and $v' = W(v)$.

Theorem 5. *If S is closed under AC then $W(DY'(S)) = DY'(W(S))$.*

Proof. Let t' be a term such that $t' \in W(DY'(S))$. There exists $t\ inDY'(S)$ such that $t' = W(t)$. We claim that if $S \vdash_{DY'} t$, then $W(S) \vdash_{DY'} W(t)$. Hence $W(S) \vdash_{DY'} W(t)$, and so $t' \in DY'(W(S))$.

Let us show by induction on $|\pi|$ that if there $S \vdash_{DY'} t$, then there exists a deduction π' of $W(S) \vdash_{DY'} W(t)$.

Base case. $|\pi| = 0$. Then the deduction π consists just of one application of the rule *axiom*. Hence, $t \in S$, and by consequence $W(t) \in W(S)$ and $W(S) \vdash_{DY'} W(t)$.

Inductive case. $|\pi| = n$. We consider the different possibilities for the last rule r of the deduction π.

- $r = pair$. It must be that $t = \mathsf{pair}(u, v)$ for some u and v. As π is a deduction of $S \vdash_{DY'} \mathsf{pair}(u, v)$, there exists two deductions π_1 of $S \vdash_{DY'} u$ and π_2 of $S \vdash_{DY'} v$ such that $|\pi_1|, |\pi_2| \leq n$. By induction hypothesis there exists a deduction π'_1 of $W(S) \vdash_{DY'} W(u)$ and a deduction π'_2 of $W(S) \vdash_{DY'} W(v)$. By an application of the *pair* rule we obtain a deduction π' of $W(S) \vdash_{DY'} \mathsf{pair}(W(u), W(v))$. By Lemma 18 $W(\mathsf{pair}(u, v)) = \mathsf{pair}(W(u), W(v))$, and so $W(S) \vdash_{DY'} W(\mathsf{pair}(u, v))$.
- $r = proj_1$, $r = proj_2$, $r = exp$, $r = enc$, $r = dec$, $r = hash'$. These cases are analogous to $r = pair$.

Let t' be a term $t' \in DY'(W(S))$. We claim that if $W(S) \vdash_{DY'} t'$ then there exists a term t such that $t' = W(t)$ and a deduction π of $S \vdash_{DY'} t$. So $t \in DY'(S)$ and $t' \in W(DY'(S))$.

Let us now show by induction on $|\pi'|$ that if there exists a deduction π' of $W(S) \vdash_{DY'} t'$ then there exists a term t such that $t' = W(t)$ and a deduction π of $S \vdash_{DY'} t$.

Base case. $|\pi'| = 0$. Then the deduction π' consists just of one application of the rule *axiom* and $t' \in W(S)$, that is $t' = W(t)$ for some $t \in S$. Hence, $S \vdash_{DY'} t$.

Inductive case. $|\pi'| = n$. We consider the different possibilities for the last rule r of the deduction π'.

- $r = pair$. It must be that $t' = \mathsf{pair}(u', v')$ for some u' and v'. As π' is a deduction of $W(S) \vdash_{DY'} \mathsf{pair}(u', v')$, there exists two deductions π'_1 of $W(S) \vdash_{DY'} u'$ and π'_2 of $W(S) \vdash_{DY'} v'$ such that $|\pi'_1|, |\pi'_2| \leq n$. By induction hypothesis there exists a term u such that $u' = W(u)$ and a deduction π_1 of $S \vdash_{DY'} u$, and a term v such that $v' = W(v)$ and a deduction π of $S \vdash_{DY'} v$. By an application of the *pair* rule we get a deduction π_2 of $S \vdash_{DY'} \mathsf{pair}(u, v)$. By Lemma 18 $W(\mathsf{pair}(u, v)) = \mathsf{pair}(W(u), W(v))$, so $W(\mathsf{pair}(u, v)) = \mathsf{pair}(u', v')$, and $S \vdash_{DY'} \mathsf{pair}(u, v)$.
- $r = proj_1$, $r = proj_2$, $r = exp$, $r = enc$, $r = dec$, $r = hash'$. These cases are analogous to $r = pair$.

Theorem 6. *For every* $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ $\mathcal{A}$*, such that* $\mathsf{pair}, \mathsf{enc} \notin \{\Sigma'_{\mathrm{INT}^{\equiv}_1} \cup \Sigma'_{\mathrm{INT}^{\equiv}_2}\}$ *and the only constant symbol of* Γ *is* $\perp$*, there exists a* $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ $\mathcal{A}_{DY}$ *such that* $L(\mathcal{A}_{DY}) = DY'(L(\mathcal{A}))$.

Proof. Let $\mathcal{A}$ be a $\mathrm{VTAM}^{\equiv}_{\not\equiv}$. Let p be the partition of $\mathcal{A}$. By Lemma 19 $DY_p(\mathcal{A}) = DY'(L(\mathcal{A}))$.

The restriction on the specification of automata in Theorem 6 do not exclude specifications of the protocols we have represented.

Let us show how to extend the known method of completion of [9] of a classical tree automaton under the DY rules to $\mathrm{VTAM}^{\equiv}_{\not\equiv}$.

To every possible partition p of Σ' except the partition where $\mathsf{pair}, \mathsf{enc} \in \{\Sigma'_{\mathrm{INT}^{\equiv}_1} \cup \Sigma'_{\mathrm{INT}^{\equiv}_2}\}$, we associate a completion system DY_p. $DY_p(\mathcal{A})$ is the fixed point of the application of the rules of DY_p.

Definition 8. *Let* p *be the partition of* Σ' *and* $\mathcal{A} = (\Gamma, Q, Q_f, \Delta)$ *a* $\mathrm{VTAM}^{\equiv}_{\not\equiv}$ *on* p*. The completion system* DY_p *is the set of rules*

$$\{Proj_1, Proj_2, Dec, Pair, Enc, Hash\}$$

defined as follows:

- $Proj_1$.

- If $\mathsf{pair} \in \Sigma'_{\text{PUSH}}$, for every state $q \in Q_f$, if $\mathsf{pair}(q_1(x), q_2(y)) \to q(h(x,y)) \in \Delta$ for some symbol h, we add q_1 to Q_f if $L(\mathcal{A}, q_2) \neq \emptyset$. This last verification is possible by Corollary 5 of [4].
- If $\mathsf{pair} \in \Sigma'_{\text{INT}_1}$ or $\mathsf{pair} \in \Sigma'_{\text{INT}_2}$. These cases are analogous to $\mathsf{pair} \in \Sigma'_{\text{PUSH}}$.
- If $\mathsf{pair} \in \Sigma'_{\text{POP}_{22}}$, for every state $q \in Q_f$, if $\mathsf{pair}(q_1(x), q_2(h(x,y))) \to q(y) \in \Delta$ for some symbol h, let $\mathcal{A}_h$ be the tree automaton recognizing the set of terms on $T(\Gamma)$ headed by h, and let $M(\mathcal{A}_{q_2})$ be the tree automaton recognizing the memory language of q_2 (this automaton can be constructed by Lemma 2 of [4]). If If $M(\mathcal{A}, q_2) \cap \mathcal{A}_h \neq \emptyset$ we add q_1 to Q_f.
  Similarly, if $\mathsf{pair}(q_1(x), q_2(\perp)) \to q(\perp) \in \Delta$, we add q to Q_f in case that $M(\mathcal{A}, q_2) \cap \mathcal{A}_\perp \neq \emptyset$.
- $\mathsf{pair} \in \Sigma'_{\text{POP}_{11}}$, $\mathsf{pair} \in \Sigma'_{\text{POP}_{12}}$, $\mathsf{pair} \in \Sigma'_{\text{POP}_{21}}$. These cases are similar to $\mathsf{pair} \in \Sigma'_{\text{POP}_{22}}$.
- $\mathsf{pair} \notin \Sigma'_{\text{INT}_1^{\equiv}}$ and $\mathsf{pair} \notin \Sigma'_{\text{INT}_2^{\equiv}}$.

– $Proj_2$ is similarily defined as $Proj_1$. We add q_2 to Q_f instead of q_1.

– Dec.
- If $\mathsf{enc} \in \Sigma'_{\text{PUSH}}$, for every state $q \in Q_f$, if $\mathsf{enc}(q_1(x), q_2(y)) \to q(h(x,y)) \in \Delta$ for some symbol h, we add q_1 to Q_f if $L(\mathcal{A}, q_2) \cap L(\mathcal{A}) \neq \emptyset$. This last verification is possible by Corollary 5 and Theorem 11 of [4].
- $\mathsf{enc} \in \Sigma'_{\text{INT}_1}$ or $\mathsf{enc} \in \Sigma'_{\text{INT}_2}$. These cases are analogous to $\mathsf{enc} \in \Sigma'_{\text{PUSH}}$.
- If $\mathsf{enc} \in \Sigma'_{\text{POP}_{22}}$, for every state $q \in Q_f$, if $\mathsf{pair}(q_1(x), q_2(h(x,y))) \to q(y) \in \Delta$ for some symbol h, we construct the $\text{VTAM}^{\equiv}_{\not\equiv}$ $\mathcal{A}_\cap = L(\mathcal{A}, q_2) \cap \mathcal{A}$. If $\mathcal{A}_\cap = \emptyset$ we do not add any state to Q_f. Otherwise we verify that $M(A_\cap) \cap \mathcal{A}_h \neq \emptyset$, $\mathcal{A}_h$ being defined as for $\mathsf{pair} \in \Sigma'_{\text{POP}_{22}}$. If it is not empty, we add q_1 to Q_f.
  The case for $\mathsf{pair}(q_1(x), q_2(\perp)) \to q(\perp) \in \Delta$ is analogous.
- $\mathsf{enc} \in \Sigma'_{\text{POP}_{11}}$, $\mathsf{enc} \in \Sigma'_{\text{POP}_{12}}$, $\mathsf{enc} \in \Sigma'_{\text{POP}_{21}}$. These cases are similar to $\mathsf{enc} \in \Sigma'_{\text{POP}_{22}}$.
- $\mathsf{enc} \notin \Sigma'_{\text{INT}_1^{\equiv}}$ and $\mathsf{enc} \notin \Sigma'_{\text{INT}_2^{\equiv}}$.

– $Pair$. Let q_{pair} be a new state in Q_f.
- $\mathsf{pair} \in \Sigma'_{\text{PUSH}}$. For every $q_1, q_2 \in Q_f$, we add $\mathsf{pair}(q_1(x), q_2(y)) \to q_{pair}(h(x,y))$ to Δ for some symbol $h \in \Gamma$.
- $\mathsf{pair} \in \Sigma'_{\text{INT}_1}$ For every $q_1, q_2 \in Q_f$, we add $\mathsf{pair}(q_1(x), q_2(y)) \to q_{pair}(x)$ to Δ.
- $\mathsf{pair} \in \Sigma'_{\text{INT}_2}$. For every $q_1, q_2 \in Q_f$, we add $\mathsf{pair}(q_1(x), q_2(y)) \to q_{pair}(y)$ to Δ.
- $\mathsf{pair} \in \Sigma'_{\text{POP}_{22}}$. For every $q_1, q_2 \in Q_f$ and for every symbol h in Γ, we add $\mathsf{pair}(q_1(x), q_2(h(y,z))) \to q_{pair}(z)$ to Δ. We also add $\mathsf{pair}(q_1(x), q_2(\perp)) \to q_{pair}(\perp)$ to Δ.
- $\mathsf{pair} \in \Sigma'_{\text{POP}_{11}}$, $\mathsf{pair} \in \Sigma'_{\text{POP}_{12}}$, $\mathsf{pair} \in \Sigma'_{\text{POP}_{21}}$. These cases are similar to $\mathsf{pair} \in \Sigma'_{\text{POP}_{22}}$.
- $\mathsf{pair} \notin \Sigma'_{\text{INT}_1^{\equiv}}$ and $\mathsf{pair} \notin \Sigma'_{\text{INT}_2^{\equiv}}$.

– Enc. Let q_{enc} be a new state in Q_f. The case is similar to pair, except that the state we consider in the transitions is not q_{pair} but q_{enc}.

- *Hash. Let q_0 and q_H be new states in Q and in Q_f. Let $0 \to q_0(\perp)$ be a new transition in Δ.*
  - $H \in \Sigma'_{\text{PUSH}}$. *For every $q_1 \in Q_f$, we add $H(q_0(x), q_1(y)) \to q_H(h(x,y))$ to Δ for some symbol $h \in \Gamma$.*
  - $H \in \Sigma'_{\text{INT}_1}$. *For every $q_1 \in Q_f$, we add $H(q_0(x), q_1(y)) \to q_H(x)$.*
  - $H \in \Sigma'_{\text{INT}_2}$. *For every $q_1 \in Q_f$, we add $H(q_0(x), q_1(y)) \to q_H(y)$.*
  - $H \in \Sigma'_{\text{POP}_{22}}$. *For every $q_1 \in Q_f$ and for every symbol h in Γ, $H(q_0(x), q_1(h(y,z))) \to q_H(z)$ to Δ. We also add $H(q_0(x), q_1(\perp) \to q_H(\perp)$ to Δ.*
  - $H \in \Sigma'_{\text{POP}_{11}}$, $H \in \Sigma'_{\text{POP}_{12}}$, $H \in \Sigma'_{\text{POP}_{21}}$. *These cases are similar to $H \in \Sigma'_{\text{POP}_{22}}$.*
  - $H \notin \Sigma'_{\text{INT}_1^{\equiv}}$ *and* $H \notin \Sigma'_{\text{INT}_2^{\equiv}}$.

Lemma 19. *For any partition p, $DY'(L(\mathcal{A})) = L(DY_p(\mathcal{A}))$.*

Proof. Let us first prove that $DY'(L(\mathcal{A})) \subseteq L(DY_p(\mathcal{A}))$.

Let p be a partition. Let t be a term in $DY'(L(\mathcal{A}))$. Then there exists a deduction π of $L(\mathcal{A}) \vdash_{DY'} t$. We claim that if $L(\mathcal{A}) \vdash_{DY'} t$ then $t \in L(DY_p(\mathcal{A}))$. Hence $t \in L(DY_p(\mathcal{A}))$ and $DY'(L(\mathcal{A})) \subseteq L(DY_p(\mathcal{A}))$.

Let us show by induction on the size of π, that if $L(\mathcal{A}) \vdash_{DY'} t$ then $t \in L(DY_p(\mathcal{A}))$.

Let $\mathcal{A}$ be $\{Q_{\mathcal{A}}, Q_{f\mathcal{A}}, \Delta_{\mathcal{A}}\}$ and $DY_p(\mathcal{A})$ be $\{Q_{DY_p(\mathcal{A})}, Q_{fDY_p(\mathcal{A})}, \Delta_{DY_p(\mathcal{A})}\}$.

Base case. $|\pi| = 0$. In this case $t \in L(\mathcal{A})$. Since $Q_{\mathcal{A}} \subseteq Q_{DY_p(\mathcal{A})}$, $Q_{f\mathcal{A}} \subseteq Q_{fDY_p(\mathcal{A})}$ and $\Delta_{\mathcal{A}} \subseteq \Delta_{DY_p(\mathcal{A})}$ we have that $t \in L(DY_p(\mathcal{A}))$.

Inductive case. Let r be the last rule of π. We consider all possibilities for the rule r.

- $r = pair$. We have that $t = \mathsf{pair}(u,v)$. Hence there exists two deductions π_1 of $L(\mathcal{A}) \vdash_{DY'} u$ and π_2 of $L(\mathcal{A}) \vdash_{DY'} v$. As $|\pi_1| < n$ and $|\pi_2| < n$, we have that $u \in L(DY_p(\mathcal{A}))$ and $v \in L(DY_p(\mathcal{A}))$. Hence there exists $m_1, m_2 \in T(\Gamma)$ such that $u \to^* q_1(m_1)$ and $v \to^* q_2(m_2)$ for some $q_1, q_2 \in Q_{fDY_p(\mathcal{A})}$.
  - $\mathsf{pair} \in \Sigma_{\text{PUSH}}$. By Definition 8, as $q_1, q_2 \in Q_{fDY_p(\mathcal{A})}$ there is a transition $\mathsf{pair}(q_1(x), q_2(y)) \to q_{pair}(h(x,y))$ for some h in $\Delta_{DY_p(\mathcal{A})}$. $\mathsf{pair}(u,v) \to q_{pair}(h(m_1,m_2))$. As by Definition 8 $q_{pair} \in Q_f$, $\mathsf{pair}(u,v) \in L(DY_p(\mathcal{A}))$.
  - $\mathsf{pair} \in \Sigma_{\text{INT}_1} \cup \Sigma_{\text{INT}_2}$. This case is similar to $\mathsf{pair} \in \Sigma_{\text{PUSH}}$.
  - $\mathsf{pair} \in \Sigma_{\text{POP}_{22}}$. By Definition 8, as $q_1, q_2 \in Q_{fDY_p(\mathcal{A})}$ there are transitions $\mathsf{pair}(q_1(x), q_2(h(y,z))) \to q_{pair}(z)$ for any $h \in \Gamma$ and $\mathsf{pair}(q_1(x), q_2(\perp)) \to q_{pair}(\perp)$. As the only symbol of arity 0 in Γ is $\perp$, whatever be the form of m_2, $\mathsf{pair}(q_1(m_2), q_2(m_2)) \to q_{pair}(t)$, for some t, and so $\mathsf{pair}(u,v) \to q_{pair}(t)$. As by Definition 8 $q_{pair} \in Q_f$, $\mathsf{pair}(u,v) \in L(DY_p(\mathcal{A}))$.
  - $\mathsf{pair} \in \Sigma_{\text{POP}_{11}} \cup \Sigma_{\text{POP}_{12}} \cup \Sigma_{\text{POP}_{21}}$. These cases are similar to $\mathsf{pair} \in \Sigma_{\text{POP}_{22}}$.
- $r = enc$ and $r = hash$. These cases are similar to $r = pair$.
- $r = proj_1$. We have that there exists a deduction π^- of $L(\mathcal{A}) \vdash_{DY'} \mathsf{pair}(u,v)$ for some term u such that $|\pi^-| < |\pi|$. By induction we have that $\mathsf{pair}(u,v) \in L(DY_p(\mathcal{A}))$. Hence there exists $m_1, m_2, m \in T(\Gamma)$, $q_1, q_2 \in Q$ and $q \in Q_f$ such that $\mathsf{pair}(q_1(m_1), q_2(m_2)) \to q(m)$.

- $\mathsf{pair} \in \Sigma_{\text{PUSH}}$. As $t \to^* q_1(m_1)$ and $u \to^* q_2(m_2)$, $L(\mathcal{A}, q_1), L(\mathcal{A}, q_2) \neq \emptyset$. Hence $q_1 \in Q_f$ by Definition 8 and $t \in L(DY_p(\mathcal{A}))$.
- $\mathsf{pair} \in \Sigma_{\text{INT}_1} \cup \Sigma_{\text{INT}_2}$. This case is similar to $\mathsf{pair} \in \Sigma_{\text{PUSH}}$.
- $\mathsf{pair} \in \Sigma_{\text{POP}_{22}}$. As $\mathsf{pair}(q_1(m_1), q_2(m_2)) \to q(m)$ and $u \to^* q_2(m_2)$, by Definition 8 we have that $q_1 \in Q_f$, and so $t \in L(DY_p(\mathcal{A}))$.
- $\mathsf{pair} \in \Sigma_{\text{POP}_{11}} \cup \Sigma_{\text{POP}_{12}} \cup \Sigma_{\text{POP}_{21}}$. These cases are similar to $\mathsf{pair} \in \Sigma_{\text{POP}_{22}}$.

– $r = proj_2$. This case is similar to $r = proj_1$.

– $r = dec$. We have that there exists two deductions π_1 of $L(\mathcal{A}) \vdash_{DY'} \mathsf{enc}(u, v)$ and π_2 of $L(\mathcal{A}) \vdash_{DY'} v$. As $|\pi_1| < n$ and $|\pi_2| < n$, we have that $\mathsf{enc}(u, v) \in L(DY_p(\mathcal{A}))$ and $v \in L(DY_p(\mathcal{A}))$. Hence there exists $m_1, m_2 \in T(\Gamma)$ such that $u \to^* q_1(m_1)$ and $v \to^* q_2(m_2)$ for some $q_1, q_2 \in Q_{DY_p(\mathcal{A})}$. As $v \in L(DY_p(\mathcal{A}))$, $L(DY_p(\mathcal{A}), q_2) \cap L(DY_p(\mathcal{A})) \neq \emptyset$.
  - $\mathsf{enc} \in \Sigma_{\text{PUSH}}$. $q_1 \in Q_f$ by Definition 8 and $t \in L(DY_p(\mathcal{A}))$.
  - $\mathsf{enc} \in \Sigma_{\text{INT}_1} \cup \Sigma_{\text{INT}_2}$. This case is similar to $\mathsf{enc} \in \Sigma_{\text{PUSH}}$.
  - $\mathsf{enc} \in \Sigma_{\text{POP}_{22}}$. As $\mathsf{enc}(q_1(m_1), q_2(m_2)) \to q(m)$ and $v \to^* q_2(m_2)$, by Definition 8 we have that $q_1 \in Q_f$, and so $t \in L(DY_p(\mathcal{A}))$.
  - $\mathsf{enc} \in \Sigma_{\text{POP}_{11}} \cup \Sigma_{\text{POP}_{12}} \cup \Sigma_{\text{POP}_{21}}$. These cases are similar to $\mathsf{enc} \in \Sigma_{\text{POP}_{22}}$.

Let us now prove that $L(DY_p(\mathcal{A})) \subseteq DY'(L(\mathcal{A}))$.

By induction on the number $|DY_p|$ of steps of completion of $\mathcal{A}$ we show that $L(DY_p(\mathcal{A})) \subseteq DY'(L(\mathcal{A}))$.

Base case. $|DY_p| = 0$. In this case $DY_p(\mathcal{A}) = \mathcal{A}$. Hence trivially $L(\mathcal{A}) \subseteq DY'(L(\mathcal{A}))$.

Inductive case. We call $DY_p^i(\mathcal{A})$ the automaton obtained after i applications of rules of DY_p for some rules. By induction hypothesis for every $DY_p^i(\mathcal{A})$, we have that $L(DY_p^i(\mathcal{A})) \subseteq DY'(L(\mathcal{A}))$. Let us show that for any r rules of DY_p applied to $DY_p^i(\mathcal{A})$ we still have that $L(r(DY_p^i(\mathcal{A}))) \subseteq DY'(L(\mathcal{A}))$.

Let us consider all possibilities for r:

– $r = Pair$. By Lemma 21, we have that $L(Pair(DY_p^i(\mathcal{A}))) \subseteq DY'(L(\mathcal{A}))$.

– $r = Enc$. $r = Hash$. These cases are similar to $r = Pair$.

– $r = Proj_1$. Let t be a term of $L(Proj_1(DY_p^i(\mathcal{A})))\backslash DY_p^i(\mathcal{A}))$. Hence $t \to^* q_1(m_1)$ for some m_1, such that there exists a transition $\mathsf{pair}(q_1(t_1), q_2(t_2)) \to q(t)$ in $DY_p^i(\mathcal{A})$, and $q \in Q_{f DY_p^i(\mathcal{A})}$, and a term u such that $u \to^* q_2(m_2)$ for some m_2. So $\mathsf{pair}(t, u) \in L(DY_p^i(\mathcal{A}))$. As $L(DY_p^i(\mathcal{A})) \subseteq DY'(L(\mathcal{A}))$, $\mathsf{pair}(t, u) \in DY'(L(\mathcal{A}))$. By an application of $proj_1$ we have that $t \in DY'(L(\mathcal{A}))$.

– $r = Proj_2$ or $r = Dec$. This case is similar to $r = Proj_1$.

Lemma 20. *Let $\mathcal{A}$ be a* $\text{VTAM}^{\equiv}_{\not\equiv}$ *and $DY_p^i(\mathcal{A})$ the obtained* $\text{VTAM}^{\equiv}_{\not\equiv}$ *by i application of rules of DY_p. Let v be a term such that $v \to^* q_{pair}(m)$ for some m. Then for every term t such that $v \in St(t)$ if $t \to^* q'(m')$ for some m', $q' = q_{pair}$ or $q' = q_{enc}$ or $q' = q_H$.*

Proof. By induction on the structure of t.

Base case. $t = v$. The result is straightforward.

Inductive case. $t = h(t_1, t_2)$ for some symbol $h \in \Sigma$ and $t \to^* q'(m')$ for some m', let us show that $q' = q_{pair}$, $q' = q_{enc}$ or $q' = q_H$. As $v \in St(t)$, either $v \in St(t_1)$ or $v \in St(t_2)$. Without loss of generality we supose that $v \in St(t_1)$. As $t \to^* q'(m')$, there are states q_1, q_2 and $m_1, m_2 \in T(\Gamma)$ such that $t_1 \to^* q_1(m_1)$ and $t_2 \to^* q_2(m_2)$.

As t_1 is a subterm of t, $v \in St(t_1)$ and there exists q_1 and m_1 such that $t_1 \to^* q_1(m_1)$, by induction we have that $q_1 = q_{pair}$, $q_1 = q_{enc}$ or $q_1 = q_H$. Let us suppose that $q_1 = q_{pair}$, if $h(q_{pair}(m_1), q_2(m_2)) \to q'(m')$ then the tansition $h(q_{pair}(t_1), q_2(t_2)) \to q(t) \in DY_p^i(\mathcal{A}) \setminus \mathcal{A}$ We can verify that for every transition of this kind $q = q_{pair}$, $q == q_{enc}$ or $q_1 = q_H$. We do the same reasoning if $q_1 = q_{enc}$ or if $q_1 = q_H$.

Lemma 21. *Let $\mathcal{A}$ be a* $\text{VTAM}_{\not\equiv}^{\equiv}$ *and $DY_p^i(\mathcal{A})$ a* $\text{VTAM}_{\not\equiv}^{\equiv}$ *obtained by i applications of rules of DY_p. If $DY_p^i(\mathcal{A}) \subseteq DY'(L(\mathcal{A}))$, then $L(DY_p^i(\mathcal{A}) \cup \{\mathsf{pair}(q_1(t_1), q_2(t_2)) \to q_{pair}(t)\}) \subseteq DY'(L(\mathcal{A}))$, for some accepting states q_1 and q_2.*

Proof. Let u be a term in $L(DY_p^i(\mathcal{A}) \cup \{\mathsf{pair}(q_1(t_1), q_2(t_2)) \to q_{pair}(t)\})$. If $u \in L(DY_p^i(\mathcal{A}))$ the result follows directly. Else $u \in L(DY_p^i(\mathcal{A}) \cup \{\mathsf{pair}(q_1(t_1), q_2(t_2)) \to q_{pair}(t)\}) \setminus L(DY_p^i(\mathcal{A}))$.

In this case there exists a term v such that $v \in St(u)$ and $v \to q_{pair}(m)$ for some m. By Lemma 20 we can conclude that $u = \mathsf{pair}(u_1, u_2)$, $u = \mathsf{enc}(u_1, u_2)$ or $u = H(u_1, u_2)$. Without loss of generality we suppose that $u = \mathsf{pair}(u_1, u_2)$.

Let d be a integer such that $d = |u| - |v|$ where $|u|$ (resp. $|v|$) is the size of a term u (resp. v), and v is the smallest subterm of u such that $v \to q_{pair}(m)$ for some m. By induction on d we show that if $u \in L(DY_p^i(\mathcal{A}) \cup \{\mathsf{pair}(q_1(t_1), q_2(t_2)) \to q_{pair}(t)\}) \setminus L(DY_p^i(\mathcal{A}))$ then $u \in DY'(L(\mathcal{A}))$.

Base case. $d = 0$. Hence $u = v$. In this case $u = \mathsf{pair}(u_1, u_2)$ and there exists m_1 and m_2 in $T(\Gamma)$ such that $u_1 \to^* q_1(m_1)$ and $u_2 \to^* q_2(m_2)$ in $DY_p^i(\mathcal{A})$. As $q_1, q_2 \in Q_f$, $u_1, u_2 \in L(DY_p^i(\mathcal{A}))$ and as $L(DY_p^i(\mathcal{A})) \subseteq DY'(L(\mathcal{A}))$, $u_1, u_2 \in DY'(L(\mathcal{A}))$. It follows that $\mathsf{pair}(u_1, u_2) \in DY'(L(\mathcal{A}))$.

Inductive case. $u = \mathsf{pair}(u_1, u_2)$. If there are strict subterms v_1 of u_1 and v_2 of u_2 such that $v_1, v_2 \in L(DY_p^i(\mathcal{A}) \cup \{\mathsf{pair}(q_1(t_1), q_2(t_2)) \to q_{pair}(t)\}) \setminus L(DY_p^i(\mathcal{A}))$. In this case by induction $u_1, u_2 \in DY'(L(\mathcal{A}))$. It follows that $\mathsf{pair}(u_1, u_2) \in DY'(L(\mathcal{A}))$. Or for u_1 or u_2 there are no subterm v such that of $v \to q_{pair}(m)$ for some m. It implies that $u_1 \to^* q_1(m_1)$ or $u_2 \to^* q_2(m_2)$ in $DY_p^i(\mathcal{A})$ for some term $m_1, m_2 \in T(\Gamma)$. Hence we have that $u_1 \in DY'(L(\mathcal{A}))$. We can conclude in a similar way as before.

C The Complete Example

$$
\begin{aligned}
E &= E_1 \cup E_2 \\
E_1 &= \{\alpha^{N_{i_1}^{j_1} \cdot \ldots \cdot N_{i_k}^{j_k}} | k < i_1\} \\
E_2 &= \{\alpha^{N_{i_1}^{j_1} \cdot N_{(i_1-1)}^{j_2} \cdot \ldots \cdot N_1^{j_{i_1}}} | i_1 < j_1\} \\
K &= \{\alpha^{N_{j_1}^{j_1} \cdot N_{(j_1-1)}^{j_2} \cdot \ldots \cdot N_1^{j_{j_1}}}\}
\end{aligned}
$$

C.1 E_1 and E_2 definitions

Representation of $\mathcal{A}_{E_1}$:

$Q_{\mathcal{A}_{E_1}} = \{q_d, q_{1s'}, q_s, q_{nt}, q_{narg}, q_\alpha, q_{acc}\}$
$Q_{f\mathcal{A}_{E_1}} = \{q_{acc}\}$

$$
\begin{array}{rcl}
0 & \rightarrow & q_d(\perp) \\
\alpha & \rightarrow & q_\alpha(\perp) \\
\\
s(q_d(m), q_d(m')) & \rightarrow & q_d(m') \\
s'(q_d(m), q_d(m')) & \rightarrow & q_{1s'}(S(m, m')) \\
s'(q_d(m), q_{1s'}(m')) & \rightarrow & q_{s'}(S'(m, m')) \\
s'(q_d(m), q_{s'}(m')) & \rightarrow & q_{s'}(S'(m, m')) \\
s(q_d(m), q_{s'}(m')) & \rightarrow & q_{s'}(S'(m, m')) \\
N(q_d(m), q_{s'}(m')) & \rightarrow & q_n(m') \\
\\
t(q_d(m), q_n(S'(m', m''))) & \rightarrow & q_{nt}(m'') \\
t(q_d(m), q_{narg}(S'(m', m''))) & \rightarrow & q_{nt}(m'') \\
\\
\mathsf{mult}(q_n(m), q_{nt}(m')) & \stackrel{m \equiv m'}{\rightarrow} & q_{narg}(m) \\
\mathsf{mult}(q_n(m), q_{nt}(m')) & \stackrel{m \not\equiv m'}{\rightarrow} & q_{narg}(m) \\
\\
\mathsf{exp}(q_\alpha(m), q_{nt}(m')) & \rightarrow & q_{acc}(h(m, m')) \\
\mathsf{exp}(q_\alpha(m), q_n(m')) & \rightarrow & q_{acc}(h(m, m'))
\end{array}
$$

Representation of $\mathcal{A}_{E_2}$:

$Q_{\mathcal{A}_{E_2}} = \{q_d, q_s, q_{nt}, q_{narg}, q_\alpha, q_{acc}\}$
$Q_{f\mathcal{A}_{E_2}} = \{q_{acc}\}$

$$\begin{array}{rcl}
0 & \to & q_d(\perp) \\
\alpha & \to & q_\alpha(\perp) \\
\\
s(q_d(m), q_d(m')) & \to & q_s(m') \\
s'(q_d(m), q_s(m')) & \to & q_{s'}(S'(m, m')) \\
s'(q_d(m), q_{s'}(m')) & \to & q_{s'}(S'(m, m')) \\
N(q_d(m), q_{s'}(m')) & \to & q_n(m') \\
\\
t(q_d(m), q_n(S'(m', m''))) & \to & q_{nt}(m'') \\
t(q_d(m), q_{narg}(S'(m', m''))) & \to & q_{nt}(m'') \\
\\
\mathsf{mult}(q_n(m), q_{nt}(m')) & \overset{m \equiv m'}{\to} & q_{narg}(m) \\
\\
\mathsf{exp}(q_\alpha(m), q_{nt}(m')) & \to & q_{acc}(h(m, m')) \\
\mathsf{exp}(q_\alpha(m), q_n(m')) & \to & q_{acc}(h(m, m'))
\end{array}$$

C.2 Execution of $\mathcal{A}_K$

We consider the term $t = \alpha^{N_3^3 \cdot N_2^3 \cdot N_1^3}$.

$\rho(t) = \mathsf{exp}(\alpha, \mathsf{mult}(N(0, s'(0, s'(0, s'(0, 0)))),$
$t(\mathsf{mult}(N(0, s'(0, s'(0, s(0, 0)))), t(N(0, s'(0, s(0, s(0, 0)))))))$

We can verify that :

$W(\rho(t)) = \mathsf{exp}(\alpha, \mathsf{mult}(N(0, s'(0, s(0, s(0, 0)))),$
$t(\mathsf{mult}(N(0, s'(0, s'(0, s(0, 0)))), t(N(0, s'(0, s'(0, s'(0, 0)))))))))$.

Let us first show that by the transition $0 \to q_d(\perp)$ we have trivially that $0 \to q_d(\perp)$.

By the transition $s'(q_d(m), q_d(m')) \to q_{s'ent}(S(m, m'))$:
$s'(0, 0) \to^* q_{s'ent}(S'(\perp, \perp))$.

By the transition $s'(q_d(m), q_{s'ent}(m')) \to q_{s'ent}(S(m, m'))$:
$s'(0, s'(0, s'(0, 0))) \to^* q_{s'ent}(S'(\perp, S'(\perp, S'(\perp, \perp))))$.

By the transition $N(q_d(m), q_{s'ent}(m')) \to q_{nent}(m')$:
$N(0, s'(0, s'(0, s'(0, 0)))) \to^* q_{nent}(S'(\perp, S'(\perp, S'(\perp, \perp))))$

By the same kind of analysis we can check that :
$N(0, s'(0, s'(0, s(0, 0)))) \to^* q_n(S'(\perp, S'(\perp, \perp)))$
and that
$N(0, s'(0, s(0, s(0, 0)))) \to^* q_{nonly1s'}(S'(\perp, \perp))$.

By the transition $t(q_d(m), q_{nent}(S'(m', m''))) \to q_{nt}(m'')$:
$t(N(0, s'(0, s'(0, s'(0, 0))))) \to^* q_{nt}(S'(\perp, S'(\perp, \perp)))$.

By the transition $\mathsf{mult}(q_n(m), q_{nt}(m')) \overset{m \equiv m'}{\to} q_{narg}(m)$:
$\mathsf{mult}(N(0, s'(0, s'(0, s(0, 0)))),$
$t(N(0, s'(0, s'(0, s'(0, 0)))))) \to^* q_{arg}(S'(\perp, S'(\perp, \perp)))$.

By the transition $t(q_d(m), q_{narg}(S'(m', m''))) \to q_{nt}(m'')$:
$t(\mathsf{mult}(N(0, s'(0, s'(0, s(0, 0)))), t(N(0, s'(0, s'(0, s'(0, 0))))))) \to^* q_{nt}(S'(\perp, \perp))$.

By the transition $\mathsf{mult}(q_{nonly1s'}(m), q_{nt}(m')) \overset{m \equiv m'}{\to} q_{exp}(m)$:

$\mathsf{mult}(N(0, s'(0, s(0, s(0, 0)))), t(\mathsf{mult}(N(0, s'(0, s'(0, s(0, 0)))), t(N(0, s'(0, s'(0, s'(0, 0)))))))) \to^*$ $q_{exp}(S'(\perp, \perp))$.

Finally as $\alpha \to q_\alpha(\perp)$ by the transition $\alpha \to q_\alpha(\perp)$, $\rho(t) \to^* q_{acc}(\perp, S'(\perp, \perp))$.