INSTITUT FÜR INFORMATIK

UND PRAKTISCHE MATHEMATIK

Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents

Yannick Chevalier, Ralf Küsters, Michaël Rusinowitch, and Mathieu Turuani

Bericht Nr. 0305

June 2003

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

KIEL

Institut für Informatik und Praktische Mathematik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents

Yannick Chevalier, Ralf Küsters, Michaël Rusinowitch, and
Mathieu Turuani

Bericht Nr. 0305
June 2003

e-mail:
chevalie@loria.fr,kuesters@theory.stanford.edu,{rusi,turuani}@loria.fr

Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents*

Yannick Chevalier†, Ralf Küsters‡, Michaël Rusinowitch†, and Mathieu Turuani†

† LORIA-INRIA-Université Henri Poincaré,
54506 Vandoeuvre-les-Nancy cedex, France
email:{chevalie,rusi,turuani}@loria.fr

‡ Department of Computer Science
Stanford University, Stanford CA 94305, USA
email: kuesters@theory.stanford.edu

Abstract

We present an NP decision procedure for the formal analysis of protocols in presence of modular exponentiation with products allowed in exponents. The number of factors that may appear in the products is unlimited. We illustrate that our model is powerful enough to uncover known attacks on the A-GDH.2 protocol suite.

1 Introduction

Most automatic analysis techniques for security protocols take as a simplifying hypothesis that the cryptographic algorithms are perfect: One needs the decryption key to extract the plaintext from the ciphertext, and also, a ciphertext can be generated only with the appropriate key and message (no collision). Under these assumptions and given a bound on the number of protocol sessions, the insecurity problem is decidable [13]. However, it is an open question whether this result remains valid when the intruder model is extended to take into account even simple properties of product or exponentiation operators, such as those of RSA and Diffie-Hellman (DH) exponentiation. This question is important since many security flaws are the consequence of these properties and many protocols are based on these operators (see, e.g., [12]).

Only recently the perfect encryption assumption for protocol analysis has been slightly relaxed. In [9, 8], unification algorithms are designed for handling properties of Diffie-Hellman cryptographic systems. Although these results are useful, they do not solve the more general insecurity problem. In [4, 6], decidability of security has been proved for protocols that employ exclusive or. When the XOR operator is replaced by an abelian group operator, decidability is mentioned as an open problem by [6] and [10]. In [10], there is a reduction from the insecurity problem to solving quadratic equation systems. However, the satisfiability of these systems is in general undecidable. Hence, this approach fails to solve the initial insecurity problem.

In this paper, using non-trivial extensions of our technique in [4], we show that the insecurity problem for protocols that use DH exponentiation with products in exponents is NP-complete. Our model is powerful enough to uncover for instance attacks on the A-GDH.2 protocol suite discovered in [12].

*This work was partially supported by PROCOPE and IST AVISPA. The second author was also supported by the DFG.

A similar problem has been addressed by Boreale and Buscemi [3]. They propose a decision procedure that is sound and complete with respect to an operational semantics for which it is however not clear whether all relevant algebraic properties of DH exponentiation are taken into account. Also, Boreale and Buscemi put an a priori bound on the number of factors that may occur in products, while in the present paper we allow an unlimited number of factors. Finally, in their paper no complexity result is provided. DH exponentiation is also studied by Millen and Shmatikov [10]. However, they do not provide a decision procedure. Also, they assume the base in exponentiations to be a fixed constant while we allow arbitrary messages.

Structure of the paper. In Section 2, we introduce our protocol and intruder model. The intruder is the standard Dolev-Yao intruder extended with what we call oracle rules which provide the intruder with additional capabilities for deriving messages. For this general framework, we state a theorem which says that under certain conditions deciding the insecurity of a protocol with respect to the extended intruder is NP-complete (Section 3). This section also contains an NP decision procedure for deciding insecurity and an overview of the proof of the theorem. Finally, in this section we point out the main differences to our proof presented in [4] for XOR. In Section 4 and 5, the theorem is proved. In Section 6, we consider a certain instance of our general framework. More specifically, we instantiate oracle rules by DH exponentiation which yields what we call the DH intruder. We show that DH exponentiation indeed falls into our general framework and that the conditions of the theorem presented in Section 3 are met. As a consequence, we can derive the main result of this paper, namely that deciding the insecurity of a protocol with respect to the DH intruder is an NP-complete problem. To illustrate our model, in Section 7 we formally specify the A-GDH.2 protocol and present an attack on it discovered by Pereira and Quisquater.

2 The Protocol and Intruder Model

The protocol and intruder model we describe here extends standard models for (automatic) analysis of security protocols [1, 13, 11] in two respects. First, messages can be build using the operator $Exp(\cdot,\cdot)$, which stands for exponentiation, and a product operator "$\cdot$". Second, in addition to the standard Dolev-Yao rewrite rules, the intruder is equipped with the mentioned oracle rules, which are later instantiated with rules to exponentiate terms. In what follows, we provide a formal definition of our model by defining terms, messages, protocols, the intruder, and attacks.

2.1 Terms and Messages

The set of terms *term* is defined by the following grammar:

$$\begin{array}{rcl} term & ::= & \mathcal{A} \mid \mathcal{V} \mid \langle term, term\rangle \mid \{term\}^s_{term} \\ & & \mid \{term\}^p_{\mathcal{K}} \mid Exp(term, product) \\ product & ::= & term^{\boldsymbol{Z}} \mid term^{\boldsymbol{Z}} \cdot product \end{array}$$

where $\mathcal{A}$ is a finite set of constants (*atomic messages*), containing principal names, nonces, keys, and the constants 1 and secret; $\mathcal{K}$ is a subset of $\mathcal{A}$ denoting the set of public and private keys; $\mathcal{V}$ is a finite set of variables; and $\boldsymbol{Z}$ is the set of integers, the *product exponents*. We assume that there is a bijection $\cdot^{-1}$ on $\mathcal{K}$ which maps every public (private) key k to its corresponding private (public) key k^{-1}. The binary symbol $\langle\cdot,\cdot\rangle$ is called *pairing*, the binary symbol $\{\cdot\}^s_\cdot$ is called *symmetric encryption*, the binary symbol $\{\cdot\}^p_\cdot$ is *public key encryption*. Note that a symmetric key can be any term and that for public key encryption only atomic keys (namely, public and private keys from $\mathcal{K}$) can be used. The product operator "$\cdot$" models multiplication in an abelian group. For instance, the product $a^2 \cdot b^3 \cdot c^{-2}$ stands for an element of this group where $a^2 = a \cdot a$, $b^3 = b \cdot b \cdot b$, and $c^{-2} = c^{-1} \cdot c^{-1}$ with c^{-1} the inverse of c. In the A-GDH.2 protocol the abelian group is a

subgroup G of of order q of the multiplicative group $\mathbb{Z}_p^*$ where p and q are prime numbers. Terms and products are read modulo commutativity and associativity of the product operator as well as the identity $t^1 = t$. For instance, $d^1 \cdot c^{-2} \cdot (b^3 \cdot a^2)$ and $a^2 \cdot b^3 \cdot c^{-2} \cdot d$ are considered the same products. Also, when we write $t_1^{z_1} \cdots t_n^{z_n}$, then we always assume that the head symbol of t_i is not a product opertor. This can always be achieved by resolving parentheses. We write $t_1^* \cdots t_n^*$ to denote a product of the form $t_1^{z_1} \cdots t_n^{z_n}$ where the z_i are some non-zero integers. The operator $Exp(\cdot, \cdot)$ stands for exponentiation. For instance, $Exp(a, b^2 \cdot c^{-1})$ is $a^{b^2 \cdot c^{-1}}$.

If $t, t_1, \ldots, t_n$ are terms with $n \geq 2$, then we call a product of the form t^z for some $z \neq 1$ or a product of the form $t_1^{z_1} \cdots t_n^{z_n}$ a *non-standard term*. We often refer to a term or a product as a "term". We say *standard term* to distinguish a term from a non-standard term.

Variables are denoted by $x, y, \ldots$, terms are denoted by s, t, u, v, finite sets of terms are written $E, F, \ldots$, and decorations thereof, respectively. We abbreviate $E \cup F$ by E, F, the union $E \cup \{t\}$ by E, t, and $E \setminus \{t\}$ by $E \setminus t$.

For a term t and a set of terms E, $\mathcal{V}(t)$ and $\mathcal{V}(E)$ denote the set of variables occurring in t and E, respectively.

A *ground term* (also called *message*) is a term without variables. We use the expressions *standard* and *non-standard messages* in the same way we use standard and non-standard terms. A *(ground) substitution* is a mapping from $\mathcal{V}$ into the set of *standard* (ground) terms. The application of a substitution σ to a term t (a set of terms E) is written $t\sigma$ ($E\sigma$), and is defined as usual.

We say that two terms t and t' *coincide modulo product exponents* ($t \approx t'$, for short) if t and t' are equal except for the product exponents. For instance, $\langle a^2, b^{-3} \cdot c \rangle \approx \langle a^0, b^2 \cdot c^5 \rangle \not\approx \langle b^1, a^2 \cdot c^5 \rangle$. This equivalence relation extends to substitutions in the obvious way: $\sigma \approx \sigma'$ iff $\sigma(x) \approx \sigma'(x)$ for every variable x.

Given a *standard* term u and a term v, the *replacement* $\delta = [u \leftarrow v]$ of u by v maps every term t to the term $t[u \leftarrow v]$ which is obtained by replacing all occurrences of u in t by v. We summarize simple properties:

Lemma 1 *Let $\delta = [u \leftarrow v]$ be a replacement, t_i be standard terms, and z_i integers. Then,*

- $(t_1^{z_1} \cdots t_p^{z_p})\delta = (t_1\delta)^{z_1} \cdots (t_p\delta)^{z_p}$,
- $Exp(t_0, t_1^{z_1} \cdots t_p^{z_p})\delta = Exp(t_0\delta, (t_1\delta)^{z_1} \cdots (t_p\delta)^{z_p})$ *in case* $u \neq Exp(t_0, t_1^{z_1} \cdots t_p^{z_p})$, *and*
- *the result of a replacement is uniquely determined.*

We can compose a substitution σ with a replacement δ: the substitution $\sigma\delta$ maps every $x \in \mathcal{V}$ to $\sigma(x)\delta$.

The set of *subterms* of a term t, denoted by $\mathcal{S}(t)$, is defined as follows:

- If $t \in \mathcal{A}$ or $t \in \mathcal{V}$, then $\mathcal{S}(t) = \{t\}$.
- If $t = \langle u, v \rangle$, $\{u\}_v^s$, or $\{u\}_v^p$, then $\mathcal{S}(t) = \{t\} \cup \mathcal{S}(u) \cup \mathcal{S}(v)$.
- If $t = Exp(u, t_1^{z_1} \cdots t_p^{z_p})$, then $\mathcal{S}(t) = \{t\} \cup \mathcal{S}(u) \cup \bigcup_i \mathcal{S}(t_i)$.
- If $t = t_1^{z_1} \cdots t_p^{z_p}$, then $\mathcal{S}(t) = \{t\} \cup \bigcup_i \mathcal{S}(t_i)$. Recall that the t_i are standard terms.

We define $\mathcal{S}(E) = \bigcup\{\mathcal{S}(t) \mid t \in E\}$. Note that $Exp(a, b^2 \cdot c^1)$ and $b^2 \cdot c^1 \cdot d^1$ are not subterms of $Exp(a, b^2 \cdot c^1 \cdot d^1)$.

We define the set $\mathcal{S}_{ext}(t)$ of *extended subterms* of t to be $\mathcal{S}_{ext}(t) = \mathcal{S}(t) \cup \{M \mid Exp(u, M) \in \mathcal{S}(t)\}$. Thus, the only difference to the definition of subterms is that for subterms $Exp(u, M)$ of t the product M also belongs to the set of (extended) subterms of t.

The set of *factors* of a term t, denoted by $\mathcal{F}(t)$, is recursively defined:

- If t is standard and not $Exp(\cdot, \cdot)$, then $\mathcal{F}(t) = \{t\}$.

- If $t = Exp(u, t_1^* \cdots t_p^*)$, then $\mathcal{F}(t) = \{u, t_1, .., t_p\}$.
- If $t = t_1^* \cdots t_p^*$, then $\mathcal{F}(t) = \{t_1, .., t_p\}$.

Note that $\mathcal{F}(t)$ only contains standard terms. For example, with $a, b, c \in \mathcal{A}$, $\mathcal{F}(a^2 \cdot b^1 \cdot c^{-1}) = \{a, b, c\}$.

We consider to different ways of measuring the size of a term, one includes the product exponents and the other one does not. In any case, the size is defined according to the DAG size of a term. We define $|t| := \mathsf{Card}(\mathcal{S}(t))$ ($|(|_{ext}t) := \mathsf{Card}(\mathcal{S}_{ext}(t))$) to be the number of (extended) subterms of t.

Remark 1 *We note that* $|t|_{ext} \leq 2 \cdot |t|$.

Note that the definition of $|\cdot|$ and $|\cdot|_{ext}$ do not measure the size of product exponents. We set

$$||t||_{exp} := \Sigma_{t_1^{z_1} \cdots t_n^{z_n} \in \mathcal{S}(t)} |z_1| + \ldots + |z_n|,$$

where $|z_i|$ is the number of bits needed to represent the integer z_i in binary, to measure the space needed to represent the product exponents occurring in t. Finally,

$$||t|| := |t| + ||t||_{exp}.$$

For a set of terms E the size is defined in the same way (replace t by E in the above definitions). For a substitution σ, we define

$$|\sigma| = \Sigma_{x \in \mathcal{V}} |\sigma(x)|$$

and analogously, we define $||\sigma||_{exp}$ and $||\sigma||$.

One easily shows by structural induction:

Lemma 2 *Let s be a standard term, t be a term, and x be a variable or an atomic message. Let δ be the replacement $[s \leftarrow x]$. Then, $|t\delta| \leq |t|$.*

We now formulate the algebraic properties of terms. Recall that terms are read modulo commutativity and associativity of the product operator as well as the identiy $t^1 = t$. In addition, we consider the following properties:

$$\begin{aligned}
t \cdot 1 &= t && (1)\\
t^0 &= 1 && (2)\\
1^z &= 1 && (3)\\
t^z \cdot t^{z'} &= t^{z+z'} && (4)\\
Exp(t, 1) &= t && (5)\\
Exp(Exp(t, t'), t'') &= Exp(t, t' \cdot t'') && (6)
\end{aligned}$$

A *normal form* of a term is obtained by iteratively applying these identities from left to write. Note that the identities can be applied to subterms of terms and that the normal forms are uniquely determined up to commutativity and associativity of the product operator. Since we consider terms modulo commutativity and associativity, normal forms are uniquely determined. The normal form of a term t is denoted by $\ulcorner t \urcorner$. We illustrate the notion of a normal form by some examples: If $a, b, c, d \in \mathcal{A}$, then

- $\ulcorner (a^2 \cdot b^1) \cdot b^{-2} \urcorner = a^2 \cdot b^{-1}$
- $\ulcorner Exp(Exp(a, b^1 \cdot c^1), c^{-1} \cdot d^1) \urcorner = Exp(a, b \cdot d)$
- $\ulcorner Exp(a, b^1 \cdot c^1 \cdot (c^{-1} \cdot d^1)^1) \urcorner = Exp(a, b \cdot d)$

- $\ulcorner Exp(Exp(a,(b^1\cdot c^{-2})^3,b^{-3}),c^6)\urcorner = a$
- $\ulcorner\langle a,b\rangle\cdot\langle a^{-1},c\rangle\urcorner \neq \langle 0,b\cdot c\rangle$.

The normal forms of sets of terms and substitutions are defined in the obvious way. A term t is *normalized* if $\ulcorner t\urcorner = t$. In the same way normalized sets and substitutions are defined. Two terms t and t' are *equivalent* (modulo *Exp* and $\cdot$) if $\ulcorner t\urcorner = \ulcorner t'\urcorner$.

One easily shows:

Lemma 3 *For every term t, t' and substitution σ:*

1. $|\ulcorner t\urcorner| \leq |t|$,
2. $||\ulcorner t\urcorner||_{exp} \leq ||t||_{exp}$,
3. $||\ulcorner t\urcorner|| \leq ||t||$, *and*
4. $\ulcorner t\sigma\urcorner = \ulcorner\ulcorner t\urcorner\sigma\urcorner = \ulcorner t\ulcorner\sigma\urcorner\urcorner = \ulcorner\ulcorner t\urcorner\ulcorner\sigma\urcorner\urcorner$.

2.2 Protocols

The following definition is explained below.

Definition 1 *A* protocol rule *is of the form $R \Rightarrow S$ where R and S are standard terms.*

A protocol *P is a tuple $(\{R_i \Rightarrow S_i, i \in \mathcal{I}\}, <_{\mathcal{I}}, E)$ where E is a finite normalized set of standard messages with $1 \in E$, the* initial intruder knowledge, *$\mathcal{I}$ is a finite (index) set, $<_{\mathcal{I}}$ is a partial ordering on $\mathcal{I}$, and $R_i \Rightarrow S_i$, for every $i \in \mathcal{I}$, is a protocol rule such that*

1. *the (standard) terms R_i and S_i are normalized;*
2. *for all $x \in \mathcal{V}(S_i)$, there exists $j \leq_{\mathcal{I}} i$ such that $x \in \mathcal{V}(R_j)$;*
3. *for every subterm $Exp(t_1, t_2^{z_2}\cdots t_n^{z_n})$ of R_i, there exists $k \in \{1,..,n\}$ such that $\mathcal{V}(t_l) \subseteq \cup_{j<_{\mathcal{I}} i}\mathcal{V}(R_j)$ for every $l \in \{1,..,n\}\setminus\{k\}$.*

A bijective mapping $\pi : \mathcal{I}' \to \{1,..,p\}$ is called execution ordering *for P if $\mathcal{I}' \subseteq \mathcal{I}$, p is the cardinality of $\mathcal{I}'$, and for all i, j we have that if $i <_{\mathcal{I}} j$ and $\pi(j)$ is defined, then $\pi(i)$ is defined and $\pi(i) < \pi(j)$. We define the* size *of π to be p.*

Given a protocol P, in the following we will assume that $\mathcal{A}$ is the set of constants occurring in P. We define $\mathcal{S}(P) := E \cup \bigcup_{i\in\mathcal{I}}(R_i \cup S_i))$ to be the *set of subterms of* P, $\mathcal{V} := \mathcal{V}(P)$ to be the set of variables occurring in P, and $|P| := |\mathcal{S}(P)|$ and $||P|| := ||\mathcal{S}(P)||$ to be the different *sizes of* P. Analogously, $|P|_{ext}$ and $||P||_{ext}$ are defined.

Intuitively, when executing a rule $R_i \Rightarrow S_i$ and on receiving a (normalized) message m in a protocol run, it is first checked whether m and R_i match, i.e., whether there exists a ground substitution σ such that $\ulcorner m\urcorner = \ulcorner R_i\sigma\urcorner$. If so, $\ulcorner S_i\sigma\urcorner$ is returned as output. We always assume that the messages exchanged between principals (and the intruder) are normalized — therefore, m is assumed to be normalized and the output of the above rule is not $S_i\sigma$ but $\ulcorner S_i\sigma\urcorner$. This is because principals and the intruder cannot distinguish between equivalent terms, and therefore, they may only work on normalized terms (representing the corresponding equivalence class of terms). Finally, we note that since the different protocol rules may share variables, some of the variables in R_i and S_i may be bounded already by substitutions obtained from applications of previous protocol rules. We are not actually interested in a normal execution of a protocol but rather in attacks on a protocol. This is the reason why the definition of a protocol contains the initial intruder knowledge. Attacks are formally defined in Section 2.3.

	Decomposition rules	Composition rules
Pair	$L_{p1}(\langle m, m' \rangle)$: $\langle m, m' \rangle \to m$ $L_{p2}(\langle m, m' \rangle)$: $\langle m, m' \rangle \to m'$	$L_c(\langle m, m' \rangle)$: $m, m' \to \langle m, m' \rangle$
Asymmetric	$L_{ad}(\{m\}^p_K)$: $\{m\}^p_K, K^{-1} \to m$	$L_c(\{m\}^p_K)$: $m, K \to \{m\}^p_K$
Symmetric	$L_{sd}(\{m\}^s_{m'})$: $\{m\}^s_{m'}, m' \to m$	$L_c(\{m\}^s_{m'})$: $m, m' \to \{m\}^s_{m'}$
Guess	$L_{od}(m)$: $E \to m$ with m subterm of E and E normalized.	$L_{oc}(m)$: $E \to m$ with E, m normalized and such that every proper subterm of $m$is a subterm of E.

Table 1: Intruder Rules w.r.t a normalization function.

Condition 1. in the above definition is not a real restriction since due to Lemma 3, the transformation performed by a protocol rule and its normalized variant coincide. Condition 2. guarantees that when with S_i an output is produced, all variables in S_i are "bounded" already, i.e., the substitution of these variables is determined. Otherwise, the output of a protocol rule would be arbitrary, since unbounded variables could be mapped to any message. Condition 3. guarantees that the bounding of variables is deterministic. For instance, we disallow a protocol rule of the form $Exp(a, x \cdot y) \Rightarrow Exp(a, x \cdot y \cdot b)$ in case neither x nor y is bounded, i.e., the substitution of neither x nor y is determined by a previous application of a rule. Otherwise, when say the principal receives $Exp(a, a \cdot b)$, there exist different ways of matching this term with $Exp(a, x \cdot y)$. We note that Condition 3. is a sufficient (not necessarily necessary) condition to guarantee that a rule can be carried out deterministically.

2.3 The Intruder Model and Attacks

Our intruder model follows the Dolev-Yao intruder [7]. That is, the intruder has complete control over the network and he can derive new messages from his initial knowledge and the messages received from honest principals during protocol runs. To derive a new message, the intruder can compose and decompose, encrypt and decrypt messages, in case he knows the key. What distinguishes the intruder we consider here from the standard Dolev-Yao intruder is that we equip the intruder with guess rules which provide him with additional capabilities of deriving messages. In Section 2.4, we consider guess rules that satisfies certain conditions. We will call these rules oracle rules.

The intruder derives new messages from a given (finite) set of message by applying intruder rules. An *intruder rule* (or *t-rule*) L is of the form $S \to t$, where S is a finite set of *standard* messages and t is a *standard* message. Given a finite set E of standard messages, the rule L *can be applied* to E if $S \subseteq E$. We define the *step relation* $\to_L$ induced by L as a binary relation on (finite) sets of standard messages. For every finite set of standard messages E we have $E \to_L E, t$ (recall that E, t stands for $E \cup \{t\}$) if L is a t-rule and L can be applied to E. If $\mathcal{L}$ denotes a (finite or infinite) set of intruder rules, then $\to_{\mathcal{L}}$ denotes the union $\bigcup_{L \in \mathcal{L}} \to_L$ of the step relations $\to_L$ with $L \in \mathcal{L}$. With $\to^*_{\mathcal{L}}$ we denote the reflexive and transitive closure of $\to_{\mathcal{L}}$.

The set of intruder rules we consider in this paper is depicted in Table 1. In this table, m, m' denote arbitrary standard messages, K is an element of $\mathcal{K}$, and E is a finite set of standard messages.

We emphasize that the notion of *intruder rule* will always refer to the rules listed in Table 1. For now, there may be any set of guess rules of the kind shown in Table 1, later we will consider certain classes of guess rules, namely oracle rules.

The intruder rules are denoted as shown in Table 1. With $L_{od}(m)$ and $L_{oc}(m)$ we denote (finite or infinite) sets of guess rules. For uniformity, we therefore consider $L_{p1}(\langle m, m' \rangle), \ldots, L_{sd}(\{m\}^s_{m'})$ and $L_c(\langle m, m' \rangle), \ldots, L_c(\{m\}^s_{m'})$ as singletons. Note that, even if there are no guess rules, the

number of decomposition and composition rules is always infinite since there are infinitely many messages m, m'.

We further group the intruder rules as follows. In the following, t ranges over all standard messages.

- $L_d(t) := L_{p1}(t) \cup L_{p2}(t) \cup L_{ad}(t) \cup L_{sd}(t)$. In case, for instance, $L_{p1}(t)$ is not defined, i.e., the head symbol of t is not a pair, then $L_{p1}(t) = \emptyset$; analogously for the other rule sets,
- $L_d := \bigcup_t L_d(t)$, $L_c := \bigcup_t L_c(t)$,
- $L_{od} := \bigcup_t L_{od}(t)$, $L_{oc} := \bigcup_t L_{oc}(t)$,
- $L_o(t) := L_{oc}(t) \cup L_{od}(t)$, $L_o := L_{oc} \cup L_{od}$,
- $\mathcal{L}_d(t)$ is the set of all decomposition t-rules in Table 1, i.e., all t-rule in the left column of the table,
- $\mathcal{L}_d := \bigcup_t \mathcal{L}_d(t)$,
- $\mathcal{L}_c(t)$ is the set of all composition t-rules in Table 1.
- $\mathcal{L}_c := \bigcup_t \mathcal{L}_c(t)$.
- $\mathcal{L} := \mathcal{L}_d \cup \mathcal{L}_c$.

Note that $\mathcal{L}$ denotes the (infinite) set of all intruder rules we consider here. The set of messages the intruder can derive from a (finite) set E of messages is:

$$\mathit{forge}(E) := \bigcup \{E' \mid E \rightarrow_{\mathcal{L}}^* E'\}.$$

From the definition of intruder rules in Table 1 it immediately follows:

Lemma 4 *If E is a normalized set of messages, then $\mathit{forge}(E)$ is normalized.*

The lemma says that if an intruder only sees normalized messages, then he only creates normalized messages. Intruders should be modeled in such a way that they cannot distinguish between equivalent messages. In what follows we always assume that the intruder's knowledge consists of a set of normalized messages, where every single normalized message in this set can be seen as a representative of its equivalence class.

We are now prepared to define attacks. In an attack on a protocol P, the intruder (nondeterministically) chooses some execution order for P and then tries to produce input messages for the protocol rules. These input messages are derived from the intruder's initial knowledge and the output messages produced by executing the protocol rules. The aim of the intruder is to derive the message secret. If different sessions of a protocol running interleaved shall be analysed, then these sessions must be encoded into the protocol P. This is the standard approach when protocols are analysed w.r.t. a bounded number of sessions, see, for instance, [13].

Definition 2 *Let $P = (\{R'_j \Rightarrow S'_j \mid j \in \mathcal{I}\}, <_{\mathcal{I}}, S_0)$ be a protocol. Then an* attack *on P is a tuple (π, σ) where π is an execution ordering on P and σ is a normalized ground substitution of the variables occurring in P such that*

1. *$\ulcorner R_i\sigma \urcorner \in \mathit{forge}(\ulcorner S_0, S_1\sigma, ..., S_{i-1}\sigma \urcorner)$ for every $i \in \{1, \ldots, k\}$ where k is the size of π, $R_i := R'_{\pi^{-1}(i)}$, and $S_i := S'_{\pi^{-1}(i)}$, and*
2. *$\mathsf{secret} \in \mathit{forge}(\ulcorner S_0, S_1\sigma, ..., S_k\sigma \urcorner)$.*

Due to Lemma 3, it does not matter whether, in the above definition, σ is normalized or not. Also note that Lemma 4 implies: $\ulcorner forge(\ulcorner S_0, S_1\sigma, ..., S_{i-1}\sigma\urcorner)\urcorner = forge(\ulcorner S_0, S_1\sigma, ..., S_{i-1}\sigma\urcorner)$.

The decision problem we are interested in is the following set of protocols:

$$\text{INSECURE} := \{P \mid \text{ there exists an attack on } P\}.$$

Later we will consider minimal attacks.

Definition 3 *Let $P = (\{R_i \Rightarrow S_i, i \in \mathcal{I}\}, <_{\mathcal{I}}, S_0)$ be a protocol. An attack (π, σ) is* minimal *if $|\sigma|$ is minimal.*

Clearly, if there is an attack, there is a minimal attack. Note, however, that minimal attacks are not necessarily uniquely determined.

2.4 Oracle Rules

Oracle rules are guess rules which satisfy certain conditions. To define these rules, we first need some new notions.

A *derivation* D of length n, $n \geq 0$, is a sequence of steps of the form $E \rightarrow_{L_1} E, t_1 \rightarrow_{L_2} \cdots \rightarrow_{L_n} E, t_1, \ldots, t_n$ with a finite set of standard messages E, standard messages $t_1, \ldots, t_n$, intruder rules $L_i \in \mathcal{L}$, such that $E, t_1, \ldots, t_{i-1} \rightarrow_{L_i} E, t_1, \ldots, t_i$ and $t_i \notin E \cup \{t_1, \ldots, t_{i-1}\}$, for every $i \in \{1, \ldots, n\}$. The rule L_i is called the *ith rule* of D and the step $E, t_1, \ldots, t_{i-1} \rightarrow_{L_i} E, t_1, \ldots, t_i$ is called the *ith step* of D. We write $L \in D$ to say that $L \in \{L_1, \ldots, L_n\}$. If S is a set of intruder rules, then we write $S \notin D$ to say $S \cap \{L_1, \ldots, L_n\} = \emptyset$. The message t_n is called the *goal* of D.

We also need *well formed* derivations which are derivations where every message generated by an intermediate step either occurs in the goal or in the initial set of messages.

Definition 4 *Let $D = E \rightarrow_{L_1} \ldots \rightarrow_{L_n} E'$ be a derivation with goal t. Then, D is* well formed *if for every $L \in D$ and t' we have that $L \in \mathcal{L}_c(t')$ implies $t' \in \mathcal{S}(E, t)$, and $L \in \mathcal{L}_d(t')$ implies $t' \in \mathcal{S}(E)$.*

We can now define oracle rules. Condition 1. in the following definition will allow us to bound the length of derivations. The remaining conditions are later used to bound the size of the substitution σ in an attack. They allow to replace a subterm u in σ by a smaller message.

Definition 5 *Let $L_o = L_{oc} \cup L_{od}$ be a (finite or infinite) set of guess rules, where L_{oc} and L_{od} denote disjoint sets of composition and decomposition guess rules, respectively. Then, L_o is a* set of oracle rules *(w.r.t. $L_c \cup L_d$ as defined above) iff:*

1. *For every message t and finite set E, if $t \in forge(E)$, then there exists a well formed derivation from E with goal t.*
2. *If $F \rightarrow_{L_{oc}(t)} F, t$ and $F, t \rightarrow_{L_d(t)} F, t, a$, then there exists a derivation D from F with goal a such that $L_d(t) \notin D$.*
3. *For every finite set F of messages with $1 \in F$, if $F \setminus u \rightarrow_{\mathcal{L}_c(u)} F$, i.e., u can be composed from $F \setminus u$ in one step, then $F \rightarrow_{L_o(t)} F, t$ implies $\ulcorner t[u \leftarrow 1]\urcorner \in forge(\ulcorner F[u \leftarrow 1]\urcorner)$ for every message t.*

Input: protocol $P = (\{R_\iota \Rightarrow S_\iota,\ \iota \in \mathcal{I}\}, <_{\mathcal{I}}, S_0)$

1. Guess an execution ordering π for P. Let k, R_i, and S_i be defined as in Definition 2.
2. Guess a normalized ground substitution σ such that $|\{\sigma(x) \mid x \in \mathcal{V}(P)\}| \leq 4|P|_{ext} + 1$ and $||\sigma||_{exp} \leq p(||P||)$.
3. Test that $\ulcorner R_i\sigma \urcorner \in \mathit{forge}(\ulcorner \{S_j\sigma \mid j < i\} \cup \{S_0\} \urcorner)$ for every $i \in \{1, \ldots, k\}$.
4. Test $\mathsf{secret} \in \mathit{forge}(\ulcorner \{S_j\sigma \mid j < k+1\} \cup \{S_0\} \urcorner)$.
5. If each test is successful, then answer "yes", and otherwise, "no".

Figure 1: NP Decision Procedure for Insecurity

3 Main Theorem and the NP Decision Algorithm

We now state one of the main theorems of this paper. In Section 6, this theorem will allow us to show that INSECURE is in NP in presence of an intruder that can perform DH exponentiation.

The *oracle rule problem* is defined as follows:

$$\text{ORACELRULE} = \{(E, m) \mid E \rightarrow_{L_o} E, m\}$$

where we assume E and m to be given as DAG. We say that the set of oracle rules L_o allows *polynomial product exponent attacks* if for every minmal attack (π, σ) on a protocol P there exists σ' such that $\sigma' \approx \sigma$ (recall that this means that σ' and σ coincide modulo the product exponents), $(\pi, \ulcorner \sigma' \urcorner)$ is an attack on P, and $||\sigma'||_{exp}$ is polynomially bounded in $||P||$. Note that by Lemma 3 this implies that $||\ulcorner \sigma' \urcorner||_{exp}$ is polynomially bounded in $||P||$.

Theorem 1 *Let L_o be a set of oracle rules. If*

- ORACELRULE $\in$ *PTIME and*
- *L_o allows polynomial product exponent attacks,*

then INSECURE *is in NP.*

The NP decision procedure is given in Figure 1 where p denotes the polynomial bounding the size of product exponents. This polynomial exists due to the assumption that L_o allows polynomial product exponent attacks.

Clearly, the procedure is sound. To show completeness, one has to prove that if there exists an attack (π, σ) on P, then there is one with the size of σ bounded as in step 2. of the procedure. In Theorem 3 we will show that $|\{\sigma'(x) \mid x \in \mathcal{V}(P)\}| \leq 4|P|_{ext} + 1$ for every minimal attack (π, σ') on P. Using the assumption that polynomial product exponent attacks exist, we can conclude that there exists an attack (π, σ) where the size of σ is bounded as in step 2. of the procedure.

In Section 4, we show that step 3. and 4. in the procedure can be carried out in polynomial time. More precisely, provided that ORACELRULE can be decided in deterministic polynomial time, we show that the following problem, henceforth called *derivation problem*, can be solved in polynomial time in $||E, t||$:

$$\text{DERIVE} := \{(E, t) \mid t \in \mathit{forge}(E)\}$$

where E is a finite set of standard messages and t is a standard message, given as a DAG (see Theorem 2). In the procedure, E is the set $\ulcorner \{S_j\sigma \mid j < i\} \cup S_0 \urcorner$ for some $i \in \{1, \ldots k\}$ and t is

$\ulcorner R_i\sigma \urcorner$ or secret. From Corollary 1, it follows that $|E,t| \leq 5 \cdot |P|_{ext} + 1$. Since $||\sigma||_{exp}$ is polynomially bounded in $||P||$, we obtain that $||\{S_j\sigma \mid j < i\} \cup S_0 \cup \{R_i\sigma\}||_{exp}$ is polynomially bounded in $||P||$, and with Lemma 3 this shows that $||E,t||_{exp}$ is polynomially bounded in $||P||$. Consequently, the procedure depicted in Figure 1 is in fact an NP decision procedure for INSECURE.

In Section 6, we will equip the intruder with intruder rules that allow him to perform DH exonentiation. We show that i) these rules are oracle rules (Section 6.1), ii) ORACELRULE $\in$ PTIME (Section 6.2), and iii) these rules allow polynomial product exponent attacks (Section 6.3). Then, using Theorem 1 we conclude that INSECURE can be decided in non-deterministic polynomial time. In a similar way, in [4] we have equipped the intruder with the ability to perform exclusive or (XOR) and have shown that for this intruder INSECURE can be decided in non-deterministic polynomial time. However, the proofs for XOR and DH exponentiation differ in two respects. First, because of the richer algebraic properties of DH exponentiation compared to XOR, proving i) and ii) from above is much more involved. Second, for XOR, property iii) does not need to be shown since due to the nilpotence of the XOR operator, i.e., the identity $a \oplus a = 0$, the product exponents (if XOR is considered a product operator) can be restricted to be 0 or 1. Clearly, for the product operator considered here this is not the case, and one therefore needs to bound the product exponents in σ. This is the main new challenge compared to the proof for XOR.

4 Deciding the Derivation Problem

We show:

Theorem 2 DERIVE $\in$ *PTIME provided* ORACELRULE $\in$ *PTIME.*

To show this theorem, let $d_t(E)$ be the set consisting of the the messages $t' \in \mathcal{S}(E,t)$ that can be derived from E in one step. Using that the number of terms $t' \in \mathcal{S}(E,t)$ is linear in $||E,t||$ and that $E \rightarrow_{L_o} E,t$ can be checked in polynomial time it is easy to see that $d_t(E)$ can be computed in polynomial time in $||E,t||$. Now, if $t \in \mathit{forge}(E)$, then Definition 5 guarantees that there exists a well formed derivation $D = E \rightarrow_{L_1} E,t_1 \rightarrow \ldots \rightarrow_{L_r} E,t_1,..,t_r$, with $t_r = t$. In particular, $t_i \in \mathcal{S}(E,t)$ for every $i \in \{1,\ldots,k\}$. By definition of derivations, all t_i are different. It follows $r \leq |t,E|$. Moreover, with $d_t^0(E) := E$ and $d_t^{l+1}(E) := d_t(d_t^l(E))$ we have that $t \in d_t^{|E,t|}(E)$ iff $t \in \mathit{forge}(E)$. Since $d_t^{|E,t|}(E)$ can be computed in polynomial time, Theorem 2 follows.

5 Bounding the Number of Subterms in Attacks

We show that $|\{\sigma(x) \mid x \in \mathcal{V}(P)\}| \leq 4|P|_{ext} + 1$ for minimal attacks (π,σ).

In what follows, we assume that L_o is a set of oracle rules. If $t \in \mathit{forge}(E)$, we denote by $D_t(E)$ a well formed derivation from E with goal t (chosen arbitrarily among the possible ones). Note that there always exists such a derivation due to the definition of oralce rules.

In Lemma 13 we prove, using Lemma 5 to 12, that minimal attacks can always be constructed by linking subterms that are initially occurring in the problem specification. This will allow us to bound σ as desired (see Theorem 3).

Let $P = (\{R_j \Rightarrow S_j,\ j \in \mathcal{I}\}, <_{\mathcal{I}}, S_0)$ be a protocol and (π,σ) be an attack on P. Let k, R_i, and S_i be defined as in Definition 2. Recall that $\mathcal{S}(P)$ is the set of subterms of P, $\mathcal{A} \subseteq \mathcal{S}(P)$, and $\mathcal{V} = \mathcal{V}(P)$ is the set of variables occurring in the protocol.

Definition 6 *Let t and t' be two terms and θ a ground substitution. Then, t is a* θ-match *of t', denoted $t \sqsubseteq_\theta t'$, if t and t' are standard, t is not a variable, and $\ulcorner t\theta \urcorner = t'$.*

Lemma 5 *Let s be a standard term, t be a normalized term, and σ be a normalized substitution such that $s \in \mathcal{S}(\ulcorner t\sigma \urcorner)$ and $s \notin \mathcal{S}(x\sigma)$ for every $x \in \mathcal{V}(t)$. Then, there exists a standard subterm t' of t with $t' \sqsubseteq_\sigma s$.*

PROOF. If t is a term, let t' be a minimal subterm of t (w.r.t. subterm ordering) such that $s \in \mathcal{S}(\lceil t'\sigma \rceil)$. Obviously, such a minimal subterm exists.

We claim that $t' \sqsubseteq_\sigma s$, i.e., $\lceil t'\sigma \rceil = s$. To show this, we distinguish several cases. First, we note that by the assumption t' cannot be a variable:

- Assume that $t' = \langle u, v \rangle$. Thus, $\lceil t'\sigma \rceil = \langle \lceil u\sigma \rceil, \lceil v\sigma \rceil \rangle$. By definition of t', $s \notin \mathcal{S}(\lceil u\sigma \rceil)$ and $s \notin \mathcal{S}(\lceil v\sigma \rceil)$. Since $s \in \mathcal{S}(\lceil t'\sigma \rceil)$, we obtain $\lceil t'\sigma \rceil = s$. Similarly, one argues for $t' = \{u\}_v^s$ and $t' = \{u\}_v^p$.

- Assume that $t' = Exp(u, M)$ with $M = t_1^{z_1} \cdots t_p^{z_p}$. Note that the t_i are standard. By definition of t' we know that $s \notin \mathcal{S}(\lceil u\sigma \rceil)$ and $s \notin \mathcal{S}(\{\lceil t_i\sigma \rceil\})$. If $\lceil u\sigma \rceil$ is not of the form $Exp(\cdot, \cdot)$, then $\lceil t'\sigma \rceil = Exp(\lceil u\sigma \rceil, \lceil M\sigma \rceil)$ or $\lceil t'\sigma \rceil = \lceil u\sigma \rceil$. Using that s is standard, in both cases we can conclude $t' \sqsubseteq_\sigma s$. Otherwise, if $\lceil u\sigma \rceil = Exp(v, M')$, then $\lceil t'\sigma \rceil = Exp(v, \lceil M' \cdot M\sigma \rceil)$ or $\lceil t'\sigma \rceil = v$. We know that $s \notin \mathcal{S}(\{\lceil t_i\sigma \rceil\})$ and $s \notin \mathcal{S}(M')$. But then, using that s is standard, we have $s \notin \mathcal{S}(\lceil M' \cdot M\sigma \rceil)$. With $s \notin \mathcal{S}(v)$, in both cases we obtain $t' \sqsubseteq_\sigma s$.

- Assume that $t' = t_1^{z_1} \cdots t_p^{z_p}$, $p > 0$, with t_i standard. If $s \in \mathcal{S}(\lceil t'\sigma \rceil) \setminus \lceil t'\sigma \rceil$, then there exists i with $s \in \mathcal{S}(\lceil t_i\sigma \rceil)$, in contradiction to the definition of t'. Thus, $t' \sqsubseteq_\sigma s$.

□

We use this lemma to prove:

Lemma 6 *Let (π, σ) be an attack on P with σ normalized, $x \in \mathcal{V}(R_i)$ for some $i \in \{1, \ldots, k\}$, and $s \in \mathcal{S}(\sigma(x))$ standard. We assume that $s \neq \sigma(y)$ for every y in case $s = Exp(\cdot, \cdot)$. Then, there exists $j \leq i$ such that $s \in \mathcal{S}(\lceil R_j\sigma \rceil)$ or there exists $t \in \mathcal{S}(P)$ with $t \sqsubseteq_\sigma s$.*

PROOF. First, assume that $s = 1$. By definition of protocols, 1 belongs to the initail intruder knowledge of P, and thus, $1 \in \mathcal{S}(P)$ and $1 \sqsubseteq_\sigma s$.

Now, assume that $s \neq 1$ and that there exists $i \in \{1, \ldots, k\}$, $x \in \mathcal{V}(R_i)$, and $s \in \mathcal{S}(\sigma(x))$ standard. We also assume that for all $j \leq i$: $s \notin \mathcal{S}(\lceil R_j\sigma \rceil)$, since otherwise the lemma is proved. Define

$$j = \min\{i' \mid y \in \mathcal{V}(R_{i'}) \text{ and } s \in \mathcal{S}(\sigma(y)) \text{ standard}\}$$

and let y be a variable of R_j such that $s \in \mathcal{S}(\sigma(y))$ standard. Note that $j \leq i$, and thus, $s \notin \mathcal{S}(\lceil R_j\sigma \rceil)$.

Let t be a maximal subterm (w.r.t. subterm ordering) of R_j such that $y \in \mathcal{V}(t)$ and s is a subterm of $\lceil t\sigma \rceil$. (Since y is a possible candidate for t, such a maximal subterm exists.) We know that $t \neq R_j$. Let $r \in \mathcal{S}(R_j)$ be minimal such that t is a strict subterm of r. Then, since $s \notin S(\lceil r\sigma \rceil)$, r must be of the form $v_1^{z_1} \cdots v_p^{z_p}$ or $Exp(u, v_1^{z_1} \cdots v_p^{z_p})$ where u and the v_i are standard. We consider both cases:

1. If $r = v_1^{z_1} \cdots v_p^{z_p}$, then there exists k with $v_k = t$ and there exists $k' \neq k$ with $s \in \mathcal{S}(\lceil v_{k'}\sigma \rceil)$. (Here we use that $s \neq 1$). By Definition 1, (3) and minimality of j, we conclude that $\mathcal{V}(v_{k'}) \subseteq \mathcal{V}(R_1, \ldots, R_{j-1})$. Now, Lemma 5 implies that there exists $t_s \in \mathcal{S}(P)$ such that $t_s \sqsubseteq_\sigma s$.[1]

2. Assume that $r = Exp(u, M)$ and $t = u$. By Definition 1, (3) and minimality of j, $\mathcal{V}(M) \subseteq \mathcal{V}(R_1, \ldots, R_{j-1})$. Since $s \notin S(\lceil r\sigma \rceil)$ we can conclude that $\lceil u\sigma \rceil = Exp(v, M')$. We distinguish different cases:

[1] Strictly speaking, we do not need to consider this case, since r cannot be a non-standard term. When allowing products outside of exponents, one has to consider this case however.

First, assume that $s = \ulcorner u\sigma \urcorner$. In particular, $s = Exp(\cdot,\cdot)$. Thus, by assumption, $s \neq \sigma(z)$ for every variable z. Consequently, u is not a variable. But then we are done since $u \in \mathcal{S}(P)$ and $u \sqsubseteq_\sigma s$.

Now, assume that $s \neq \ulcorner u\sigma \urcorner$. Then, $s \in \mathcal{S}(v)$ or $s \in \mathcal{S}(M')$. The former case cannot occur since this would imply $s \in \mathcal{S}(\ulcorner r\sigma \urcorner)$ because $\ulcorner r\sigma \urcorner = Exp(v, \ulcorner M' \cdot M\sigma \urcorner)$ or $\ulcorner r\sigma \urcorner = v$. Thus, $s \in \mathcal{S}(M')$. Since $s \notin \mathcal{S}(\ulcorner r\sigma \urcorner)$, we can conclude that $s \notin \mathcal{S}(\ulcorner M' \cdot M\sigma \urcorner)$. Together with $s \in \mathcal{S}(M')$ it follows that $s \in \mathcal{S}(\ulcorner M\sigma \urcorner)$. (Here we use again that $s \neq 1$.) Now, given $\mathcal{V}(M) \subseteq \mathcal{V}(R_1, \ldots, R_{j-1})$ and by minimality of j, according to Lemma 5 there exists $t_s \in \mathcal{S}(P)$ such that $t_s \sqsubseteq_\sigma s$.

3. Assume that $r = Exp(u, M)$ with $M = t^z \cdot M'$. By Definition 1, (3) and minimality of j, $\mathcal{V}(u) \subseteq \mathcal{V}(R_1, \ldots, R_{j-1})$. If $s \in \mathcal{S}(\ulcorner u\sigma \urcorner)$, Lemma 5 implies that there exists $t_s \in \mathcal{S}(P)$ such that $t_s \sqsubseteq_\sigma s$. Assume that $s \notin \mathcal{S}(\ulcorner u\sigma \urcorner)$. Then, since $s \notin \mathcal{S}(\ulcorner r\sigma \urcorner)$, $s \notin \mathcal{S}(\ulcorner M\sigma \urcorner)$. We can proceed as in 1.

□

The proof of the following lemma is trivial.

Lemma 7 *For every normalized finite set E of messages, message t, and t-rule L, if $E \to_L E, t$ then all proper subterms of t are subterms of E.*

PROOF. For $L \in L_{od} \cup L_{oc}$ use the definition of decomposition and composition guess rules, respectively. For $L \in L_d \cup L_c$ this is obvious. □

The next lemma states that if a term t' is a subterm of a term t and t can be derived from a set E but t' is not a subterm of E, then t' can be derived from E and the last step of the derivation is a composition rule.

Lemma 8 *Assume that $t' \in \mathcal{S}(t) \setminus \mathcal{S}(E)$ and $t \in forge(E)$, then $t' \in forge(E)$ and there exists a (well formed) derivation from E with goal t' ending with a composition rule.*

PROOF. Let $D = E_0 \to_{L_1} E_1 \cdots \to_{L_n} E_n$ be a derivation of t from $E_0 = E$. Then, there exists a least $i \neq 0$ such that $t' \in \mathcal{S}(E_i)$ since t' is a subterm of E_n. Assume that L_i is an s-rule for some s. Then, t' is a subterm of s. If t' is a proper subterm of s, Lemma 7 implies that t' is a subterm of E_{i-1} in contradiction to the minimality of i. Thus, $t' = s$ and therefore, $t' \in forge(E)$. By the definition of oracle rules, there exists a well formed derivation D' of t'. If the last step in this derivation is a decomposition rule, then this implies $t' \in \mathcal{S}(E)$ in contradiction to the assumption. Thus, the last step of D' is a composition rule. □

The following lemma is used in the proof of Lemma 10. First, we need some notation: If $D_1 = E_1 \to \ldots \to F_1$ and $D_2 = E_2 \to \ldots \to F_2$ are two derivations such that $E_2 \subseteq F_1$, then $D = D_1.D_2$ is defined as the *concatenation* of the steps of D_1 and the ones in D_2. In addition, to obtain a derivation, we remove in D the steps from D_2 that generate terms already present in F_1.

Lemma 9 *Let $t \in forge(E)$ and $\gamma \in forge(E)$ be given with a derivation D_γ from E ending with an application of a rule in $\mathcal{L}_c$. Then, there is a derivation D' from E with goal t satisfying $L_d(\gamma) \notin D'$.*

PROOF. By definition of a derivation, $L_d(\gamma) \notin D_\gamma$. Let D be D_γ without its last rule, i.e., D_γ is D followed by some $L \in \mathcal{L}_c$. Define $D'' = D.Deriv_t(E) = D.D''' - D'''$ is obtained from $Deriv_t(E)$ by removing redundant steps. Note that D'' is a derivation with goal t. We distinguish two cases:

- Assume $L = L_c(\gamma)$. Then $L_d(\gamma) \notin D''$ since the (two) direct subterms of γ are created in D, and thus, $L_d(\gamma) \notin D''$. In other words, $D' = D''$ is the derivation we are looking for.

- Assume $L = L_{oc}(\gamma)$. Then, if $L_d(\gamma) \notin D''$, nothing is to show. Otherwise, let F_1 be the final set of messages of D. Now, Definition 5, (2) implies that every step in D''' of the form $F_1, F_2, \gamma \rightarrow_{L_d(\gamma)} F_1, F_2, \gamma, \beta$ can be replaced by a derivation from F_1, F_2 with goal β that does not contain rules from $L_d(\gamma)$. Replacing steps in this way and then removing redundant steps yields the derivation D' we are looking for.

□

The subsequent lemma will allow us to replace certain subterms occurring in a substitution of an attack by smaller terms. Note that from the assumption made in this lemma it follows that s can be derived from E such that the last rule is a composition rule. This allows to replace s by a smaller term since when deriving t, decomposing s will not be necessary.

Lemma 10 *Let E and F be two sets of normalized messages such that $1 \in E \cup F$. Let $t \in \mathit{forge}(E, F)$ and $s \in \mathit{forge}(E)$ such that s is non atomic and $s \notin \mathcal{S}(E)$. Finally, let δ be the replacement $[s \leftarrow 1]$. Then, $\ulcorner t\delta \urcorner \in \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$.*

Proof. By Lemma 8 there exists a well formed derivation D_s from E with goal s such that the last step is a composition rule. By Lemma 9, there exists a derivation D_t from E, F with goal t such that $L_d(s) \notin D_t$. Assume that $D_t = E, F \rightarrow_{L_1} E, F, t_1 \rightarrow_{L_2} E, F, t_2 \cdots \rightarrow_{L_n} E, F, t_1 \ldots, t_n$. We show $\ulcorner t_i\delta \urcorner \in \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$ by induction on $i \leq n$ where t_0 is some term in E, F. Induction base: If $t_0 \in E \cup F$, then clearly $\ulcorner t_0\delta \urcorner \in \ulcorner E\delta \cup F\delta \urcorner$. Induction step: We distinguish several cases.

- If $L_i = L_c(\langle a, b \rangle)$, then either $t_i = s$, and thus, $t_i\delta = 1 \in \mathit{forge}(E\delta, F\delta)$, or $t_i\delta = \langle a\delta, b\delta \rangle \in \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$ since by induction we have $\{a\delta, b\delta\} \subseteq \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$. Analogously, the terms $\{a\}_b^s$ and $\{a\}_K^p$ are treated.

- If $L_i = L_{p1}(\langle t_i, a \rangle)$, then $s \neq \langle t_i, a \rangle$ since $L_i \notin L_d(s)$. Therefore, $\langle t_i, a \rangle \delta = \langle t_i\delta, a\delta \rangle$. By induction, $\langle t_i, a \rangle \delta \in \mathit{forge}(E\delta, F\delta)$, and thus, $t_i\delta \in \mathit{forge}(E\delta, F\delta)$. Analogously, the cases for L_{p2}, L_{sd}, and L_{ad} are shown.

- If $L_i \in L_o$, then we use Definition 5, (3). Let E' be the set of messages obtained in D_s before the last step is applied. Then, $E' \setminus s \rightarrow_{\mathcal{L}_c(s)} E', s$. Also, $E, F, t_1, \ldots, t_{i-1} \rightarrow_{L_i} E, F, t_1, \ldots, t_{i-1}, t_i$. In particular, $E', E, F, t_1, \ldots, t_{i-1} \rightarrow_{L_i} E', E, F, t_1, \ldots, t_{i-1}, t_i$. By Definition 5, (3) we obtain $\ulcorner t_i\delta \urcorner \in \mathit{forge}(\ulcorner E'\delta, E\delta, F\delta, t_1\delta, \ldots, t_{i-1}\delta \urcorner)$. We know that $E'\delta = E'$, $E\delta = E$, and all terms in E' can be derived from E. Also, E and E' are normalized. By induction, $\ulcorner t_1\delta \urcorner, \ldots, \ulcorner t_{i-1}\delta \urcorner \in \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$. Thus, $\mathit{forge}(\ulcorner E'\delta, E\delta, F\delta, t_1\delta, \ldots, t_{i-1}\delta \urcorner) \subseteq \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$, and therefore, $\ulcorner t_i\delta \urcorner \in \mathit{forge}(\ulcorner E\delta, F\delta \urcorner)$

For $i = n$, this gives us $t\delta \in \mathit{forge}(E\delta, F\delta)$. □

The following lemma is used in the proof of Lemma 12.

Lemma 11 *Let u be a normalized term, M, M' be two products such that for all $t \in \mathcal{F}(M)$ $(t \in \mathcal{F}(M'))$, t is normalized. Let s be a standard normalized term and δ the replacement $[s \leftarrow 1]$. Then:*

1. $\ulcorner (M \cdot M')\delta \urcorner = \urcorner\urcorner M \cdot M \ulcorner\ulcorner \delta \urcorner$, *in particular,* $\ulcorner M\delta \urcorner = \urcorner\urcorner M \ulcorner\ulcorner \delta \urcorner$.
2. $\ulcorner \mathit{Exp}(u, M)\delta \urcorner = \urcorner\urcorner \mathit{Exp}(u, M) \ulcorner\ulcorner \delta \urcorner$ *if* $s \neq \ulcorner \mathit{Exp}(u, M) \urcorner$ *and, in case s is of the form $\mathit{Exp}(\cdot, \cdot)$, also* $s \neq u$.

PROOF. Statement 1 is straightforward. We prove 2. given the restrictions on s.

First, assume that u is *not* of the form $Exp(\cdot,\cdot)$. Then i) $\ulcorner Exp(u,M)\urcorner = u$, and thus, $\ulcorner M\urcorner = 1$, or ii) $\ulcorner Exp(u,M)\urcorner = Exp(u,\ulcorner M\urcorner)$ and $\ulcorner M\urcorner \neq 1$. We consider both cases.

In case i) we obtain that $\ulcorner M\urcorner\delta = 1$. By 1. we know that $\ulcorner M\delta\urcorner = \urcorner\urcorner M\urcorner\delta\urcorner (= 1)$. Thus,

$$\begin{aligned}\ulcorner Exp(u,M)\delta\urcorner &= \ulcorner Exp(u\delta, M\delta)\urcorner \quad (*)\\ &= \ulcorner Exp(u\delta, \ulcorner M\delta\urcorner)\urcorner\\ &= \ulcorner u\delta\urcorner\\ &= \urcorner\urcorner Exp(u,M)\urcorner\delta\urcorner\end{aligned}$$

where in (*) we use that $Exp(u,M) \neq s$ (otherwise $\ulcorner Exp(u,M)\urcorner = s$ since s is normalized).

In case ii), we obtain

$$\begin{aligned}\urcorner\urcorner Exp(u,M)\urcorner\delta\urcorner &= \ulcorner Exp(u,\ulcorner M\urcorner)\delta\urcorner\\ &= \ulcorner Exp(u\delta, \ulcorner M\urcorner\delta)\urcorner \quad (*)\\ &= \ulcorner Exp(u\delta, \urcorner\urcorner M\urcorner\delta\urcorner)\urcorner\\ &= \ulcorner Exp(u\delta, \ulcorner M\delta\urcorner)\urcorner \quad (**)\\ &= \ulcorner Exp(u\delta, M\delta)\urcorner\\ &= \ulcorner Exp(u,M)\delta\urcorner \quad (***)\end{aligned}$$

where in (*) we use that $Exp(u,\ulcorner M\urcorner) \neq s$ (otherwise $\ulcorner Exp(u,M)\urcorner = \ulcorner Exp(u,\ulcorner M\urcorner)\urcorner = s$), in (**) we use 1., and in (***) that $Exp(u,M) \neq s$. Note that both in i) and ii) the fact that $u \neq s$ in case s is of the form $Exp(\cdot,\cdot)$ is not needed.

Now, assume that $u = Exp(v,M')$ for some v and M'. Then, we obtain

$$\begin{aligned}\urcorner\urcorner Exp(u,M)\urcorner\delta\urcorner &= \urcorner\urcorner Exp(v,M'\cdot M)\urcorner\delta\urcorner\\ &= \ulcorner Exp(v,M'\cdot M)\delta\urcorner \quad (*)\\ &= \ulcorner Exp(v\delta,(M'\delta\cdot M\delta)\urcorner \quad (**)\\ &= \ulcorner Exp(Exp(v\delta,M'\delta),M\delta)\urcorner\\ &= \ulcorner Exp(u\delta,M\delta)\urcorner \quad (***)\\ &= \ulcorner Exp(u,M)\delta\urcorner \quad (****)\end{aligned}$$

where (*) is obtained just as in the first case. We use that v is not of the form $Exp(\cdot,\cdot)$ and that $Exp(v,M'\cdot M) \neq s$ (otherwise $\ulcorner Exp(v,M'\cdot M)\urcorner = \ulcorner Exp(u,M)\urcorner = s$). Recall that the first case works even without the assumption that $v \neq s$. In (**), again we use that $Exp(v,M'\cdot M) \neq s$. In (***), we use that $u \neq s$ (not that otherwise $u = s$ and s is of the form $Exp(\cdot,\cdot)$). Finally, (****) uses that $Exp(u,M) \neq s$. □

We note that 2. in the previous lemma does not hold without the restrictions on s. The following example shows the problem in case $s = \ulcorner Exp(u,M)\urcorner$: Assume that $s = Exp(a,b)$, $u = a$ and $M = b\cdot c\cdot c^{-1}$. Then, $s = \ulcorner Exp(u,M)\delta\urcorner \neq \urcorner\urcorner Exp(u,M)\urcorner\delta\urcorner = 1$. The next example illustrates why $s \neq u$ is necessary: Define $s = u = Exp(a,b)$ and $M = c$. Then, $Exp(1,c) = \ulcorner Exp(u,M)\delta\urcorner \neq \urcorner\urcorner Exp(u,M)\urcorner\delta\urcorner = Exp(a,b\cdot c)$.

The next lemma will be used to remove one application of the normalization function.

Lemma 12 *Let σ be a normalized ground substitution, E a set of normalized terms, s a normalized standard non atomic term, and δ the replacement $[s \leftarrow 1]$. We assume that if s is of the form $Exp(\cdot,\cdot)$, then $s \neq \sigma(x)$ for all $x \in \mathcal{V}$. Let $\sigma' = \sigma\delta$. If there is no standard subterm t of E such that $t \sqsubseteq_\sigma s$, then $\ulcorner E\sigma\urcorner = \urcorner\urcorner E\sigma\urcorner\delta\urcorner$.*

PROOF. Since there is no standard subterm t' of E such that $t' \sqsubseteq_\sigma s$, we have $(E\sigma)\delta = E(\sigma\delta)$ and therefore $\ulcorner E\sigma\urcorner = \ulcorner(E\sigma)\delta\urcorner$. Let us prove, by induction on the structure of terms, that for all $t \in \mathcal{S}(E)$, we have $\ulcorner t\sigma\urcorner = \urcorner\urcorner t\sigma\urcorner\delta\urcorner$. This will conclude the proof of the lemma. In what follows, assume that $t \in \mathcal{S}(E)$.

- If $t \in \mathcal{A}$, then $t \neq s$ by assumption. Thus, $\urcorner t\sigma\urcorner\delta\urcorner = t = \ulcorner t\sigma\urcorner$.
- If $t \in \mathcal{V}$, then $\ulcorner t\sigma\urcorner = t\sigma$, and therefore, $\ulcorner t\sigma\urcorner = \ulcorner(t\sigma)\delta\urcorner = \urcorner t\sigma\urcorner\delta\urcorner$.
- If $t = \langle v, w\rangle$, we have $s \neq \langle \ulcorner v\sigma\urcorner, \ulcorner w\sigma\urcorner\rangle = \ulcorner t\sigma\urcorner$ since otherwise $t \sqsubseteq_\sigma s$, and $\ulcorner t\sigma\urcorner = \langle \ulcorner v\sigma\urcorner, \ulcorner w\sigma\urcorner\rangle$. By induction, this gives $\ulcorner t\sigma\urcorner = \langle \urcorner v\sigma\urcorner\delta\urcorner, \urcorner w\sigma\urcorner\delta\urcorner\rangle$, and therefore, $\ulcorner t\sigma\urcorner = \urcorner\langle v\sigma, w\sigma\rangle\urcorner\delta\urcorner = \urcorner t\sigma\urcorner\delta\urcorner$ since $s \neq \ulcorner t\sigma\urcorner$. The cases $t = \{u\}^s_v$ and $t = \{u\}^p_K$ are similar.
- If $t = v_1^{z_1} \cdots v_p^{z_p}$, we have:

$$\begin{aligned}
\ulcorner t\sigma\urcorner &= \ulcorner (v_1\sigma')^{z_1} \cdots (v_p\sigma')^{z_p}\urcorner \\
&= \urcorner v_1\sigma\urcorner^{z_1} \cdots \ulcorner v_p\sigma\urcorner^{z_p}\urcorner \\
&= \urcorner\urcorner v_1\sigma\urcorner\delta\urcorner^{z_1} \cdots \urcorner v_p\sigma\urcorner\delta\urcorner^{z_p}\urcorner && (*) \\
&= \ulcorner (\ulcorner v_1\sigma\urcorner\delta)^{z_1} \cdots (\ulcorner v_p\sigma\urcorner\delta)^{z_p}\urcorner \\
&= \ulcorner (\ulcorner v_1\sigma\urcorner^{z_1} \cdots \ulcorner v_p\sigma\urcorner^{z_p})\delta\urcorner && (**) \\
&= \urcorner\urcorner v_1\sigma\urcorner^{z_1} \cdots \ulcorner v_p\sigma\urcorner^{z_p}\urcorner\delta\urcorner && (***) \\
&= \urcorner t\sigma\urcorner\delta\urcorner
\end{aligned}$$

  where (*) is by induction. For (**) we use that s is a normalized standard term. The step (***) is an immediate consequence of Lemma 11.
- If $t = \mathit{Exp}(u, M)$, we have:

$$\begin{aligned}
\ulcorner t\sigma\urcorner &= \ulcorner \mathit{Exp}(u\sigma', M\sigma')\urcorner \\
&= \ulcorner \mathit{Exp}(\ulcorner u\sigma\urcorner, \ulcorner M\sigma\urcorner)\urcorner \\
&= \ulcorner \mathit{Exp}(\urcorner u\sigma\urcorner\delta\urcorner, \urcorner M\sigma\urcorner\delta\urcorner)\urcorner && (*) \\
&= \ulcorner \mathit{Exp}(\ulcorner u\sigma\urcorner\delta, \ulcorner M\sigma\urcorner\delta)\urcorner \\
&= \ulcorner \mathit{Exp}(\ulcorner u\sigma\urcorner, \ulcorner M\sigma\urcorner)\delta\urcorner && (**) \\
&= \urcorner \mathit{Exp}(\ulcorner u\sigma\urcorner, \ulcorner M\sigma\urcorner)\urcorner\delta\urcorner && (***)
\end{aligned}$$

  where (*) is by induction. For (**) we use that $\mathit{Exp}(\ulcorner u\sigma\urcorner, \ulcorner M\sigma\urcorner) \neq s$. Otherwise, we have $\ulcorner \mathit{Exp}(\ulcorner u\sigma\urcorner, \ulcorner M\sigma\urcorner)\urcorner = \ulcorner t\sigma\urcorner = s$, in contradiction to the assumption that $t \not\sqsubseteq_\sigma s$. Finally, (***) is by Lemma 11. Note that the preconditions for this lemma are satisfied: We have that $\ulcorner \mathit{Exp}(\ulcorner u\sigma\urcorner, \ulcorner M\sigma\urcorner)\urcorner = \ulcorner t\sigma\urcorner \neq s$. Also, $\ulcorner u\sigma\urcorner \neq s$ in case s is of the form $\mathit{Exp}(\cdot,\cdot)$: If u is not a variable, then $u \in \mathcal{S}(P)$ and $u \sqsubseteq_\sigma s$, in contradiction to the assumptions. If u is a variable and s is of the form $\mathit{Exp}(\cdot,\cdot)$, then by assumption $\ulcorner \sigma(u)\urcorner = \sigma(u) \neq s$.

□

The main lemma, which shows that a substitution of a minimal attack can be build up from subterms of terms occurring in P, is proved next.

Lemma 13 *Let (π, σ) be a minimal attack, x a variable, and v_x a factor of $\sigma(x)$ such that if v_x is of the form $\mathit{Exp}(\cdot,\cdot)$, then $v_x \neq \sigma(y)$ for all y. Then, there exists $t \in \mathcal{S}(P)$ such that $t \sqsubseteq_\sigma v_x$.*

Proof. Let (π, σ), x, and v_x be as above. Assume (*): For every t, $t \sqsubseteq_\sigma v_x$ implies $t \notin \mathcal{S}(P)$. We will lead this to a contradiction. Since $\mathcal{A} \subseteq \mathcal{S}(P)$, we have $v_x \notin \mathcal{A}$. Also, note that by definition of factors, v_x is standard. By Lemma 6 and (*), there exists j such that $v_x \in \mathcal{S}(\ulcorner R_j\sigma\urcorner)$. Let N_x be minimal among the possible j. If $v_x \in \mathcal{S}(\ulcorner S_i\sigma\urcorner)$ for some i, (*) together with Lemma 5 implies that there exists $y \in \mathcal{V}(S_i)$ with $v_x \in \mathcal{S}(\sigma(y))$. Then, by Definition 1, (2) there exists $R_{i'}, i' \leq i$ such that $y \in \mathcal{V}(R_{i'})$. Thus, Lemma 6 and (*) imply that there exists $j \leq i$ with $v_x \in \mathcal{S}(\ulcorner R_j\sigma\urcorner)$. Note also that $v_x \notin \mathcal{S}(S_0)$ since otherwise $v_x \in \mathcal{S}(P)$. Now, the minimality of N_x yields $i \geq N_x$. Let $E_j = \ulcorner S_0\sigma, \ldots, S_{j-1}\sigma\urcorner$.

Summarizing, we have: v_x is not a subterm of E_{N_x}, and v_x is a subterm of $\ulcorner R_{N_x}\sigma\urcorner$. Thus, by Lemma 8, $v_x \in \mathit{forge}(E_{N_x})$ and v_x is non atomic.

Let us define the replacement $\delta = [v_x \leftarrow 1]$. Since (π, σ) is an attack, we have for all $1 \leq j \leq k+1$ and $R_{k+1} = \mathsf{secret}$:

$$\ulcorner R_j\sigma\urcorner \in \mathit{forge}(E_j)$$

We distinguish two cases:

- Assume $j < N_x$. Then, by minimality of N_x, v_x is neither a subterm of $\ulcorner R_j\sigma\urcorner$ nor a subterm of E_j. Hence, with $\ulcorner R_j\sigma\urcorner \in \mathit{forge}(E_j)$ it follows $\ulcorner\ulcorner R_j\sigma\urcorner\delta\urcorner \in \mathit{forge}(\ulcorner E_j\delta\urcorner)$.
- Assume $j \geq N_x$. With $t = \ulcorner R_j\sigma\urcorner$, $s = v_x$, $E = E_{N_x}$, and $F = E_j$, Lemma 10 implies $\ulcorner\ulcorner R_j\sigma\urcorner\delta\urcorner \in \mathit{forge}(\ulcorner E_j\delta\urcorner)$.

Thus, $\ulcorner\ulcorner R_j\sigma\urcorner\delta\urcorner \in \mathit{forge}(\ulcorner E_j\delta\urcorner)$ in both cases. Now, with $E = \{S_0, \ldots, S_{j-1}\}$ and $E = \{R_j\}$, respectively, (*) and Lemma 12 imply for all j:

$$\ulcorner R_j\sigma\urcorner \in \mathit{forge}(\ulcorner S_0\sigma'\urcorner, \ldots, \ulcorner S_{j-1}\sigma'\urcorner)$$

where $\sigma' = \sigma\delta$. Hence, (π, σ') is an attack. (Note that the conditions for applying Lemma 12 are satisfied.) But since σ' is obtained from σ by replacing v_x by a strictly smaller message, namely 1, we obtain $|\sigma'| < |\sigma|$, a contradiction to the assumption that (π, σ) is a minimal attack. □

We now use Lemma 13 to bound the size of every $\sigma(x)$. We first need to introduce the notion of an F-normal form.

Let F be a finite number of normalized standard messages. Let $V_F = \{x_s \mid s \in F\}$ where x_s is a new variable for every $s \in F$. Let $\sigma'(x_s) = s$. For a (not necessarily normalized) standard or non-standard message u we recursively define its *F-normal form* t_u. Although t_u depends on F, we simply write t_u. We define $t'_u = x_{\ulcorner u\urcorner}$ if $\ulcorner u\urcorner \in F$ and $t'_u = t_u$ otherwise. While constructing t_u, we show that

A) $t_u\sigma' = \ulcorner u\urcorner$ (and thus, $t'_u\sigma' = \ulcorner u\urcorner$).

B) $t_u = t_{\ulcorner u\urcorner}$. Note that from this and A), it immediately follows that $t'_u = t'_{\ulcorner u\urcorner}$, $t_{t_u\sigma'} = t_u$, and $t'_{t'_u\sigma'} = t'_u$.

C) $t'_{s\sigma'} = s$ for all $s \in \mathcal{S}_{ext}(t_u)$ with $t'_{s\sigma'} = t_{s\sigma'}$.

D) $t_u = \mathit{Exp}(\cdot,\cdot)$ if $\ulcorner u\urcorner = \mathit{Exp}(\cdot,\cdot)$ and u is standard.

Note that B) implies C) for the case that $s = t_u$. Also, if $s \in V_F$, then obviously $t'_{s\sigma'} = s$. Consequently, to show C) is suffices to only consider $s \in \mathcal{S}_{ext}(t_u) \setminus (\{t_u\} \cup V_F)$ in case B) has been proved. We now define t_u inductively:

1. If $\ulcorner u\urcorner \in F$ and $\ulcorner u\urcorner$ is an atom, a pair, or encryption, then $t_u = x_{\ulcorner u\urcorner}$.

   Obviously, A) to D) are satisfied.

2. If $u \in \mathcal{A}$, then $t_u = u$.

   Obviously, A) to D) are satisfied.

3. If $u = \langle u_1, u_2\rangle$, then $t_u = \langle t'_{u_1}, t'_{u_2}\rangle$.

   Induction yields A) and B). Obviously, D) is satisfied. We now show C). Let $s \in \mathcal{S}_{ext}(t_u) \setminus (\{t_u\} \cup V_F)$ with $t'_{s\sigma'} = t_{s\sigma}$. Then, $s \in \mathcal{S}_{ext}(t'_{u_i})$ and since $s \notin V_F$, we have that $t'_{u_i} \notin V_F$, and thus, $t'_{u_i} = t_{u_i}$. By induction and $s \in \mathcal{S}_{ext}(t_{u_i})$, we have $t'_{s\sigma'} = s$. Encryption is dealt with analogously.

4. If $u = u_1^{z_1} \cdots u_n^{z_n}$, where the u_i are standard, let $C_1, \ldots, C_k \subseteq \{u_1, \ldots, u_n\}$ be the equivalence classes over $\{u_1, \ldots, u_n\}$ such that $\ulcorner v \urcorner = \ulcorner v' \urcorner$ for every i and $v, v' \in C_i$ and $\ulcorner v \urcorner \neq \ulcorner v' \urcorner$ for every $i \neq j$, $v \in C_i$, and $v' \in C_j$. W.l.o.g. assume that $v\sigma' = 1$ for every $v \in C_1$. (This class may be empty.) Note that $\ulcorner u_i \urcorner$ are standard messages since products only occur inside of exponents, which remains true when normalizing messages). Let $s_{C_i} \in C_i$, $i \geq 2$, be some representative of C_i and let $z_{C_i} = \Sigma_{u_i \in C_i} z_i$. Let $J = \{i \mid z_{C_i} = 0\}$. Define

$$t_u = \Pi_{i \notin J \cup \{1\}} (t'_{s_{C_i}})^{z_{C_i}}.^2$$

We show A): Using that the $\ulcorner u_i \urcorner$ are standard, it is easy to see that $\ulcorner u \urcorner = \Pi_{i \notin J \cup \{1\}} \ulcorner s_{C_i} \urcorner^{z_{C_i}}$. By induction, $\ulcorner s_{C_i} \urcorner = t'_{s_{C_i}} \sigma'$. Thus, $t_u \sigma' = \ulcorner u \urcorner$. We now show B): We have $\ulcorner u \urcorner = \Pi_{i \notin J \cup \{1\}} \ulcorner s_{C_i} \urcorner^{z_{C_i}}$. Since the factors of $\ulcorner u \urcorner$ are normalized, since they belong to different equivalence classes, and since no factor is 1, we have by induction that

$$t_{\ulcorner u \urcorner} = \Pi_{i \notin J \cup \{1\}} (t'_{\ulcorner s_{C_i} \urcorner})^{z_{C_i}} = \Pi_{i \notin J \cup \{1\}} (t'_{s_{C_i}})^{z_{C_i}} = t_u.$$

Obviously, D) is satisfied since u is non-standard. We now show C). Let $s \in \mathcal{S}_{ext}(t_u) \setminus (\{t_u\} \cup V_F)$ with $t'_{s\sigma'} = t_{s\sigma}$. Then, $s \in \mathcal{S}_{ext}(t'_{s_{C_i}})$. Since $s \notin V_F$, we have that $t'_{s_{C_i}} \notin V_F$, and thus, $t'_{s_{C_i}} = t_{s_{C_i}}$. We know that $s_{C_i} = u_j$ for some j. Thus, $s \in \mathcal{S}_{ext}(t_{u_j})$ and induction yields $t'_{s\sigma'} = s$.

5. If $u = Exp(v, M)$, we distinguish different cases:

   (a) If $\ulcorner v \urcorner \neq Exp(\cdot, \cdot)$, then $t_u = t'_v$ if $\ulcorner M \urcorner = 1$ and $t_u = Exp(t'_v, t'_M)$ otherwise.

   To show A) to D), first assume that $\ulcorner M \urcorner = 1$. We have by induction $t_u \sigma' = t'_v \sigma' = \ulcorner v \urcorner = \ulcorner u \urcorner$. This shows A). To show B), note that $\ulcorner v \urcorner$ is an atom, a pair, or encryption. We have that $t_u = t'_v$. If $t'_v \in V_F$, then $\ulcorner v \urcorner \in F$, and thus, $t_v = t'_v \in V_F$. Otherwise, $\ulcorner v \urcorner \notin F$, and therefore, $t'_v = t_v$. Thus, in both cases by induction we obtain $t_u = t'_v = t_v = t_{\ulcorner v \urcorner} = t_{\ulcorner u \urcorner}$. For D) nothing is to show. Now, we prove C). Let $s \in \mathcal{S}_{ext}(t_u) \setminus (\{t_u\} \cup V_F)$ with $t'_{s\sigma'} = t_{s\sigma}$. We have $t_u = t'_v$, and thus, $s \in \mathcal{S}_{ext}(t'_v)$. Since $s \notin V_F$, we conclude $t'_v = t_v$. Thus, $s \in \mathcal{S}_{ext}(t_v)$ and induction yields $t'_{s\sigma'} = s$.
   Now, assume that $\ulcorner M \urcorner \neq 1$. By induction, $t_u \sigma' = Exp(t'_v \sigma', t'_M \sigma') = Exp(\ulcorner v \urcorner, \ulcorner M \urcorner) = \ulcorner u \urcorner$. This shows A). By induction we have $t_{\ulcorner u \urcorner} = Exp(t'_{\ulcorner v \urcorner}, t'_{\ulcorner M \urcorner}) = Exp(t'_v, t'_M) = t_u$ which shows B). By definition of t_u, we immediately obtain D). Now, we prove C). Let $s \in \mathcal{S}_{ext}(t_u) \setminus (\{t_u\} \cup V_F)$ with $t'_{s\sigma'} = t_{s\sigma}$. We have $t_u = Exp(t'_v, t'_M)$, and thus, $s \in \mathcal{S}_{ext}(t'_v)$ (or $s \in \mathcal{S}_{ext}(t'_M)$). Since $s \notin V_F$, we conclude $t'_v = t_v$ (or $t'_M = t_M$). Thus, $s \in \mathcal{S}_{ext}(t_v)$ (or $s \in \mathcal{S}_{ext}(t_M)$) and induction yields $t'_{s\sigma'} = s$. Note that M is a proper extended subterm of u.

   (b) If $\ulcorner v \urcorner = Exp(v', M')$ for some v' and M', then by induction and D) we know that $t_v = Exp(v'', M'')$ for some v'' and M'' with $v''\sigma' = v'$ and $M''\sigma' = M'$. (Note that v is a standard message, by definition of messages.) If $\ulcorner M' \cdot M \urcorner = 1$, then we define $t_u = v''$. Otherwise, $t_u = Exp(v'', t'_{M' \cdot M})$.

   To show A) to D), first assume that $\ulcorner M' \cdot M \urcorner = 1$. We have $\ulcorner u \urcorner = v' = v''\sigma' = t_u \sigma'$, and thus, A) follows. We now show B). We have that $t_u = v''$ and by induction we know $Exp(v'', M'') = t_v = t_{\ulcorner v \urcorner} = Exp(t'_{v'}, t'_{M'})$. (Note that $v' \neq Exp(\cdot, \cdot)$, and thus, v' is an atom, a pair, or encryption.) Thus, $v'' = t'_{v'}$. We need to show that $t'_{v'} = t_{v'}$.

[2] The empty product has value 1.

If $t'_{v'} \in V_F$, then $v' \in F$, and thus, $t_{v'} = t'_{v'} \in V_F$. Otherwise, $v' \notin F$, and thus, $t'_{v'} = t_{v'}$. In both cases we obtain $t_u = v'' = t'_{v'} = t_{v'} = t_{\ulcorner\urcorner_u}$. For D) nothing is to show since $\ulcorner u \urcorner = v' \neq Exp(\cdot,\cdot)$. Property C) easily follows from the fact that $s \in \mathcal{S}_{ext}(v'') \subseteq \mathcal{S}_{ext}(t_v)$.
Now, let us assume that $\ulcorner M' \cdot M \urcorner \neq 1$. By induction we have $\ulcorner u \urcorner = Exp(v', \ulcorner M' \cdot M \urcorner) = Exp(v''\sigma', t'_{M' \cdot M}\sigma') = t_u\sigma'$. This shows A). To see B), we first conclude as above that $t'_{v'} = v''$. We have $\ulcorner u \urcorner = Exp(v', \ulcorner M' \cdot M \urcorner)$, and thus, $t_{\ulcorner\urcorner_u} = Exp(t'_{v'}, t'_{\ulcorner M' \cdot M \urcorner}) = Exp(v'', t'_{M' \cdot M}) = t_u$. We obtain D) by definition of t_u. Now, we prove C). Let $s \in \mathcal{S}_{ext}(t_u) \setminus (\{t_u\} \cup V_F)$ with $t'_{s\sigma'} = t_{s\sigma}$. We have $t_u = Exp(v'', t'_{M' \cdot M}) = Exp(v'', t'_{M''\sigma' \cdot M})$. Thus, $s \in \mathcal{S}_{ext}(v'')$ or $s \in \mathcal{S}_{ext}(t'_{M''\sigma' \cdot M})$. The former case is dealt with as above. For the latter case, first assume that $s = t'_{M''\sigma' \cdot M}$. By B) we have $t'_{s\sigma'} = s$. Now, assume that $s \neq t'_{M''\sigma' \cdot M}$. Since $s \notin V_F$, we know that $t'_{M''\sigma' \cdot M} \notin V_F$, and thus, $t'_{M''\sigma' \cdot M} = t_{M''\sigma' \cdot M}$. Assume that $M = t_1^{z_1} \cdots t_n^{z_n}$ and $M'' = {t''}_1^{z''_1} \cdots {t''}_{n''}^{z''_{n''}}$. We know that $t_{M''\sigma' \cdot M}$ is of the form $\Pi_{i \notin J \cup \{1\}} (t'_{s_i})^{z_{C_i}}$ where $s_i = t''_j\sigma'$ or $s_i = t_j$ for some j. Thus, $s \in \mathcal{S}_{ext}(t'_{s_i})$. First, consider the case where $s_i = t_j$. Then, $s \in \mathcal{S}_{ext}(t'_{t_j}) = \mathcal{S}_{ext}(t_{t_j})$ since $s \notin V_F$. By induction and since t_j is a proper subterm of u, we obtain $t'_{s\sigma'} = s$. Now, assume that $s_i = t''_j\sigma'$. Thus, $s \in \mathcal{S}_{ext}(t'_{t''_j\sigma'})$. Since $s \notin V_F$, we know that $t'_{t''_j\sigma'} \notin V_F$. In particular, $s \in \mathcal{S}_{ext}(t_{t''_j\sigma'})$. Also, we know that $t''_j \in \mathcal{S}_{ext}(t_v)$. Since $t'_{t''_j\sigma'} \notin V_F$, and thus, $t'_{t''_j\sigma'} = t_{t''_j\sigma'}$, induction yields that $t_{t''_j\sigma'} = t''_j$. Hence, $s \in \mathcal{S}_{ext}(t''_j) \subseteq \mathcal{S}_{ext}(t_v)$, and thus, since v is a proper subterm of u, induction yields $t'_{s\sigma'} = s$.

We just proved the following lemma:

Lemma 14

1. $t_u\sigma' = \ulcorner u \urcorner$ *and* $t'_u\sigma' = \ulcorner u \urcorner$.
2. $t_u = t_{\ulcorner\urcorner_u}$, $t'_u = t'_{\ulcorner\urcorner_u}$, $t_{t_u\sigma'} = t_u$, *and* $t'_{t'_u\sigma'} = t'_u$.
3. $t'_{s\sigma'} = s$ *for all* $s \in \mathcal{S}_{ext}(t_u)$ *with* $t'_{s\sigma'} = t_{s\sigma'}$.
4. *If* $\ulcorner u \urcorner = Exp(\cdot,\cdot)$ *and* u *is standard, then* $t_u = Exp(\cdot,\cdot)$.

To prove Theorem 3, we also need the following lemma.

Lemma 15 *Let* u *be a normalized term,* σ *be a normalized substitution, and* $\mathcal{F} = \bigcup_{x \in \mathcal{V}(u)} \mathcal{F}(\sigma(x))$ *be the set of factors of* σ, $V_{\mathcal{F}} = \{x_s \mid s \in \mathcal{F}\}$, *and* $\sigma'(x_s) = s$ *for every* $s \in \mathcal{F}$. *For a standard or non-standard message* s, *let* t_s *denote its* $\mathcal{F}$*-normal form. Then,*

$$\mathcal{S}(t_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\},$$

and thus,

$$\mathcal{S}(t'_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}.$$

Proof. We proceed by induction on (the depth of) u.

If $\ulcorner u\sigma \urcorner \in \mathcal{F}$ and $\ulcorner u\sigma \urcorner$ is an atom, a pair, or encryption, then $t_{u\sigma} \in V_{\mathcal{F}}$, and the inclusions follow immediately. Otherwise, if $\ulcorner u\sigma \urcorner \notin \mathcal{F}$, we distinguish different cases:

First assume that u is a variable. Then, $u\sigma \in \mathcal{F}$ in case $u\sigma$ is an atom, a pair, or encryption. Thus, $t_{u\sigma} \in V_{\mathcal{F}}$. If $u\sigma = Exp(v, t_1^{z_1} \cdots t_n^{z_n})$, then $v, t_1, \ldots, t_n \in \mathcal{F}$. Since $u\sigma$ is normalized, the definition of $t_{u\sigma}$ yields that $t_{u\sigma} = Exp(x_v, x_{t_1}^{z_1} \cdots x_{t_n}^{z_n})$. In both cases we obtain that $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$.

If $u\sigma \in \mathcal{A}$, the statement is obvious.

If $u = \langle u_1, u_2 \rangle$, then $t_{u\sigma} = \langle t_{u_1\sigma}, t_{u_2\sigma} \rangle$. Thus, by induction and since $\mathcal{S}_{ext}(u_1), \mathcal{S}_{ext}(u_2) \subseteq \mathcal{S}_{ext}(u)$ we have that $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{u\sigma}\} \cup \mathcal{S}(t_{u_1\sigma}) \cup \mathcal{S}(t_{u_2\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$.

Now assume that $u = u_1^{z_1} \cdots u_n^{z_n}$ and $\ulcorner u\sigma \urcorner \neq 1$. With the notation from above, we have that $t_{u\sigma} = \Pi_{i \notin J \cup \{1\}} (t'_{s_{C_i}\sigma})^{z_{C_i}}$. Note that s_{C_i} is some u_j. Thus, by induction and since $\mathcal{S}_{ext}(u_i) \subseteq \mathcal{S}_{ext}(u)$ we have $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{u\sigma}\} \cup \bigcup_{i=1}^{n} \mathcal{S}(t'_{u_i\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$.

Finally, assume that $u = Exp(v, M)$. We distinguish two cases:

First, assume that $\ulcorner v\sigma \urcorner \neq Exp(\cdot, \cdot)$. Then, $t_{u\sigma} = t'_v$ or $t_{u\sigma} = Exp(t'_{v\sigma}, t'_{M\sigma})$. Thus, $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{u\sigma}\} \cup \mathcal{S}(t'_{v\sigma}) \cup \mathcal{S}(t'_{M\sigma})$. By induction and since $\mathcal{S}_{ext}(v), \mathcal{S}_{ext}(M) \subseteq \mathcal{S}_{ext}(u)$ this yields $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$.

Second, assume that $\ulcorner v\sigma \urcorner = Exp(v', M')$. We know by Lemma 14, 4. that $t_{v\sigma} = Exp(v'', M'')$ with $v''\sigma' = v'$ and $M''\sigma' = M'$. If $t_u = v''$, then by induction and since $\mathcal{S}_{ext}(v) \subseteq \mathcal{S}_{ext}(u)$ we have that $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$. Otherwise, $t_u = Exp(v'', t'_{M' \cdot M\sigma}) = Exp(v'', t'_{M''\sigma' \cdot M\sigma})$. If $t'_{M''\sigma' \cdot M\sigma} \in V_{\mathcal{F}}$, then by induction and since $\mathcal{S}_{ext}(v) \subseteq \mathcal{S}_{ext}(u)$ we have $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$. Otherwise, $t'_{M''\sigma' \cdot M\sigma} \notin V_{\mathcal{F}}$, and thus, $t'_{M''\sigma' \cdot M\sigma} = t_{M''\sigma' \cdot M\sigma}$. Assume $M = t_1^{z_1} \cdots t_n^{z_n}$ and $M'' = {t''}_1^{z''_1} \cdots {t''}_{n''}^{z''_{n''}}$. We know that $t_{M''\sigma' \cdot M\sigma}$ is of the form $\Pi_{i \notin J \cup \{1\}} (t'_{s_i})^{z_{C_i}}$ where $s_i = t_j\sigma$ or $s_j = t''_j\sigma'$ for some j. (Recall that by convention if the product is empty, then its value is 1.) Thus, $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{u\sigma}\} \cup \mathcal{S}(t_{v\sigma}) \cup \mathcal{S}(\{t'_{t_j\sigma} \mid j = 1, \ldots, n\}) \cup \mathcal{S}(\{t'_{t''_j\sigma'} \mid j = 1, \ldots, n''\})$. By Lemma 14, 3. we know that $t'_{t''_j\sigma'} = t''_j$ in case $t'_{t''_j\sigma'} \notin V_{\mathcal{F}}$, and thus, $t'_{t''_j\sigma'} = t_{t''_j\sigma'}$, since $t''_j \in \mathcal{S}(t_{v\sigma})$. Thus, $\mathcal{S}(\{t'_{t''_j\sigma'} \mid j = 1, \ldots, n''\}) \subseteq \mathcal{S}(t_{v\sigma}) \cup V_{\mathcal{F}}$. Induction yields that $\mathcal{S}(t_{v\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$ since v is a proper subterm of $\mathcal{S}_{ext}(u)$. Also, $\mathcal{S}(t'_{t_j\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$ since t_j is a proper subterm of u. Thus, $\mathcal{S}(t_{u\sigma}) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(u)\} \cup V_{\mathcal{F}} \cup \{1\}$. □

Theorem 3 *For every protocol P, if (π, σ) is a minimal attack on P, then $|\{\sigma(x) \mid x \in \mathcal{V}\}| \leq 4 \cdot |P|_{ext} + 1$, where $|P|_{ext}$ is the size of P as defined in Section 2.2.*

PROOF. We first need some notation. For a set E of terms and a term t, let $E_{<t} = \{t' \in E \mid t' < t\}$ be the set of all terms in E that are smaller than t (w.r.t. the subterm ordering $<$). Recall that $\mathsf{Card}(E)$ denote the cardinality of the set E.

Let F' be the set of all terms s of the form $Exp(\cdot, \cdot)$ such that there exists $x \in \mathcal{V}$ with $s \in \mathcal{F}(\sigma(x))$ and there exists $y \in \mathcal{V}$ with $s = \sigma(y)$. We associate with each $s \in F'$ one variable y such that $s = \sigma(y)$. Let $V_{F'} \subseteq \mathcal{V}$ be the set of variables in $\mathcal{V}$ associated to some message.

Let $F = \{s \mid x \in \mathcal{V}, s \in \mathcal{F}(\sigma(x))\} \setminus F'$. For every $s \in F$ we introduce a new variable x_s. Let $\mathcal{V}_F$ be the set of these variables.

By Lemma 13 we know that for every $s \in F$ there exists $\hat{s} \in \mathcal{S}(P)$ such that $\hat{s} \sqsubseteq_\sigma s$. This immediately implies:

Claim I: $\mathsf{Card}(F) \leq |P|$.

Let $\mathcal{F} = F \cup F'$ be the set of factors occurring in σ. For $s \in F$ define $\sigma'(x_s) = s$ and for $s \in F'$ define $\sigma'(x) = s$ in case x is the variable associated with s. For a standard or non-standard message m let t_m denote its $\mathcal{F}$-normal form.

Define

$$G = \{t_{u\sigma} \mid u \in \mathcal{S}(P)\}.$$

By Lemma 14, we know that

$$t_{u\sigma}\sigma' = \ulcorner u\sigma \urcorner. \tag{7}$$

Claim II: $|G| \leq 3|P|_{ext} + 1$.

Proof of Claim II: By Lemma 15 we obtain $\mathcal{S}(G) \subseteq \{t_{s\sigma} \mid s \in \mathcal{S}_{ext}(P)\} \cup V_F \cup V_{F'} \cup \{1\}$. Thus, $|G| = \mathsf{Card}(\mathcal{S}(G)) \leq 3 \cdot |P|_{ext} + 1$.

Define $S = \{\sigma(x) \mid x \in \mathcal{V}\}$ and $S' = \{\sigma'(x) \mid x \in V_F \cup V_{F'}\}$. To bound the number of subterms of S we first consider the subterms of F. From Lemma 13 and (7) we can conclude that $F \subseteq \{t_{u\sigma}\sigma' \mid u \in \mathcal{S}(P)\} \subseteq \{t\sigma' \mid t \in \mathcal{S}(G)\}$.

Let $s \in F$. Then,

$$\begin{aligned} \mathcal{S}(s) &\subseteq \{t\sigma' \mid t \in \mathcal{S}(G)\} \cup \mathcal{S}(S'_{<s}) \\ &\subseteq \{t\sigma' \mid t \in \mathcal{S}(G)\} \cup \mathcal{S}((F \cup F')_{<s}) \\ &\subseteq \{t\sigma' \mid t \in \mathcal{S}(G)\} \cup \mathcal{S}(F_{<s}) \cup \mathcal{S}(F'_{<s}) \end{aligned} \tag{8}$$

From this we obain for $x \in \mathcal{V}$ that

$$\begin{aligned} \mathcal{S}(F_{<\sigma(x)}) &= \bigcup_{s \in F_{<\sigma(x)}} \mathcal{S}(s) \\ &\subseteq \{t\sigma' \mid t \in \mathcal{S}(G)\} \cup \mathcal{S}(F'_{<\sigma(x)}) \\ &\subseteq \{t\sigma' \mid t \in \mathcal{S}(G)\} \cup \mathcal{S}(S_{<\sigma(x)}). \end{aligned} \tag{9}$$

For $\mathcal{S}(\sigma(x))$ we obtain:

$$\begin{aligned} \mathcal{S}(\sigma(x)) &= \{\sigma(x)\} \cup \mathcal{S}(\mathcal{F}(\sigma(x))) \\ &= \{\sigma(x)\} \cup \mathcal{S}(\mathcal{F}(\sigma(x)) \cap F) \cup \mathcal{S}(\mathcal{F}(\sigma(x)) \cap F') \\ &\subseteq \{\sigma(x)\} \cup \mathcal{S}(F_{<\sigma(x)}) \cup \mathcal{S}(S_{<\sigma(x)}) \\ &\subseteq \{\sigma(x)\} \cup \{t\sigma' \mid t \in \mathcal{S}(G)\} \cup \mathcal{S}(S_{<\sigma(x)}) \end{aligned} \tag{10}$$

where in (10) we use (9). Using (10) we conclude

$$\begin{aligned} \mathcal{S}(S) &= \bigcup_{x \in \mathcal{V}} \mathcal{S}(\sigma(x)) \\ &\subseteq S \cup \{t\sigma' \mid t \in \mathcal{S}(G)\}. \end{aligned} \tag{11}$$

Thus, $|S| = \mathsf{Card}(\mathcal{S}(S)) \leq \mathsf{Card}(S) + |G| \leq \mathcal{V} + 3|P|_{ext} + 1 \leq 4|P|_{ext} + 1$. □

From the proof of this theorem we immediately obtain:

Corollary 1 *For every protocol P and normal attack (π, σ) on P we have $|R_i\sigma, S_0\sigma, \ldots, S_{i-1}\sigma| \leq 5 \cdot |P|_{ext}$ and $|\mathsf{secret}, S_0\sigma, \ldots, S_k\sigma| \leq 5 \cdot |P|_{ext} + 1$, for every $i \in \{1, \ldots, k\}$ with S_j and R_j as defined above.*

6 Extending the Dolev-Yao Intruder by Diffie-Hellman Exponentiation

We extend the Dolev-Yao intruder given by the intruder rules $L_c \cup L_d$ (see Section 2.3) by rules L_o that allow the intruder to perform exponentiation. This extended intruder is called the DH intruder. We show that L_o is a set of oracle rules and that the preconditions of Theorem 1 are satisfied. From this we can conclude that the insecurity problem with respect to the DH intruder is NP-complete.

Definition 7 *We define $L_o = L_{oc} \cup L_{od}$ to be the set of* DH rules *of the form*

$$t, t_1, \ldots, t_n \to \ulcorner Exp(t, t_1^{z_1} \cdots t_n^{z_n}) \urcorner =: u$$

with $n \geq 1$, $z_i \in \boldsymbol{Z} \setminus \{0\}$, $1 \leq i \leq n$, $t, t_1, \ldots, t_n$ normalized standard messages. If u is of the form $Exp(\cdot,\cdot)$, then the above rule belongs to $L_{oc}(u)$ (the set of composition DH rules*), and to $L_{od}(u)$ (the set of* decomposition DH rules*) otherwise. We call the intruder using the rules L_o as oracle rules the* DH intruder*. We call t in the above DH rule the* head *of this rule and we refer to $z_1, \ldots, z_n$ as the* product exponents *of this rule. We require w.l.o.g. that the head t of a decompsition DH rule is of the form $Exp(\cdot,\cdot)$ since otherwise $t = u$. Also, w.l.o.g. we may assume that $t_i \neq t_j$ for every $i \neq j$, and that $z_i \neq 0$ for every i.*

Using that the messages on the right-hand side of a DH rule are normalized, one easily observes the following.

Lemma 16 *The rules in L_{oc} and L_{od} are composition and decomposition guess rules, respectively.*

In the following section, we show that the set L_o is in fact a set of oracle rules. We then show that OracelRule is decidable in deterministic polynomial time. Finally, we prove that DH rules allow polynomial product exponent attacks. Having shown all this, using Theorem 1 we conclude that Insecure is in *NP*.

6.1 Diffie-Hellman Rules are Oracle Rules

To prove that L_o is a set of oracle rules we first show the following lemma.

Lemma 17 *Let E be a finite set of normalized standard messages and t be a standard message such that t can be derived from t (w.r.t. $\mathcal{L}$). Let D be a derivation from E with goal t. Then, there exists a derivation D' from E with goal t such that*

1. *D' is of the same length as D, and*
2. *for every DH rule $L \in D' \cap L_o$ with head t' we have that $t' \in E$ or there exists a t'-rule $L' \in D' \cap (L_d \cup L_c)$. Moreover, if L is a decomposition DH rule, then $t' \in E$ or there exists a t'-rule $L' \in D' \cap L_d$.*

Proof. Let D be a derivation from E with goal t. From D we construct D' as follows. Assume that $L \in D \cap L_o$ with head t' and that neither $t' \in E$ nor there exists a t'-rule $L' \in D' \cap (L_d \cup L_c)$. Then, there exists $L' \in D \cap L_o(t')$. Assume that L is of the form $t', t_1, \ldots, t_n \to t$ with product exponents $z_1, \ldots, z_n$ and that L' is of the form $t'', t'_1, \ldots, t'_{n'} \to t'$ with product exponents $z'_1, \ldots, z'_{n'}$. Then, obviously $t = \ulcorner Exp(t'', t_1^{z_1} \cdots t_n^{z_n} \cdot {t'}_1^{z'_1} \cdots {t'}_{n'}^{z'_{n'}}) \urcorner$ and t can be obtained by the DH rule $\hat{L} = t'', t_1, \ldots, t_n, t'_1, \ldots, t'_{n'} \to t$ with head t''. Thus, L can be replaced by $\hat{L}$. Iterating this replacement we obtain D' satisfying 1. and for every DH rule $L \in D' \cap L_o$ with head t' there exists no $L' \in D' \cap L_o(t')$ preceding L in D'. From this 2. immediately follows. Note that if L is a decomposition DH rule, then t' is of the form $Exp(\cdot,\cdot)$, and thus, cannot be created by a rule in L_c. □

To show that well formed derivations exist we use the following lemma.

Lemma 18 *Let $D = E_0 \to_{L_1} \ldots E_{n-1} \to_{L_n} E_n$ be a derivation with goal g.*

1. *Assume that for every j with $E_{j-1} \to_{L_j} E_{j-1}, t$ the jth step in D and $L_j \in \mathcal{L}_d(t)$, there exists $t' \in E_{j-1}$ such that t is a subterm of t' and either $t' \in E_0$ or there exists i with $i < j$ and $L_i \in \mathcal{L}_d(t')$. Then, if $L \in D \cap \mathcal{L}_d(t)$ for some L and t, then $t \in \mathcal{S}(E_0)$.*
2. *Assume that for every $i < n$ and t with $L_i \in \mathcal{L}_c(t)$, there exists j with $i < j$ such that L_j is a t'-rule and $t \in \mathcal{S}(\{t'\} \cup E_0)$. Then, if $L \in D \cap \mathcal{L}_c(t)$ for some L and t, then $t \in \mathcal{S}(E_0, g)$.*

Given both the assumptions in 1. and 2., it follows that D is a well formed derivation with goal g.

PROOF. 1. is immediate by induction on $j \in \{1,\ldots,n\}$. Given the assumptions in 2., we prove by induction on $n-i$ that for all $i \in \{1,\ldots,n\}$, $L_i \in \mathcal{L}_c(t)$ implies $t \in \mathcal{S}(E_0, g)$. If $n-i=0$, then $t=g$ and therefore $t \in \mathcal{S}(E_0,g)$. For the induction step, the assumptions in 2. imply that there exists $j>i$ such that L_j is a t'-rule and $t \in \mathcal{S}(E_0,t')$. If $L_j \in \mathcal{L}_d(t')$, then $t' \in \mathcal{S}(E_0)$ (see above). If $L_j \in \mathcal{L}_c(t')$, then by induction $t' \in \mathcal{S}(E_0,g)$, and hence, $t \in \mathcal{S}(E_0,g)$.

Given both the assumptions in 1. and 2., it immediately follows that D is a well formed derivation with goal g. □

We are now prepared to prove the following lemma.

Lemma 19 *For every finite normalized set E of standard messages and normalized standard message g, $g \in forge(E)$ implies that there exists a well formed derivation from E with goal g.*

PROOF. Let $E_0 = E$ and $D = E_0 \rightarrow_{L_1} \ldots \rightarrow_{L_n} E_n$ be a derivation of goal g of minimal length. We may assume that D satisfies the properties stated in Lemma 17, 2.

We prove that D satisfies the assumptions Lemma 18, 1. and 2.

1. If $L_j \in L_d(s) \cap \mathcal{L}_d(t)$, and thus, $t \in \mathcal{S}(s)$, then $L_i \notin L_{oc}(s)$, for all $i<j$, since rules in L_{oc} do not create standard terms, and $L_i \notin L_c(s)$, for all $i<j$, by the definition of derivation (since otherwise t would be in the left-hand side of L_i). Therefore, either $s \in E_0$ or there exists $i<j$ with $L_i \in \mathcal{L}_d(s)$.

   If $L_j \in L_{od}(t)$ and t' is the head of L_j, by definition of decomposition DH rules it is easy to see that $t \in \mathcal{S}(t')$. By Lemma 17, 2. it follows that $t' \in E_0$ or there exists $L' \in D \cap L_d(t')$.

   By Lemma 18, 1. it follows that if $L \in D \cap \mathcal{L}_d(t)$ for some L and t, then $t \in \mathcal{S}(E_0)$.

2. If $L_i \in \mathcal{L}_c(t)$ and $i<n$, then by minimality of D, there exists $j>i$ such that t belongs to the left-hand side of L_j. If $L_j \in L_d$, then as in 1. we can conclude that $t \in \mathcal{S}(E_0)$. If $L_j \in L_c(t')$, then $t \in \mathcal{S}(t')$. If $L_j \in L_o(t')$, then first assume that t is the head of L_j. Lemma 17, 2. implies that there exists a t-rule $L' \in D \cap (L_d \cup L_c)$. Since $L_i \in \mathcal{L}_c(t)$ and because of the minimality of D, t can only be generated by one rule, we have $L_i = L' \in L_c$. Thus, $t \neq Exp(\cdot,\cdot)$. But then, by definition of DH rules, $t \in \mathcal{S}(t')$. Now, assume that t is not the head of L_j. If $t \notin \mathcal{S}(t')$, we have that $t \in \mathcal{S}(t'')$ where t'' is the head of L_j and t'' is of the form $Exp(\cdot,\cdot)$. (Otherwise, t can not disappear from t'.) By Lemma 17, 2. it follows that $t'' \in E_0$ or there exists a t''-rule $L' \in D \cap (L_d \cup L_c)$. Since t'' is of the form $Exp(\cdot,\cdot)$ we know that $L' \in D \cap L_d$. Now, 1. implies that $t'' \in \mathcal{S}(E_0)$, and thus, $t \in \mathcal{S}(E_0)$. □

Before we show that DH rules are oracle rules, we need the following lemma.

Lemma 20 *Let $t', t_1, \ldots, t_n, t, u$ be normalized standard terms, $z_1 \ldots, z_n \in \mathbb{Z}$, and let δ be the replacement $[u \leftarrow 1]$ such that $u \neq t$, and $t = \ulcorner Exp(t', t_1^{z_1} \cdots t_n^{z_n})\urcorner$. If $t' = Exp(\cdot,\cdot)$, then we also assume that $u \neq t'$. Then,*

$$\ulcorner t\delta\urcorner = \ulcorner Exp(\ulcorner t'\delta\urcorner, \ulcorner t_1\delta\urcorner^{z_1} \cdots \ulcorner t_n\delta\urcorner^{z_n})\urcorner.$$

PROOF. We distinguish two cases. First, assume that $t' = Exp(v, M)$ for some normalized term v and a normalized product M. Note that $v \neq Exp(\cdot,\cdot)$ since t' is normalized.Then,

$$\begin{aligned}
\ulcorner Exp(\ulcorner t'\delta\urcorner, \ulcorner t_1\delta\urcorner^{z_1} \cdots \ulcorner t_n\delta\urcorner^{z_n})\urcorner &= \ulcorner Exp(\ulcorner v\delta\urcorner, \ulcorner M\delta\urcorner \cdot \ulcorner t_1\delta\urcorner^{z_1} \cdots \ulcorner t_n\delta\urcorner^{z_n})\urcorner && (*)\\
&= \ulcorner Exp(v\delta, M\delta \cdot (t_1\delta)^{z_1} \cdots (t_n\delta)^{z_n})\urcorner \\
&= \ulcorner Exp(v, M \cdot t_1^{z_1} \cdots t_n^{z_n})\delta\urcorner && (**)\\
&= \ulcorner\ulcorner Exp(v, M \cdot t_1^{z_1} \cdots t_n^{z_n})\urcorner\delta\urcorner && (***)\\
&= \ulcorner t\delta\urcorner
\end{aligned}$$

where in (*) we use that $u \neq t'$, and thus, $\ulcorner t'\delta \urcorner = \ulcorner v\delta \urcorner$ (in this case, $\ulcorner M\delta \urcorner = 1$) or $\ulcorner t'\delta \urcorner = Exp(\ulcorner v\delta \urcorner, \ulcorner M\delta \urcorner)$. In (**) we use that $u \neq t$: If $u = Exp(v, M{\cdot}t_1^{z_1} \cdots t_n^{z_n})$, then since u is normalized $Exp(v, M{\cdot}t_1^{z_1} \cdots t_n^{z_n})$ must be normalized, but then we have that $u = Exp(v, M{\cdot}t_1^{z_1} \cdots t_n^{z_n}) = \ulcorner Exp(v, M{\cdot}t_1^{z_1} \cdots t_n^{z_n}) \urcorner = t$, in contradiction to $u \neq t$. Finally, in (***) we use that $v \neq Exp(\cdot,\cdot)$, $u \neq t$, and Lemma 11, 2.

Now, assume that $t' \neq Exp(\cdot,\cdot)$. For both cases, $u \neq t'$ and $u = t'$, the argument is similar to the one above. (Replace v by t' and omit $\ulcorner M\delta \urcorner$, $M\delta$, and M in the above identities.) □

We also need the following lemma.

Lemma 21 *Let $z_1, \ldots, z_n \in \mathbb{Z} \setminus \{0\}$, and $s, s_1, \ldots, s_n, u$ be normalized standard terms such that $s_i \neq s_j$ for every $i \neq j$, $s_i \neq 1$ and $s_i \neq u$ for every i, $s \neq u$, $u = \ulcorner Exp(s, s_1^{z_1} \cdots s_n^{z_n}) \urcorner$, and $u = Exp(\cdot,\cdot)$. Let δ be the replacement $[u \to 1]$. Then, $u = \ulcorner Exp(\ulcorner s\delta \urcorner, \ulcorner s_1\delta \urcorner^{z_1} \cdots \ulcorner s_n\delta \urcorner^{z_n}) \urcorner$.*

Proof. First, assume that $s \neq Exp(\cdot,\cdot)$. Then, $u = Exp(s, s_1^{z_1} \cdots s_n^{z_n})$. Consequently, $u \notin \mathcal{S}(s, s_1, \ldots, s_n)$, and thus, $s = s\delta$, and $s_i = s_i\delta$ for every i. Therefore, we obtain that $u = \ulcorner Exp(\ulcorner s\delta \urcorner, \ulcorner s_1\delta \urcorner^{z_1} \cdots \ulcorner s_n\delta \urcorner^{z_n}) \urcorner$.

Now, assume that $s = Exp(v, M)$. Since s is normalized we know that $v \neq Exp(\cdot,\cdot)$. Using that $u = Exp(\cdot,\cdot)$, we obtain $u = Exp(v, \ulcorner M \cdot s_1^{z_1} \cdots s_n^{z_n} \urcorner)$ with $\ulcorner M \cdot s_1^{z_1} \cdots s_n^{z_n} \urcorner \neq 1$. Also, $u \notin \mathcal{S}(v)$. Then, with $E = \mathcal{F}(M) \cup \{s_1, \ldots, s_n\}$ there exists a set $E' = \{s'_1, \ldots, s'_{n'}\} \subseteq E$ and $z'_1, \ldots, z'_{n'} \in \mathbb{Z} \setminus \{0\}$ such that $u = Exp(v, {s'}_1^{z'_1} \cdots {s'}_{n'}^{z'_{n'}})$ and $u \notin \mathcal{S}(v, E')$.

Claim. $\ulcorner \ulcorner M\delta \urcorner \cdot \ulcorner s_1\delta \urcorner^{z_1} \cdots \ulcorner s_n\delta \urcorner^{z_n} \urcorner = {s'}_1^{z'_1} \cdots {s'}_{n'}^{z'_{n'}}$.

Proof of the claim. Assume that $M = s_{n+1}^{z_{n+1}} \cdots s_{n''}^{z_{n''}}$. Let $C_i = \{j \in \{1, \ldots, n''\} \mid s_j = s'_i\}$. Then, $z'_i = \Sigma_{j \in C_i} z_j$. Let $C = \bigcup_{i=1}^{n'} C_i$. We have that $\ulcorner s_1^{z_1} \cdots s_n^{z_n} {\cdot} s_{n+1}^{z_{n+1}} \cdots s_{n''}^{z_{n''}} \urcorner = \ulcorner \Pi_{i=1}^{n'} \Pi_{j \in C_i} s_j^{z_j} \urcorner = \Pi_{i=1}^{n'} {s'}_i^{z'_i}$ and $\ulcorner \Pi_{j \notin C} s_j^{z_j} \urcorner = 1$. Using that the s_j are normalized, it follows that $\ulcorner \Pi_{j \notin C} s_j\delta^{z_j} \urcorner = \ulcorner \Pi_{j \notin C} \ulcorner s_j\delta \urcorner^{z_j} \urcorner = 1$. Now with $s'_i\delta = s'_i$ the claim follows.

Using that $s\delta = Exp(v\delta, M\delta)$, $u \neq M$, $u \notin \mathcal{S}(v)$, and $u \neq s$, the claim implies that

$$\ulcorner Exp(\ulcorner s\delta \urcorner, \ulcorner s_1\delta \urcorner^{z_1} \cdots \ulcorner s_n\delta \urcorner^{z_n}) \urcorner = \ulcorner Exp(\ulcorner v\delta \urcorner, \ulcorner M\delta \urcorner {\cdot} \ulcorner s_1\delta \urcorner^{z_1} \cdots \ulcorner s_n\delta \urcorner^{z_n}) \urcorner = u.$$

□

We are prepared to prove that DH rules are oracle rules.

Proposition 1 *The set L_o of DH rules is a set of oracle rules.*

Proof. We check each condition in Definition 5:

1. The first point is an immediate consequence of Lemma 19.
2. No term created with L_{oc} can be decomposed with L_d.
3. Let u be a normalized standard message, F be a set of standard messages with $1 \in F$, and t be a standard message such that $F \backslash u \to_{\mathcal{L}_c(u)} F$ and $F \to_{L_o(t)} F, t$. Let $\delta := [u \leftarrow 1]$. If $u = t$, then $t\delta = 1 \in forge(F\delta)$, and we are done. Now, assume that $u \neq t$. Since $F \to_{L_o(t)} F, t$, there exist $t', t_1, \ldots, t_n \in F$ and $z_1, \ldots, z_n \in \mathbb{Z} \setminus \{0\}$ such that $t_i \neq t_j$ for every $i \neq j$ and $t = \ulcorner Exp(t', t_1^{z_1} \cdots t_n^{z_n}) \urcorner$. If $t' \neq Exp(\cdot,\cdot)$ or $u \neq t'$, then by Lemma 20 we obtain that

$$\ulcorner t\delta \urcorner = \ulcorner Exp(\ulcorner t'\delta \urcorner, \ulcorner t_1\delta \urcorner^{z_1} \cdots \ulcorner t_n\delta \urcorner^{z_n}) \urcorner.$$

Thus, $\ulcorner t\delta \urcorner \in \mathit{forge}(\ulcorner F\delta \urcorner)$. Now, assume that $u = t' = \mathit{Exp}(v, M)$. Then,

$$\begin{aligned}
\ulcorner t\delta \urcorner &= \ulcorner\ulcorner \mathit{Exp}(v, M \cdot t_1^{z_1} \cdots t_n^{z_n})\urcorner \delta \urcorner \\
&= \ulcorner \mathit{Exp}(v, M \cdot t_1^{z_1} \cdots t_n^{z_n})\delta \urcorner && (*) \\
&= \ulcorner \mathit{Exp}(v\delta, M\delta \cdot (t_1\delta)^{z_1} \cdots (t_n\delta)^{z_n})\urcorner && (**) \\
&= \ulcorner \mathit{Exp}(v, M \cdot (t_1\delta)^{z_1} \cdots (t_n\delta)^{z_n})\urcorner && (***) \\
&= \ulcorner \mathit{Exp}(v, M \cdot \ulcorner t_1\delta \urcorner^{z_1} \cdots \ulcorner t_n\delta \urcorner^{z_n})\urcorner \\
&= \ulcorner \mathit{Exp}(u, \ulcorner t_1\delta \urcorner^{z_1} \cdots \ulcorner t_n\delta \urcorner^{z_n})\urcorner
\end{aligned}$$

where in (*) we apply Lemma 11, 2. using that $v \neq u$ and $u \neq t$. In (**) we again use that $u \neq t$. We obtain (***) since $u \notin \mathcal{S}(v, M)$.

Now, to show that $\ulcorner t\delta \urcorner \in \mathit{forge}(\ulcorner F\delta \urcorner)$ it suffices to show that $u \in \mathit{forge}(\ulcorner F\delta \urcorner)$. We know $F \setminus u \rightarrow_{\mathcal{L}_c(u)} F$, and since $u = \mathit{Exp}(\cdot, \cdot)$, we have $F \setminus u \rightarrow_{L_{oc}(u)} F$. Hence, there exist normalized terms $s, s_1, \ldots, s_n \in F \setminus u$ and $z'_1, \ldots, z'_{n'} \in \mathbb{Z} \setminus \{0\}$ such that s and the s_i meet the conditions stated in Lemma 21 and $u = \ulcorner \mathit{Exp}(s, s_1^{z_1} \cdots s_n^{z_n})\urcorner$. Then, Lemma 21 implies $u = \ulcorner \mathit{Exp}(\ulcorner s\delta \urcorner, \ulcorner s_1\delta \urcorner^{z_1} \cdots \ulcorner s_n\delta \urcorner^{z_n})\urcorner$. Thus, $u \in \mathit{forge}(\ulcorner F\delta \urcorner)$. □

6.2 Deciding DH Rules

We show the following proposition.

Proposition 2 *For the DH intruder, the problem* OracelRule *is decidable in deterministic polynomial time.*

Proof. We need to show that there is a deterministic polynomial time algorithm that given E and t decides whether there there exists $t', t_1, \ldots, t_n \in E$ and $z_1, \ldots, z_n \in \mathbb{Z}$ such that $t = \ulcorner \mathit{Exp}(t', t_1^{z_1} \cdots t_n^{z_n})\urcorner$. It is easy to see that $E \rightarrow_{L_o} E, t$ iff

1. $t \neq \mathit{Exp}(\cdot, \cdot)$ and
   (a) $t \in E$, or
   (b) there exits M with $\mathit{Exp}(t, M) \in E$ and $\mathcal{F}(M) \subseteq E$, or
2. $t = \mathit{Exp}(v, M)$ and
   (a) $v \in E$ and $\mathcal{F}(M) \subseteq E$, or
   (b) there exists M' such that $\mathit{Exp}(v, M') \in E$ and $E' := \{t' \mid$ the product exponents in M and M' for t' differ$\} \subseteq E$.

From this characterization of $E \rightarrow_{L_o} E, t$ it is easy to derive a polynomial time algorithm for deciding $E \rightarrow_{L_o} E, t$. □

As an immediate consequence of this proposition, the fact that DH rules are oracle rules Proposition 1, and Theorem 2 we obtain the following corollary.

Corollary 2 *For the DH intruder,* Derive *can be decided in deterministic polynomial time.*

6.3 The DH Rules Allow Polynomial Product Exponent Attacks

In order to show that DH rules allow polynomial product exponent attacks, we will associate with a minimal attack (π, σ) a subsitution σ^Z and a linear equation system such that σ^Z coincides with σ except that the product exponents in σ are replaced by new (integer) variables and such that, for every solution of the linear equation system, (π, σ') where σ' is an instance of σ^Z induced by solution of the linear equation system is an attack. Since the size of the linear equation system can be bounded polynomially in the size of the protocol, and thus, the size of the solutions of this equation system can be bounded polynomially (see [2]), we obtain an attack with polynomially bounded product exponents (see Proposition 3).

We first need to define messages that may have linear expressions as product exponents.

Definition 8 *Let Z be a set of variables. The set $\mathcal{M} = \mathcal{M}(Z)$ of* open messages *over Z, the set $\mathcal{P} = \mathcal{P}(Z)$ of* open products *over Z, the set $\mathcal{L}_{exp} = \mathcal{L}_{exp}(Z)$ of* linear expressions *over Z are defined by the following grammar:*

$$\begin{array}{rcl} \mathcal{M} & ::= & \mathcal{A} \mid \langle \mathcal{M}, \mathcal{M} \rangle \mid \{\mathcal{M}\}^s_{\mathcal{M}} \\ & & \mid \{\mathcal{M}\}^p_{\mathcal{K}} \mid Exp(\mathcal{M}, \mathcal{P}) \\ \mathcal{P} & ::= & \mathcal{M}^{\mathcal{L}exp} \mid \mathcal{M}^{\mathcal{L}exp} \cdot \mathcal{P} \\ \mathcal{L}_{exp} & ::= & \boldsymbol{Z} \mid Z \mid \mathcal{L}_{exp} + \mathcal{L}_{exp} \mid \boldsymbol{Z} \cdot \mathcal{L}_{exp} \end{array}$$

The size $|e|$ of a linear expression e is the number of characters to represent e where integers are encoded in binary. We say that e and e' are *equal* if they are equal modulo associativity and commutativity of addition (modulo AC_+, for short). In particular, for a set of linear expressions S and a linear expression e, we say that e belongs to S if the equivalence class modulo AC_+ of e is one of the equivalences classes induced by S. In the same way, the subset relationship between sets of linear expressions is defined.

For an open message or an open product t let $\mathcal{L}_{exp}(t)$ denote the set of linear expressions occurring in t. The set $\mathcal{S}(t)$ of subterms of t and $|t|$, i.e., the number of subterms of t, are defined as usual. Also, recall that set $\mathcal{S}_{ext}(t)$ of *extended subterms* of t is defined as $\mathcal{S}(t) \cup \{M \mid Exp(u, M) \in \mathcal{S}(t)\}$. We define $|t| = \mathsf{Card}(\mathcal{S}(t))$ and $|t|_{ext} = \mathsf{Card}(\mathcal{S}_{ext}(t))$. Also, we set $|t|_{exp} = 0$ if t is not a product and $|t|_{exp} = |e_1| + \ldots + |e_n|$ if $t = t_1^{e_1} \ldots t_n^{e_n}$. As usual, if E is a finite set of open messages or products, $|E|_{exp} = \Sigma_{s \in E} |s|_{exp}$. With this we define $||t||_{exp} = |\mathcal{S}(t)|_{exp}$. Finally, $||t|| = |t| + ||t||_{exp}$, and $||t||_{ext} = |t|_{ext} + ||t||_{exp}$. We note

Lemma 22 *For every open message or product t we have that $|t|_{ext} \leq 2 \cdot |t|$, and thus, $||t||_{ext} \leq 2 \cdot ||t||$.*

Note that the definitions of $|\cdot|$, $||\cdot||_{exp}$, and $||\cdot||$ for open messages and products correspond to those for messages.

The above definitions and measures of the size of open messages and products extend in the obvious way to sets of open messages, open products, etc.

We call a mapping $\beta : Z \to \boldsymbol{Z}$ an *evaluation mapping*. The evaluation $\beta(e) \in \boldsymbol{Z}$ of a linear expression e w.r.t. β is defined as usual. The evaluation mapping β extends in the obvious way to open messages, open products, sets of open messages, etc.

Throughout this section, β will always denote an evaluation mapping from Z into $\boldsymbol{Z}$.

A *linear equation system* $\mathcal{E}$ (over Z) is a finite set of equations of the form $e = e'$ where e and e' are linear expressions over Z. The *size* $|\mathcal{E}|$ of $\mathcal{E}$ is $\Sigma_{e=e' \in \mathcal{E}} |e| + |e'|$. An evaluation mapping β is a *solution* of $\mathcal{E}$ ($\beta \models E$) if $\beta(e) = \beta(e')$ for every equation $e = e' \in \mathcal{E}$. Let $\mathcal{L}_{exp}(\mathcal{E}) = \{e \mid e = e' \in \mathcal{E} \text{ or } e' = e \in \mathcal{E}\}$ denote the set of linear expressions occurring in $\mathcal{E}$. Let $R_{\mathcal{E}} = \{(e, e') \mid e = e' \in \mathcal{E}\} \subseteq \mathcal{L}_{exp}(\mathcal{E}) \times \mathcal{L}_{exp}(\mathcal{E})$ and let $R^*_{\mathcal{E}}$ denote the reflexive and transitive closure of $R_{\mathcal{E}}$. We write $\mathcal{E} = \mathcal{E}'$ if $R^*_{\mathcal{E}} = R^*_{\mathcal{E}'}$. We write $\mathcal{E} \subseteq \mathcal{E}'$ if $R^*_{\mathcal{E}} \subseteq R^*_{\mathcal{E}'}$. Thus, we consider linear equations modulo reflexivity and transitivity of equality. Recall also that linear expressions are considered modulo AC_+.

6.3.1 β-equivalence, β-tuples, and $\approx_\beta$-equation Systems

Definition 9 *Given β and open messages or open products t and t', we say that t and t' are* β-equal *($t =_\beta t'$) iff $\beta(t) = \beta(t')$.*[3] *We call t and t'* β-equivalent *($t \approx_\beta t'$) iff $\ulcorner\beta(t)\urcorner = \ulcorner\beta(t')\urcorner$.*

Definition 10 *Given β and open messages or open products t and t' such that $t =_\beta t'$, we say that $\mathcal{E}$ is a $=_\beta$-equation system for t and t' iff*

1. *$\beta \models \mathcal{E}$ and*
2. *$t =_{\beta'} t'$ for all $\beta' \models \mathcal{E}$.*

We now show that "small" $=_\beta$-equation systems exist. Recall that, for instance, when we write $||t, t'||_{ext}$ we mean $||\{t, t\}||_{ext}$.

Lemma 23 *Given β and open messages or open products t and t' such that $t =_\beta t'$. Then there exists a $=_\beta$-equation system $\mathcal{E}^{=_\beta}_{t,t'}$ of size $\leq 2||t, t'||^3_{ext}$ for t and t'.*

Proof. We define $R \subseteq \mathcal{S}_{ext}(t, t') \times \mathcal{S}_{ext}(t, t')$ such that $s =_\beta s'$ for all $(s, s') \in R$. More precisely, R is the smallest binary relation over $\mathcal{S}_{ext}(t, t')$ such that

- $(t, t') \in R$,
- If $(s, s') \in R$, and thus, by construction $s =_\beta s'$, and $s = \langle t_1, t_2\rangle$ it follows that $s' = \langle t'_1, t'_2\rangle$ for some open messages t'_1 and t'_2. Then, $\langle t_1, t'_1\rangle \in R$ and $\langle t_2, t'_2\rangle \in R$. For encryption we have analogous conditions on R.
- If $(s, s') \in R$, and s is the product $t_1^{e_1} \cdots t_n^{e_n}$, $n \geq 1$, we have that s' is a product of the form ${t'}_1^{e'_1} \cdots {t'}_n^{e'_n}$, $n \geq 1$. If $t_i =_\beta t'_j$ for some i and j, then $(t_i, t'_j) \in R$. Note that since $s =_\beta s'$, for every t_i there exists at least one $=_\beta$-equal term t'_j.
- If $t = \mathit{Exp}(u, M)$ for some open message u and an open product M, we have that $t' = \mathit{Exp}(u', M')$ for some open message u' and an open message M'. Then, $(u, u') \in R$ and $(M, M') \in R$.

For every $(s, s') \in R$, we define the equation system $\mathcal{E}_{(s,s')}$ as follows: If s (and thus, s') is not a product, then $\mathcal{E}_{(s,s')}$ is the empty set. Otherwise, s is of the form $t_1^{e_1} \cdots t_n^{e_n}$, $n \geq 1$, and s' is of the form ${t'}_1^{e'_1} \cdots {t'}_n^{e'_n}$. We define $\mathcal{E}_{(s,s')} = \{e_i = e'_j \mid (t_i, t'_j) \in R\}$.

By structural induction it is easy to see that $\mathcal{E}_R = \bigcup_{(s,s')\in R} \mathcal{E}_{(s,s')}$ is a $=_\beta$-equation system for t and t'. Obviously, the size of $\mathcal{E}_R$ is $\leq 2||t, t'||^3_{ext}$.

□

In what follows, we refer to $\mathcal{E}^{=_\beta}_{t,t'}$, as defined in the proof, as the *$=_\beta$-equation system induced by t and t'*.

Remark 2 *The equation system $\mathcal{E}^{=_\beta}_{t,t'}$ as constructed in Lemma 23 is uniquely determined.*

We will also need to associate with $\approx_\beta$-equivalent terms an equation system, which we call a $\approx_\beta$-equation system.

Definition 11 *Given β and open messages or open products t and t' such that $t \approx_\beta t'$, we say that $\mathcal{E}$ is a $\approx_\beta$-equation system for t and t' iff*

[3] Recall that equality means equality modulo associativity and commutativity of multiplication in products, e.g., $a^2 \cdot b^3 \cdot c^{-2} = c^{-2} \cdot a^2 \cdot b^3$.

1. $\beta \models \mathcal{E}$ *and*

2. $t \approx_{\beta'} t'$ *for all* $\beta' \models \mathcal{E}$.

To construct such an equation system given t and t', we introduce the notion of a β-tuple.

Definition 12 *Given β and an open message or open product t we say that $(t', \mathcal{E})$ where t' is an open message or an open product and $\mathcal{E}$ is an equation system is a* β-tuple *for t iff*

1. $\beta(t') = \ulcorner \beta(t) \urcorner$,

2. $\beta \models \mathcal{E}$, *and*

3. $t \approx_{\beta'} t'$ *for every* $\beta' \models \mathcal{E}$.

We call t' a β-term *(or the* β-normal form*) of t and $\mathcal{E}$ a* β-equation system *for t.*

The following lemma shows how a $\approx_\beta$-equation system can be obtained using β-tuples.

Lemma 24 *Let t and t' be open messages or open products such that $t \approx_\beta t'$. Assume that there exists a β-tuple $(s, \mathcal{E})$ for t and a β-tuple $(s', \mathcal{E}')$ for t'. Let $\mathcal{E}^{=_\beta}_{s,s'}$ be a $=_\beta$-equation system for s and s' (such a system always exists). Then,*

$$\mathcal{E}^{\approx_\beta}_{t,t'} = \mathcal{E} \cup \mathcal{E}' \cup \mathcal{E}^{=_\beta}_{s,s'}$$

is a $\approx_\beta$-equation system for t and t'.

Proof. We first show that there always exists a $=_\beta$-equation for s and s'. We have that $\beta(s) = \ulcorner \beta(t) \urcorner = \ulcorner \beta(t') \urcorner = \beta(s')$. Thus, $s =_\beta s'$ and by Lemma 23 there exists a $=_\beta$-equation system, which we call $\mathcal{E}^{=_\beta}_{s,s'}$.

We need to show that $\mathcal{E}^{\approx_\beta}_{t,t'}$ is a $\approx_\beta$-equation system for t and t'. Obviously, $\beta \models \mathcal{E}^{\approx_\beta}_{t,t'}$. Let $\beta' \models \mathcal{E}^{\approx_\beta}_{t,t'}$. We need to show that $\ulcorner \beta'(t) \urcorner = \ulcorner \beta'(t') \urcorner$. Since $\beta' \models \mathcal{E}^{=_\beta}_{s,s'}$ we know $\beta(s) = \beta(s')$. From $\beta' \models \mathcal{E}$ ($\beta' \models \mathcal{E}'$) we conclude $\ulcorner \beta'(s) \urcorner = \ulcorner \beta'(t) \urcorner$ ($\ulcorner \beta'(s') \urcorner = \ulcorner \beta'(t') \urcorner$). Thus, $\ulcorner \beta'(t) \urcorner = \ulcorner \beta'(s) \urcorner = \ulcorner \beta'(s') \urcorner = \ulcorner \beta'(t') \urcorner$. □

We now show that β-tuples always exist (Lemma 25). We also show that the size of β-tuples can polynomially be bounded (Lemma 26 to 32). By Lemma 24, this implies that $\approx_\beta$-equation systems always exist and that their size can also be bounded polynomially (Lemma 33).

6.3.2 Existence and Size of β-tuples

Lemma 25 *Let t be an open message or an open product and β be an evaluation mapping. Then, there exists a β-tuple for t.*

Proof. We construct a β-tuple $(t^\beta, \mathcal{E}^\beta_t)$ of t inductively.

If $t \in \mathcal{A}$, then $t^\beta = t$ and $\mathcal{E}^\beta_t = \emptyset$. Obviously, $(t^\beta, \mathcal{E}^\beta_t)$ is a β-tuple for t.

If $t = \langle t_1, t_2 \rangle$, let $(t^\beta_1, \mathcal{E}^\beta_{t_1})$ and $(t^\beta_2, \mathcal{E}^\beta_{t_2})$ be β-tuples for t_1 and t_2. We define $t^\beta = \left\langle t^\beta_1, t^\beta_2 \right\rangle$ and $\mathcal{E}^\beta_t = \mathcal{E}^\beta_{t_1} \cup \mathcal{E}^\beta_{t_2}$. Analogously, β-tuples are constructed in case of encryption. By induction, it is easy to see that $(t^\beta, \mathcal{E}^\beta_t)$ is a β-tuple for t.

If $t = t_1^{e_1} \cdots t_n^{e_n}$, then let $(t^\beta_i, \mathcal{E}^\beta_{t_i})$ be the β-tuple for t_i for every i. Assume that $C_1, \ldots, C_l \subseteq \{t_1, \ldots, t_n\}$ are the equivalence classes over $t_1, \ldots, t_n$ modulo $\approx_\beta$. Define $e_{C_j} = \Sigma_{t_i \in C_j} e_i$ and let $s_{C_j} \in C_j$ be some representative of C_j. W.l.o.g. assume that for $s \in C_1$ we have $s \approx_\beta 1$. (The set C_1 may be empty.) By induction we have that $s^\beta = 1$ for every $s \in C_1$ since $\beta(s^\beta) = \ulcorner \beta(s) \urcorner = 1$. Define

$J = \{j \in \{2, \ldots l\} \mid \beta(e_{C_j}) = 0\}$. If $C = \{s_1, \ldots, s_k\}$ where the s_j are pairwise $\approx_\beta$-equivalent and s_j^β is a β-term for s_j, we define $\mathcal{E}_C^\beta$ to be

$$\bigcup_{i \neq j} \mathcal{E}^{=_\beta}_{s_i^\beta, s_j^\beta}.$$

Note that $\beta(s_i^\beta) = \ulcorner\beta(s_i)\urcorner = \ulcorner\beta(s_j)\urcorner = \beta(s_j^\beta)$, and thus, $s_i^\beta =_\beta s_j^\beta$, and due to Lemma 23, $\mathcal{E}^{=_\beta}_{s_i^\beta, s_j^\beta}$ exists. If $J = \{2, \ldots, l\}$, we set

$$t^\beta = 1$$

and otherwise

$$t^\beta = \Pi_{j \notin J \cup \{1\}} (s_{C_j}^\beta)^{e_{C_j}}.$$

Furthermore, we define

$$\mathcal{E}_t^\beta = \bigcup_{i=1}^{n} \mathcal{E}_{t_i}^\beta \cup \bigcup_{j=2}^{l} \mathcal{E}_{C_j}^\beta \cup \bigcup_{j \in J} \{e_{C_j} = 0\}.$$

By induction, it is easy to see that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t: Induction yields that $\beta \models \mathcal{E}_t^\beta$. By the definition of the normalization function one easily verifies that if $J = \{2, \ldots, l\}$, then $\ulcorner\beta(t)\urcorner = 1$, and thus, $t^\beta = \ulcorner\beta(t)\urcorner$. Otherwise, it is easy to see that $\ulcorner\beta(t)\urcorner = \Pi_{j \notin J \cup \{1\}} \ulcorner\beta(s_{C_j})\urcorner^{\beta(e_{C_j})}$. By induction, $\beta(s_{C_j}^\beta) = \ulcorner\beta(s_{C_j})\urcorner$. Thus, $\beta(t^\beta) = \ulcorner\beta(t)\urcorner$. Now, let $\beta' \models \mathcal{E}_t^\beta$. Let $s, s' \in C_j$ with $s \neq s'$. By definition of $\mathcal{E}_t^\beta$ we have $\beta' \models \mathcal{E}_s^\beta \cup \mathcal{E}_{s'}^\beta \cup \mathcal{E}^{=_\beta}_{s,s'} (= \mathcal{E}^{\approx_\beta}_{s,s'})$. Thus, by Lemma 24, $\ulcorner\beta'(s)\urcorner = \ulcorner\beta'(s')\urcorner$. We also know $s^\beta = 1$, and thus, $\beta'(s^\beta) = 1$ for every $s \in C_1$. Finally, if $j \in J$, we have that $\beta'(e_{C_j}) = 0$. Now, it is easy to see that $\ulcorner\beta'(t)\urcorner = \ulcorner\beta'(t^\beta)\urcorner$.

If $t = \mathit{Exp}(u, M)$, we distinguish two cases.

1. First, assume that $\ulcorner\beta(u)\urcorner \neq \mathit{Exp}(\cdot, \cdot)$. By induction, there exists a β-tuple $(u^\beta, \mathcal{E}_u^\beta)$ for u and a β-tuple $(M^\beta, \mathcal{E}_M^\beta)$ for M. If $\ulcorner\beta(M)\urcorner = 1$. We define

$$t^\beta \quad = \quad u^\beta,$$

and otherwise,

$$t^\beta \quad = \quad \mathit{Exp}(u^\beta, M^\beta).$$

Furthermore, in both cases we set

$$\mathcal{E}_t^\beta \quad = \quad \mathcal{E}_u^\beta \cup \mathcal{E}_M^\beta.$$

By induction, we conclude that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t: Obviously, $\beta \models \mathcal{E}_t^\beta$. We now show that $\beta(t^\beta) = \ulcorner\beta(t)\urcorner$ and $\ulcorner\beta'(t)\urcorner = \ulcorner\beta'(t^\beta)\urcorner$ for every $\beta' \models \mathcal{E}_t^\beta$.

First, assume that $\ulcorner\beta(M)\urcorner = 1$, and thus, $\ulcorner\beta(t)\urcorner = \ulcorner\beta(u)\urcorner$. Thus, by induction, $\ulcorner\beta(t)\urcorner = \ulcorner\beta(u)\urcorner = \beta(u^\beta) = \beta(t^\beta)$. Also, we have that $\beta(M^\beta) = \ulcorner\beta(M)\urcorner = 1$, and thus, $M^\beta = 1$. By induction, $1 = \ulcorner\beta'(M^\beta)\urcorner = \ulcorner\beta'(M)\urcorner$. Hence, $\ulcorner\beta'(t)\urcorner = \ulcorner\beta'(u)\urcorner \stackrel{(*)}{=} \ulcorner\beta'(u^\beta)\urcorner \stackrel{(**)}{=} \ulcorner\beta'(t^\beta)\urcorner$ where $(*)$ is by induction and definition of $\mathcal{E}_t^\beta$, and $(**)$ by definition of t.

Now, assume that $\ulcorner\beta(M)\urcorner \neq 1$. Thus, $\ulcorner\beta(t)\urcorner = Exp(\ulcorner\beta(u)\urcorner, \ulcorner\beta(M)\urcorner) \stackrel{(*)}{=} Exp(\beta(u^\beta), \beta(M^\beta)) \stackrel{(**)}{=} \beta(t^\beta)$ where $(*)$ is by induction and $(**)$ is by definition of t. Now, let $\beta' \models \mathcal{E}_t^\beta$. We have $\ulcorner\beta'(t)\urcorner = \ulcorner Exp(\ulcorner\beta'(u)\urcorner, \ulcorner\beta'(M)\urcorner)\urcorner \stackrel{(*)}{=} \ulcorner Exp(\ulcorner\beta'(u^\beta)\urcorner, \ulcorner\beta'(M^\beta)\urcorner)\urcorner = \ulcorner Exp(\beta'(u^\beta), \beta'(M^\beta))\urcorner \stackrel{(**)}{=} \ulcorner\beta'(t^\beta)\urcorner$ where $(*)$ is by induction and definition of $\mathcal{E}_t^\beta$, and $(**)$ by definition of t^β.

2. If $\ulcorner\beta(u)\urcorner = Exp(u', M')$, by induction there exists a β-tuple $(u^\beta, \mathcal{E}_u^\beta)$ for u. In particular, $\beta(u^\beta) = \ulcorner\beta(u)\urcorner$, and thus, u^β is of the form $Exp(u'', M'')$ where $\beta(u'') = u'$ and $\beta(M'') = M'$. Moreover, by induction there exists a β-tuple $((M'' \cdot M)^\beta, \mathcal{E}^\beta_{(M''\cdot M)})$ for $(M'' \cdot M)$. If $\ulcorner\beta(M'' \cdot M)\urcorner = 1$, and thus, $\ulcorner M' \cdot \beta(M)\urcorner = 1$, we define

$$t^\beta \quad = \quad u''$$

and otherwise

$$t^\beta \quad = \quad Exp(u'', (M'' \cdot M)^\beta).$$

In both cases, we set

$$\mathcal{E}_t^\beta \quad = \quad \mathcal{E}_u^\beta \cup \mathcal{E}^\beta_{(M''\cdot M)}.$$

By induction, we conclude that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t: Obviously, $\beta \models \mathcal{E}_t^\beta$. We now show that $\beta(t^\beta) = \ulcorner\beta(t)\urcorner$ and $\ulcorner\beta'(t)\urcorner = \ulcorner\beta'(t^\beta)\urcorner$ for every $\beta' \models \mathcal{E}_t^\beta$.

Let $(u^\beta, \mathcal{E}_u^\beta) = Exp(u'', M'')$ as above. We have that $\ulcorner\beta(t)\urcorner = \ulcorner Exp(\ulcorner\beta(u)\urcorner, \beta(M))\urcorner = \ulcorner Exp(u', \ulcorner M' \cdot \beta(M)\urcorner)\urcorner$. If $\ulcorner M' \cdot \beta(M)\urcorner = 1$, then $\ulcorner\beta(t)\urcorner = u' = \beta(u'') = \beta(t^\beta)$. Otherwise, $\ulcorner\beta(t)\urcorner = Exp(u', \ulcorner M' \cdot \beta(M)\urcorner) = Exp(\beta(u''), \ulcorner\beta(M'') \cdot \beta(M)\urcorner) = Exp(\beta(u''), \ulcorner\beta(M'' \cdot M)\urcorner) \stackrel{(*)}{=} Exp(\beta(u''), \beta((M'' \cdot M)^\beta)) \stackrel{(**)}{=} \beta(t^\beta)$ where we obtain $(*)$ by induction and $(**)$ by definition of t^β. Now, let $\beta' \models \mathcal{E}_t^\beta$. We have that $\ulcorner\beta'(t)\urcorner = \ulcorner Exp(\beta'(u), \beta'(M))\urcorner$. Since $\beta' \models \mathcal{E}_u^\beta$, it follows by induction that $\ulcorner\beta'(u)\urcorner = \ulcorner\beta'(u^\beta)\urcorner = \ulcorner\beta'(Exp(u'', M''))\urcorner$. Thus, $\ulcorner\beta'(t)\urcorner = \ulcorner Exp(\beta'(u^\beta), \beta'(M))\urcorner = \ulcorner Exp(\beta'(u''), \beta'(M'') \cdot \beta'(M))\urcorner = \ulcorner Exp(\beta'(u''), \beta'(M'' \cdot M))\urcorner$. Since $\beta' \models \mathcal{E}^\beta_{(M''\cdot M)}$, induction yields $\ulcorner\beta'((M'' \cdot M)^\beta)\urcorner = \ulcorner\beta'(M'' \cdot M)\urcorner$. Consequently, $\ulcorner\beta'(t)\urcorner = \ulcorner Exp(\beta'(u''), \beta'((M'' \cdot M)^\beta))\urcorner$. If $t^\beta = Exp(u'', (M'' \cdot M)^\beta)$, then we obtain $\ulcorner\beta'(t)\urcorner = \ulcorner t^\beta\urcorner$. Otherwise, $t^\beta = u''$ and $\ulcorner\beta(M'' \cdot M)\urcorner = 1$, and thus, $(M'' \cdot M)^\beta = 1$. Hence, $\ulcorner\beta'(t)\urcorner = \ulcorner t^\beta\urcorner$. □

In what follows, we denote by t^β the β-term of t as constructed in the proof of the previous lemma.

We will now show that there exists a β-tuple of t of size polynomially bounded in $||t||$. We first bound the size of t^β.

Bounding the size of β-terms. The next lemma implies that β-terms are uniquely determined.

Lemma 26 *For every open message or product t such that $\beta(t) = \ulcorner\beta(t)\urcorner$, we have that $t^\beta = t$.*

PROOF. The proof proceeds by structural induction on t. If t is atomic, a pair, or encryption, the claim is obvious.

If $t = t_1^{e_1} \cdots t_n^{e_n}$, we know that that $\beta(t_i) = \ulcorner\beta(t_i)\urcorner \neq 1$ for every i, $\ulcorner\beta(t_i)\urcorner \neq \ulcorner\beta(t_j)\urcorner$ for every $i \neq j$, and $\beta(e_i) \neq 0$ for every i. From this, one easily concludes that $t^\beta = t$.

Finally, assume that $t = \mathit{Exp}(u, M)$. First we note that $\ulcorner\beta(u)\urcorner \neq \mathit{Exp}(u', M')$ for any normalized u' and M': Otherwise, $\ulcorner\beta(t)\urcorner = \mathit{Exp}(u', M'') = \beta(t)$, and thus, $\beta(u) = u'$. Hence, $\mathit{Exp}(u', M') = \ulcorner\beta(u)\urcorner = \ulcorner u'\urcorner = u'$, which is a contradiction. Also, $\ulcorner\beta(M)\urcorner \neq 1$ since otherwise $\mathit{Exp}(\beta(u), \beta(M)) = \beta(t) = \ulcorner\beta(t)\urcorner = \ulcorner\beta(u)\urcorner$, but we know that $\ulcorner\beta(u)\urcorner \neq \mathit{Exp}(\cdot,\cdot)$. Hence, $\mathit{Exp}(\beta(u), \beta(M)) = \beta(t) = \ulcorner\beta(t)\urcorner = \mathit{Exp}(\ulcorner\beta(u)\urcorner, \ulcorner\beta(M)\urcorner)$, and thus, $\beta(u) = \ulcorner\beta(u)\urcorner$ and $\beta(M) = \ulcorner\beta(M)\urcorner$. By induction, this yields $u^\beta = u$ and $M^\beta = M$. By definition of t^β, we obtain $t^\beta = \mathit{Exp}(u^\beta, M^\beta) = \mathit{Exp}(u, M) = t$. □

For a set E of open messages or products, define $E^\beta = \{t^\beta \mid t \in E\}$. In the following lemma we bound $|t^\beta|_{ext}$ and in Lemma 29 we bound $||t^\beta||_{exp}$. Both lemmas are put together in Lemma 30 to bound $||t^\beta||_{ext}$.

Lemma 27 *For open messages and products $t, t_1, \ldots, t_n$ and an evaluation mapping β we have that*

1. $\mathcal{S}(t^\beta) \subseteq \mathcal{S}(t)^\beta$.
2. $|\ulcorner\beta(t)\urcorner| \leq |t|$ *and* $|\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_1)\urcorner| \leq |t_1, \ldots, t_n|$,
3. $|t^\beta|_{ext} \leq 2 \cdot |t|$.

PROOF. 1. To prove 1., we proceed by structural induction on t.

If $t \in \mathcal{A}$, we have $\mathcal{S}(t^\beta) = \{t\} = \mathcal{S}(t)^\beta$. If $t = \langle t_1, t_2\rangle$, induction yields that $\mathcal{S}(t^\beta) = \{t^\beta\} \cup \mathcal{S}(t_1^\beta) \cup \mathcal{S}(t_2^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}(t_1)^\beta \cup \mathcal{S}(t_2)^\beta = \mathcal{S}(t)^\beta$; analogously for encryption.

Now, assume that $t = t_1^{e_1} \cdots t_n^{e_n}$. If $t^\beta = 1$, we obviously have $\mathcal{S}(t^\beta) \subseteq \mathcal{S}(t)^\beta$. Otherwise, $t^\beta = \Pi_{j \notin J \cup \{1\}} (s_{C_j}^\beta)^{e_{C_j}}$ (see the proof of Lemma 25) where for each $s_{C_j}^\beta$ there exists a t_i such that $t_i^\beta = s_{C_j}^\beta$. Thus, induction yields $\mathcal{S}(t^\beta) \subseteq \{t^\beta\} \cup \bigcup_{j \notin J \cup \{1\}} \mathcal{S}(s_{C_j}^\beta) \subseteq \{t^\beta\} \cup \bigcup_{i=1}^n \mathcal{S}(t_i^\beta) \subseteq \{t^\beta\} \cup \bigcup_{i=1}^n \mathcal{S}(t_i)^\beta = \mathcal{S}(t)^\beta$.

If $t = \mathit{Exp}(u, M)$ and $t^\beta = u^\beta$, induction immediately yields $\mathcal{S}(t^\beta) \subseteq \mathcal{S}(t)^\beta$.

If $t = \mathit{Exp}(u, M)$, $t^\beta = \mathit{Exp}(u^\beta, M^\beta)$, and $M = t_1^{e_1} \cdots t_n^{e_n}$, then $M^\beta \neq 1$ and, by induction, $\mathcal{S}(t^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}(u^\beta) \cup \bigcup_i \mathcal{S}(t_i^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}(u)^\beta \cup \bigcup_i \mathcal{S}(t_i)^\beta = \mathcal{S}(t)^\beta$.

Now, assume that $t = \mathit{Exp}(u, M)$, $u^\beta = \mathit{Exp}(u'', M'')$, and $t^\beta = u''$. Then, $\mathcal{S}(t^\beta) \subseteq \mathcal{S}(u^\beta) \subseteq \mathcal{S}(u)^\beta \subseteq \mathcal{S}(t)^\beta$.

Otherwise, $t = \mathit{Exp}(u, M)$, $u^\beta = \mathit{Exp}(u'', M'')$, and $t^\beta = \mathit{Exp}(u'', (M'' \cdot M)^\beta)$ where $(M'' \cdot M)^\beta \neq 1$. Thus, $(M'' \cdot M)^\beta$ is of the form $\bigcup_{j \notin J \cup \{1\}} (s_{C_j}^\beta)^{e'_j}$ for some e'_j such that with $M'' = t''^{e''_1}_1 \cdots t''^{e''_{n''}}_{n''}$ and $M = t_1^{e_1} \cdots t_n^{e_n}$ every $s_{C_j}^\beta$ equals some t''_i or t_i^β. (Note that $t''_i = t''^\beta_i$ by Lemma 26.) By induction, $t''_i \in \mathcal{S}(u^\beta) \subseteq \mathcal{S}(u)^\beta$. We conclude that $\mathcal{S}(t^\beta) = \{t^\beta\} \cup \mathcal{S}(u'') \cup \bigcup_{j \notin J \cup \{1\}} \mathcal{S}(s_{C_j}^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}(u'') \cup \bigcup_{i=1}^n \mathcal{S}(t_i^\beta) \cup \bigcup_{i=1}^{n''} \mathcal{S}(t''_i) \subseteq \{t^\beta\} \cup \bigcup_{i=1}^n \mathcal{S}(t_i)^\beta \cup \mathcal{S}(u^\beta) \subseteq \{t^\beta\} \cup \bigcup_{i=1}^n \mathcal{S}(t_i)^\beta \cup \mathcal{S}(u)^\beta = \mathcal{S}(t)^\beta$.

2. To see 2. note that $\ulcorner\beta(t)\urcorner = \beta(t^\beta)$. It is easy to see that $|\beta(s)| \leq |s|$ for every open message and product s. Thus, $|\ulcorner\beta(t)\urcorner| = |\beta(t^\beta)| \leq |t^\beta| \overset{(*)}{\leq} \mathsf{Card}(\mathcal{S}(t)^\beta) \leq \mathsf{Card}(\mathcal{S}(t)) = |t|$ where for $(*)$ we use 1. The same argument works for $|\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_1)\urcorner| \leq |t_1, \ldots, t_n|$.

3. Statement 3. is an immediate consequence of 1. and Lemma 22: $|t^\beta|_{ext} \leq 2 \cdot |t^\beta| \leq 2 \cdot \mathsf{Card}(\mathcal{S}(t)^\beta) \leq 2 \cdot |t|$. □

We now bound $||t^\beta||_{exp}$. We need the following lemma.

Lemma 28 *For every open message or product t with $t^\beta = Exp(u, M)$ it follows that $|M|_{exp} \leq |t| \cdot ||t||_{exp} \leq ||t||^2_{ext}$.*

Proof. We show by structural induction on t that $|M|_{exp} \leq |t| \cdot ||t||_{exp}$. Since $|t| \leq ||t||_{ext}$ and $||t||_{exp} \leq ||t||_{ext}$, the lemma follows.

First consider the case were $t = Exp(u', M')$ and $\ulcorner \beta(u') \urcorner \neq Exp(\cdot, \cdot)$. Then, $|M|_{exp} \leq |M'|_{exp} \leq ||t||_{exp}$.

Now, assume that $t = Exp(u', M')$ and $\ulcorner \beta(u') \urcorner = Exp(\cdot, \cdot)$, and thus, $u'^\beta = Exp(u'', M'')$ for some u'' and M''. Then, $M = (M'' \cdot M')^\beta$, and therefore, by induction $|M|_{exp} \leq |M''|_{exp} + |M'|_{exp} \leq |u'| \cdot ||u''||_{exp} + ||t||_{exp} \leq |t| \cdot ||t||_{exp}$ where we use that $||u'||_{exp} \leq ||t||_{exp}$ and $|u'| < |t|$.

In all other cases, if $t^\beta = Exp(u, M)$, then there exists a subterm v of t such that $v^\beta = t^\beta$. In these cases the bound follows by induction. □

We use this lemma to show:

Lemma 29 *For every open message or product t, it follows that $||t^\beta||_{exp} \leq ||t||^3_{ext}$.*

Proof. We show the following claim where E is a finite set of open messages or products such that $\mathcal{S}_{ext}(E) = E$ and t is maximal (w.r.t. subterm ordering) in E.

Claim. $|\bigcup_{s \in E} \mathcal{S}_{ext}(s^\beta)|_{exp} \leq |\bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)|_{exp} + ||t||^2_{ext}$.

Before we prove the claim, we show that from it the lemma follows. With $E = \mathcal{S}_{ext}(t)$ we have that $||t^\beta||_{exp} = |\mathcal{S}_{ext}(t^\beta)|_{exp} \leq |\bigcup_{s \in E} \mathcal{S}_{ext}(s^\beta)|_{exp}$. The claim allows us to iteratively extract a (maximal) term from E and shows that $|\bigcup_{s \in E} \mathcal{S}_{ext}(s^\beta)|_{exp} \leq |t|_{ext} \cdot ||t||^2_{ext}$. (Note that $\mathsf{Card}(E) = |t|_{ext}$.) This yields $||t^\beta||_{exp} \leq ||t||^3_{ext}$.

Proof of the claim. We proceed by structural induction on t. If t is an atomic message, then obviously $|\bigcup_{s \in E} \mathcal{S}_{ext}(s^\beta)|_{exp} = |\bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)|_{exp}$.

If $t = \langle t_1, t_2 \rangle$, then $t^\beta = \left\langle t_1^\beta, t_2^\beta \right\rangle$, and thus, $\mathcal{S}_{ext}(t^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}_{ext}(t_1^\beta) \cup \mathcal{S}_{ext}(t_2^\beta)$. We know that $t_1, t_2 \in E \setminus t$ since $\mathcal{S}_{ext}(E) = E$. Since $|t^\beta|_{exp} = 0$, we obtain $|\bigcup_{s \in E} \mathcal{S}_{ext}(s^\beta)|_{exp} = |\bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)|_{exp}$. For encryption the argument is analogously.

If $t = t_1^{e_1} \cdots t_n^{e_n}$, then $\mathcal{S}_{ext}(t^\beta) \subseteq \{t^\beta\} \cup \bigcup_i \mathcal{S}_{ext}(t_i^\beta)$. We know that $t_1, \ldots, t_n \subseteq E \setminus \{t\}$, and thus, $\mathcal{S}_{ext}(t^\beta) \subseteq \bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta) \cup \{t^\beta\}$. With $|t^\beta|_{exp} \leq |t|_{exp} \leq ||t||^2_{ext}$ the claim follows.

Now, assume that $t = Exp(u, M)$ and $t^\beta = u^\beta$, $t^\beta = u''$, or $t^\beta = Exp(u^\beta, M^\beta)$ (see the cases in Lemma 25). In the first two cases we have $\mathcal{S}_{ext}(t^\beta) \subseteq \mathcal{S}_{ext}(u^\beta) \cup \mathcal{S}_{ext}(M^\beta) \subseteq \bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)$. In the latter case, we have $\mathcal{S}_{ext}(t^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}_{ext}(u^\beta) \cup \mathcal{S}_{ext}(M^\beta) \subseteq \{t^\beta\} \cup \bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)$ and $|t^\beta|_{exp} = 0$. Thus, in every case $|\bigcup_{s \in E} \mathcal{S}_{ext}(s^\beta)|_{exp} = |\bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)|_{exp}$.

Finally, assume that $t = Exp(u, M)$, $\ulcorner \beta(u) \urcorner = Exp(u', M')$, $u^\beta = Exp(u'', M'')$. Then, $t^\beta = Exp(u'', (M'' \cdot M)^\beta)$, and thus, $\mathcal{S}_{ext}(t^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}_{ext}(u^\beta) \cup \mathcal{S}_{ext}((M'' \cdot M)^\beta) \subseteq \{t^\beta\} \cup \mathcal{S}_{ext}(u^\beta) \cup \{(M'' \cdot M)^\beta\} \cup \mathcal{S}_{ext}(M^\beta)$. (Note that by Lemma 26, $\mathcal{S}_{ext}(M''^\beta) = \mathcal{S}_{ext}(M'')$ and that $\mathcal{S}_{ext}(M'') \subseteq \mathcal{S}_{ext}(u^\beta)$.) Consequently, $\mathcal{S}_{ext}(t^\beta) \subseteq \{t^\beta\} \cup \{(M'' \cdot M)^\beta\} \cup \bigcup_{s \in E \setminus \{t\}} \mathcal{S}_{ext}(s^\beta)$. We know that $|t^\beta|_{exp} = 0$ and by Lemma 28, $|(M'' \cdot M)^\beta|_{exp} \leq ||t||^2_{ext}$. This concludes the proof of the claim. □

Putting Lemma 29 and Lemma 27 together, we obtain:

Lemma 30 *For every open message or product t we have $||t^\beta||_{ext} \leq 3 \cdot ||t||^3_{ext}$.*

Bounding the size of β-equation systems. We will now define a β-equation system for t whose size can polynomially be bounded in $||t||_{ext}$.

We first describe the equation system ${\mathcal{E}'}_t^\beta$ for an open message or product t added when going from the equation system of the subterms of t to that of t:

- If t is atomic, a pair, or encryption, then ${\mathcal{E}'}_t^\beta = \emptyset$.
- If $t = t_1^{e_1} \cdots t_n^{e_n}$, then with the notation used in the proof of Lemma 25, we set ${\mathcal{E}'}_t^\beta = \bigcup_{j=2}^{l} \mathcal{E}_{C_j}^\beta \cup \bigcup_{j \in J} \{e_{C_j} = 0\}$.
- If $t = Exp(u, M)$ and $\ulcorner \beta(u) \urcorner \neq Exp(\cdot, \cdot)$, then ${\mathcal{E}'}_t^\beta = \emptyset$. Otherwise, $u^\beta = Exp(u'', M'')$ and we set ${\mathcal{E}'}_t^\beta = {\mathcal{E}'}_{(M'' \cdot M)}^\beta$.

Remark 3 *The equation system ${\mathcal{E}'}_t^\beta$ is uniquely determined (modulo AC_+).*

We define

$$\mathcal{E}_t^\beta = \bigcup_{s \in \mathcal{S}_{ext}(t)} {\mathcal{E}'}_s^\beta.$$

Before we can show that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t, we need to prove the following lemma.

Lemma 31 *Let $M = t_1^{e_1} \cdots t_n^{e_n}$ and $M' = {t'}_1^{e'_1} \cdots {t'}_{n'}^{e'_{n'}}$ such that $\beta(t'_i) = \ulcorner \beta(t'_i) \urcorner$. Let $(t_i^\beta, \mathcal{E}_i)$ be a β-tuple for t_i for every i. Then, $((M' \cdot M)^\beta, {\mathcal{E}'}_{(M' \cdot M)}^\beta \cup \bigcup_i \mathcal{E}_i)$ is a β-tuple for $M' \cdot M$.*

Proof. Let $t = M' \cdot M$. Obviously, $\beta \models {\mathcal{E}'}_t^\beta \cup \bigcup_i \mathcal{E}_i$. We also know that $\beta(t^\beta) = \ulcorner \beta(t) \urcorner$. Let $\beta' \models {\mathcal{E}'}_t^\beta \cup \bigcup_i \mathcal{E}_i$. We need to show that $\ulcorner \beta'(t^\beta) \urcorner = \ulcorner \beta'(t) \urcorner$.

Claim I. $\ulcorner \beta'(t_i) \urcorner = \ulcorner \beta'(t_i^\beta) \urcorner$ and $\ulcorner \beta'(t'_j) \urcorner = \ulcorner \beta'({t'}_j^\beta) \urcorner$ for every i and j.

Proof of Claim I. Since $\beta' \models \mathcal{E}_i$ we immediately have $\ulcorner \beta'(t_i) \urcorner = \ulcorner \beta'(t_i^\beta) \urcorner$. Lemma 26 implies that ${t'}_j^\beta = t'_j$. Thus, $\ulcorner \beta'(t'_j) \urcorner = \ulcorner \beta'({t'}_j^\beta) \urcorner$.

Let the classes $C_1, \ldots, C_l$ be defined as in the proof of Lemma 25.

Claim II. $\ulcorner \beta'(s) \urcorner = \ulcorner \beta'(s') \urcorner$ for every k and $s, s' \in C_k$.

Proof of the Claim II. First, assume that $s = t'_i$ and $s' = t'_j$ for some i and j. Due to Lemma 26, we have that $s^\beta = s$ and $s'^\beta = s'$. Then, by definition of ${\mathcal{E}'}_t^\beta$ we obtain that $\beta'(s) = \beta'(s')$. Now, assume that $s = t_i$ and $s' = t'_j$. Again, we have $s'^\beta = s'$. The definition of ${\mathcal{E}'}_t^\beta$ yields that $\beta'(s') = \beta'(s^\beta)$. Since $\beta' \models \mathcal{E}_i$ we have that $\ulcorner \beta'(s) \urcorner = \ulcorner \beta'(s^\beta) \urcorner$. Thus, $\ulcorner \beta'(s') \urcorner = \ulcorner \beta'(s) \urcorner$. A similar argument can be applied if $s = t_i$ and $s' = t_j$. This concludes the proof of the claim.

By definition of ${\mathcal{E}'}_t^\beta$ we know that $\beta'(e_{C_j}) = 0$ for every $j \in J$. Using Claim I and II it is now easy to see that $\ulcorner \beta'(t^\beta) \urcorner = \ulcorner \beta'(t) \urcorner$. □

We now show that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple and we will bound the size of this tuple.

Lemma 32 *For every open message or product t and every evaluation mapping β the following is true.*

1. *The tuple $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t.*
2. *The size of ${\mathcal{E}'}_t^\beta$ is bounded by a polynomial in $||t||_{ext}$.*

3. *The size of* $(t^\beta, \mathcal{E}_t^\beta)$*, which is the some of the size of* t^β *and* $\mathcal{E}_t^\beta$ *is bounded by a polynomial in* $||t||_{ext}$.

PROOF. 1. We first show 1. by structural induction on t using the construction in Lemma 25. The case where $t \in \mathcal{A}$ is obvious. Now, assume that $t = \langle t_1, t_2 \rangle$. Then, $\mathcal{E}_t^\beta = \mathcal{E}'^\beta_t \cup \bigcup_{s \in \mathcal{S}_{ext}(t_1)} \mathcal{E}'^\beta_s \cup \bigcup_{s \in \mathcal{S}_{ext}(t_2)} \mathcal{E}'^\beta_s$. By definition, $\mathcal{E}_t^\beta = \mathcal{E}_{t_1}^\beta \cup \mathcal{E}_{t_2}^\beta$. (Note that here we use Remark 3.) Just as in the proof of Lemma 25, from this we can conclude that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple. The argument for encyrption is similar.

If $t = t_1^{e_1} \cdots t_n^{e_n}$ and with the notation introduced in the proof of Lemma 25, by definition of $\mathcal{E}'^\beta_t$ and using Remark 3, we can conclude that $\mathcal{E}_t^\beta = \bigcup_i \mathcal{E}_{t_i}^\beta \cup \bigcup_{j=2}^{l} \mathcal{E}_{C_j}^\beta \cup \bigcup_{j \in J} \{e_{C_j} = 0\}$. Again, as in the proof of Lemma 25 it follows that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t.

If $t = Exp(u, M)$ and $\ulcorner \beta(u) \urcorner \neq Exp(\cdot, \cdot)$, then by definition of $\mathcal{E}_t^\beta$ and using Remark 3 we have $\mathcal{E}_t^\beta = \mathcal{E}_u^\beta \cup \mathcal{E}_M^\beta$ and as in Lemma 25 this implies that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t.

Finally, assume that $t = Exp(u, M)$, $\ulcorner \beta(u) \urcorner \neq Exp(\cdot, \cdot)$, and $u^\beta = Exp(u'', M'')$. Assume that $M = t_1^{e_1} \cdots t_n^{e_n}$. Lemma 31 implies that $((M'' \cdot M)^\beta, \mathcal{E}'^\beta_{(M'' \cdot M)} \cup \bigcup_i \mathcal{E}_{t_i}^\beta)$ is a β-tuple for $M'' \cdot M$. By definition of $\mathcal{E}_t^\beta$ and Remark 3 we have that $\mathcal{E}_u^\beta \cup \bigcup_i \mathcal{E}_{t_i}^\beta \cup \mathcal{E}'^\beta_{(M'' \cdot M)} \subseteq \mathcal{E}_t^\beta$. Then, as in the proof of Lemma 25 it follows that $(t^\beta, \mathcal{E}_t^\beta)$ is a β-tuple for t.

2. We now show 2. In case t is atomic, a pair, or encryption, nothing is to show. In case t is a product, using Lemma 23 and 30 it is easy to see that $\mathcal{E}'^\beta_t$ can be bounded by a polynomial in $||t||_{ext}$. For the case $t = Exp(\cdot, \cdot)$, one can obtains a polynmial in $||t||_{ext}$ bounding the size of $\mathcal{E}'^\beta_t$ using Lemma 30 and the case where t is a product.

3. If p is the polynomial bounding the size of $\mathcal{E}'^\beta_t$, then $p(||t||_{ext}) \cdot ||t||_{ext}$ bounds the size of $\mathcal{E}_t^\beta$. By Lemma 30 we know that the size of t^β can be bounded by a polynomial in $||t||_{ext}$. □

As an immediate consequence of Lemma 23, 24, and 32 we obtain the following lemma.

Lemma 33 *Let* t *and* t' *be open messages or open products and* β *be an evaluation mapping such that* $t \approx_\beta t'$*. Then, there exists a* $\approx_\beta$*-equation system for* t *and* t' *of size polynomially bounded in* $||t, t'||_{ext}$.

We will denote such an equation system by $\mathcal{E}_{t,t'}^{\approx_\beta}$.

6.3.3 Bounding the Size of Product Exponents in Attacks

We are now ready to show that DH rules allow polynomial product exponents attacks. Lemma 35 deals with an application of a single intruder rule and Lemma 36 with an iterated application of intruder rules. Finally, Proposition 3 shows that DH rules allow polynomial product attacks.

We first need to show the following lemma.

Lemma 34 *Let* $t_0, \ldots, t_n$ *be open messages and* β *be an evaluating mapping such that* $\beta(t_i) = \ulcorner \beta(t_i) \urcorner$ *for every* i*. Let* $z_1, \ldots, z_n$ *be integer variables not occurring in the* t_i*. Finally, let* $t = Exp(t_0, t_1^{z_1} \cdots t_n^{z_n})^\beta$*. Then, the size of every linear expression in* t *is bounded by* $\max\{|e| \mid e \in \mathcal{L}exp(t_0, t_1, \ldots, t_n)\} + n$.

PROOF. First note that due to Lemma 26 we have $t_i^\beta = t_i$. Let $M = t_1^{z_1} \cdots t_n^{z_n}$. We distinguish different cases.

The case where $\beta(t_0) = \ulcorner \beta(t_0) \urcorner \neq Exp(\cdot, \cdot)$ is easy to prove.

Now, assume that $\beta(t_0) = \ulcorner\beta(t_0)\urcorner = Exp(u', M')$, and thus, $t_0 = Exp(u'', M'')$ with $\beta(u'') = u'$ and $\beta(M'') = M'$. If $t = u''$, the statement of the lemma is obvious. Otherwise, $t = Exp(u'', (M'' \cdot M)^\beta)$ and $(M'' \cdot M)^\beta \neq 1$. Assume that $M'' = {t''_1}^{e''_1} \cdots {t''_{n''}}^{e''_{n''}}$. We know that $\ulcorner\beta(t''_i)\urcorner = \beta(t''_i)$ for every i and $\beta(t''_i) \neq \beta(t''_j)$ for every $i \neq j$. Let $C_1, \ldots, C_l$ be the equivalence classes as defined in the proof of Lemma 25. Then, $t = \Pi_{j \notin J \cup \{1\}} (s^\beta_{C_j})^{e_{C_j}}$. Using that $\ulcorner\beta(t''_i)\urcorner = \beta(t''_i)$, and thus, ${t''_i}^\beta = t''_i$, and $t_i^\beta = t_i$ we know that $s^\beta_{C_j} = t''_i$ or $s^\beta_{C_j} = t_i$ for some i. Also, there is at most one t''_i in every class C_j. Consequently, the size of e_{C_j} is bounded as required. All proper subterms of t are subterms of some t_i. Thus, the size of linear expressions in t can be bounded as required. □

An *extension* of an evaluation mapping $\beta : Z \to \mathbf{Z}$ is a mapping $\beta' : Z' \to \mathbf{Z}$ such that $Z \subseteq Z'$ and $\beta'(z) = \beta(z)$ for all $z \in Z$. By abuse of notation, we often refer to an extension of β by β.

Lemma 35 *Let $t_1, \ldots, t_n$ be open messages, s be a normalized message, β be an evaluation mapping, and $L \in \mathcal{L}$ an intruder rule such that $\beta(t_i) = \ulcorner\beta(t_i)\urcorner$ for all i and $\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner \to s \in L$. Then, there exists an open message t, an equation system $\mathcal{E}$, and an extension β of β such that*

1. *$\beta(t) = s$,*
2. *$\beta \models \mathcal{E}$,*
3. *$\ulcorner\beta'(t_1)\urcorner \ldots, \ulcorner\beta'(t_n)\urcorner \to \ulcorner\beta'(t)\urcorner \in \mathcal{L}$ for every $\beta' \models \mathcal{E}$,*
4. *$\max\{|e| \mid e \in \mathcal{L}exp(t)\} \leq \max\{|e| \mid e \in \mathcal{L}exp(t_1, \ldots, t_n)\} + n$, and*
5. *the size of $\mathcal{E}$ is polynomially bounded in $||t_1, \ldots, t_n||_{ext}$.*

Proof. We consider the different intruder rules L.

First, assume that $L = L_{p1}$. Then, $n = 1$, $\ulcorner\beta(t_1)\urcorner = \langle a, b\rangle$, $s = a$, $t_1 = \langle a', b'\rangle$ where a and b are normalized messages, and a' and b' are open messages with $\beta(a') = a$ and $\beta(b') = b$. We define $t = a'$, $\mathcal{E} = \emptyset$, and β is not extended. Then, obviously conditions 1. to 5. are satisfied. The cases for L_{p2}, L_{ad}, and L_c are similar.

Now, assume that $L = L_{sd}$. Then, $n = 2$, $\ulcorner\beta(t_1)\urcorner = \{a\}^s_b$ and $\ulcorner\beta(t_2)\urcorner = b$ for normalized messages a and b. Thus, $t_1 = \{a'\}^s_{b'}$ and $t_2 = b''$ where a', b', and b'' are open messages such that $\beta(a') = a$ and $\beta(b') = \beta(b'') = b$. We define $t = a'$, $\mathcal{E} = \mathcal{E}^{=_\beta}_{b',b''}$, and β is not extended. Using Lemma 23, it is easy to check that the conditions 1. to 5. are satisfied.

Finally, assume that $L = L_o$. Then, $s = \ulcorner Exp(\beta(t_1), \beta(t_2)^{a_2} \cdots \beta(t_n)^{a_n})\urcorner$ for some $a_i \in \mathbf{Z}$. We define $t = Exp(t_1, t_2^{z_2} \cdots t_n^{z_n})^\beta$, where the z_i are new variables, and $\mathcal{E} = \mathcal{E}^\beta_t$. We extend β such that $\beta(z_i) = a_i$ for every i. By Lemma 25, we have $s = \ulcorner\beta(Exp(t_1, t_2^{z_2} \cdots t_n^{z_n}))\urcorner = \beta(t)$, $\beta \models \mathcal{E}$, $\ulcorner\beta'(t)\urcorner = \ulcorner\beta'(Exp(t_1, t_2^{z_2} \cdots t_n^{z_n}))\urcorner = \ulcorner Exp(\ulcorner\beta'(t_1)\urcorner, \ulcorner\beta'(t_2)\urcorner^{\beta'(z_2)} \cdots \ulcorner\beta'(t_2)\urcorner^{\beta'(z_2)})\urcorner$ for every $\beta' \models \mathcal{E}$. Lemma 34 implies 4. and from Lemma 32 we obtain 5. □

Lemma 36 *Let $t, t_1, \ldots, t_n$ be open messages such that there exists a derivation witnessing $\ulcorner\beta(t)\urcorner \in forge(\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner)$. Then, there exists an extension of β and an equation system $\mathcal{E}$ such that*

1. *$\beta \models \mathcal{E}$,*
2. *$\ulcorner\beta'(t)\urcorner \in forge(\ulcorner\beta'(t_1)\urcorner, \ldots, \ulcorner\beta'(t_n)\urcorner)$ for every $\beta' \models \mathcal{E}$, and*
3. *the size of $\mathcal{E}$ is polynomially bounded in $||t_1, \ldots, t_n, t||_{ext}$.*

PROOF. Let $E = \{\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner\}$ and let D be a well formed derivation witnessing $\ulcorner\beta(t)\urcorner \in$ *forge*(E). We know that the length l of D is polynomially bounded in $|\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner, \ulcorner\beta(t)\urcorner|$. By Lemma 27, 2. we can bound $|\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner, \ulcorner\beta(t)\urcorner|$ by a polynomial in $|t_1, \ldots, t_n, t|$, and thus, in $||t_1, \ldots, t_n, t||_{ext}$. Assume that the ith step of D is $E, s_1, \ldots, s_{i-1} \rightarrow_{L_i} E, s_1, \ldots, s_i$ for every $1 \leq i \leq l$ where s_i is a normalized message for every i and $s_l = \ulcorner\beta(t)\urcorner$. Since D is well formed we have that $s_i \in \mathcal{S}(\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner, \ulcorner\beta(t)\urcorner)$ for every i.

Let $(t^\beta, \mathcal{E}_t^\beta)$ be a β-tuple of t and t_i^β be a β-tuple of t_i for every i. It follows that $\beta(t_i^\beta) = \ulcorner\beta(t_i)\urcorner$, and thus, $E = \{\beta(t_1^\beta), \ldots, \beta(t_n^\beta)\}$. Hence, to the first step of D we can apply Lemma 35 and obtain an open message s_1', an equation system $\mathcal{E}_1$, and an extension of β such that $\beta(s_1') = s_1$, $\beta \models \mathcal{E}_1$, and $\ulcorner\beta'(t_1^\beta)\urcorner, \ldots, \ulcorner\beta'(t_n^\beta)\urcorner \rightarrow_{L_1} \ulcorner\beta'(t_1^\beta)\urcorner, \ldots, \ulcorner\beta'(t_n^\beta)\urcorner, \ulcorner\beta'(s_1')\urcorner$ for every $\beta' \models \mathcal{E}_1$.

Note that $\beta(t_i^\beta) = \ulcorner\beta(t_i^\beta)\urcorner$ for every i and $\beta(s_1') = \ulcorner\beta(s_1')\urcorner = s_1$. Thus, we can apply Lemma 35 inductively and obtain s_j', $\mathcal{E}_j$, and an extension of β such that $\beta(s_j') = s_j$, $\beta \models \mathcal{E}_j$ and $\ulcorner\beta'(t_1^\beta)\urcorner, \ldots, \ulcorner\beta'(t_n^\beta)\urcorner, \ulcorner\beta'(s_1')\urcorner, \ldots, \ulcorner\beta'(s_{j-1}')\urcorner \rightarrow_{L_j} \ulcorner\beta'(t_1^\beta)\urcorner, \ldots, \ulcorner\beta'(t_n^\beta)\urcorner, \ulcorner\beta'(s_1')\urcorner, \ldots, \ulcorner\beta'(s_j')\urcorner$ for every $\beta' \models \mathcal{E}_j$ and $1 \leq j \leq l$. Consequently, $\beta \models \bigcup_{j=1}^{l} \mathcal{E}_j$ and $\ulcorner\beta'(s_l')\urcorner \in$ *forge*$(\ulcorner\beta'(t_1^\beta)\urcorner, \ldots, \ulcorner\beta'(t_n^\beta)\urcorner)$ for every $\beta' \models \bigcup_{j=1}^{l} \mathcal{E}_j$.

If $\beta' \models \bigcup_{i=1}^{n} \mathcal{E}_{t_i}^\beta$, then $\ulcorner\beta'(t_i)\urcorner = \ulcorner\beta'(t_i^\beta)\urcorner$. We know that $\beta(t^\beta) = \ulcorner\beta(t)\urcorner = s_l = \beta(s_l')$. Thus, $t^\beta =_\beta s_l'$. Consequently, due to Lemma 23, the $=_\beta$-equation system $\mathcal{E}_{t^\beta, s_l'}^{=_\beta}$ for t^β and s_l' exists. Now, if $\beta' \models \mathcal{E}_t^\beta \cup \mathcal{E}_{t^\beta, s_l'}^{=_\beta}$ we obtain $\ulcorner\beta'(t)\urcorner = \ulcorner\beta'(t^\beta)\urcorner = \ulcorner\beta'(s_l')\urcorner$. We set

$$\mathcal{E} = \bigcup_{i=1}^{n} \mathcal{E}_{t_i}^\beta \cup \mathcal{E}_t^\beta \cup \bigcup_{j=1}^{l} \mathcal{E}_j \cup \mathcal{E}_{t^\beta, s_l'}^{=_\beta}.$$

It follows that $\beta \models \mathcal{E}$ and $\ulcorner\beta'(t)\urcorner \in$ *forge*$(\ulcorner\beta'(t_1)\urcorner, \ldots, \ulcorner\beta'(t_n)\urcorner)$ for every $\beta' \models \mathcal{E}$.

It remains to show that $\mathcal{E}$ is polynomially bounded in $||t_1, \ldots, t_n, t||_{ext}$. By Lemma 35, it follows that every $\mathcal{E}_j$ is polynomially bounded in $||t_1, \ldots, t_n, s_1', \ldots, s_{j-1}'||_{ext}$. We have that $\beta(s_j') = s_j \in \mathcal{S}(\ulcorner\beta(t_1)\urcorner, \ldots, \ulcorner\beta(t_n)\urcorner, \ulcorner\beta(t)\urcorner)$. Thus, using Lemma 27, $|s_j'|$ can polynomially be bounded in $|t_1, \ldots, t_n, t|$. By Lemma 22, $|s_j'|_{ext}$ can polynomially be bounded in $|t_1, \ldots, t_n, t|$. From Lemma 35 it follows that $\max\{|e| \mid e \in \mathcal{L}exp(s_j')\} \leq \max\{|e| \mid e \in \mathcal{L}exp(t_1^\beta, \ldots, t_n^\beta)\} + n \cdot (j-1)$. We know that $j \leq l$ and that l is polynomially bounded in $||t_1, \ldots, t_n, t||_{ext}$. Also, $\max\{|e| \mid e \in \mathcal{L}exp(t_1^\beta, \ldots, t_n^\beta)\} \leq ||t_i^\beta||_{exp} \leq ||t_i||_{ext}^3$ for some i (see Lemma 29). Thus, there exists a polynomial p such that $||s_j'||_{ext}$ is bounded by $p(||t_1, \ldots, t_n, t||_{ext})$. Consequently, $||t_1, \ldots, t_n, t, s_1', \ldots, s_j'||_{ext}$ is bounded by $p(||t_1, \ldots, t_n, t||_{ext}) \cdot (p'(||t_1, \ldots, t_n, t||_{ext}) + 1)$ where p' is the polynomial bounding l. By Lemma 35, this shows that $\mathcal{E}_j$ is polynomially bounded in $||t_1, \ldots, t_n, t||_{ext}$. Lemma 32 and 23 now imply that $\mathcal{E}$ is polynomially bounded in $||t_1, \ldots, t_n, t||_{ext}$. □

Proposition 3 *DH rules allow polynomial product exponent attacks.*

PROOF. Let (π, σ) be a minimal attack on P. Let σ^Z be σ where all product exponents are replaced by new variables. Let β assign to every of these variables the corresponding product exponent. Thus, $\sigma(x) = \beta(\sigma^Z(x))$ for every $x \in \mathcal{V}(P)$. Note that, due to Theorem 3, $|\sigma^Z|$ can polynomially be bounded in $|P|$. Since the product exponents ini σ^Z are variables, we have that $||\sigma^Z||_{exp} \leq |\sigma^Z|^2$. Thus, $||\sigma^Z||_{ext}$ can polynomially be bounded in $||P||_{ext}$.

Let k, $R_1, \ldots, R_k$, $S_0, \ldots, S_k$ be defined as usual. W.l.o.g. we assume that S_0 is a single message instead of a set of messages. (Otherwise, represent $S_0 = \{a_1, \ldots, a_n\}$ by the term $\langle a_1, \langle a_2 \ldots \langle a_{n-1}, a_n\rangle \cdots\rangle\rangle$.)

Define $R_{k+1} =$ secret. We know that $\ulcorner\beta(R_i\sigma^Z)\urcorner \in$ *forge*$(\ulcorner\beta(S_0\sigma^Z)\urcorner, \ldots, \ulcorner\beta(S_{i-1}\sigma^Z)\urcorner$, for every $1 \leq i \leq k+1$.

By Lemma 36, for every i there exists an extension of β (the different extensions are independent from each other) and an equation system $\mathcal{E}_i$ such that

- $\beta \models \mathcal{E}_i$,
- $\ulcorner \beta'(R_i\sigma^Z) \urcorner \in forge(\ulcorner \beta'(S_0\sigma^Z) \urcorner, \ldots, \ulcorner \beta'(S_{i-1}\sigma^Z) \urcorner$ for every $\beta' \models \mathcal{E}_i$, and
- the size of $\mathcal{E}_i$ is polynomially bounded in $||S_0\sigma^Z, \ldots, S_k\sigma^Z, R_1\sigma^Z, \ldots, R_{k+1}\sigma^Z||_{ext}$ which in turn can polynomially be bounded in $||P||_{ext}$.

Consequently, $\beta \models \bigcup_{i=1}^{k} \mathcal{E}_i =: \mathcal{E}$, and thus, $\mathcal{E}$ is solvable, and for every $\beta' \models \mathcal{E}$ we have that $(\pi, \beta'(\sigma^Z))$ is an attack on P. By [2], there exists a solution β' of $\mathcal{E}$ where the binary representation of the integers can polynomially be bounded in the size of $\mathcal{E}$, and thus, by Lemma 36, polynomially be bounded in $||P||_{ext}$. We define σ' to be $\beta'(\sigma^Z)$. Then, (π, σ') is an attack on P. Also, $\sigma \approx \sigma'$, i.e., σ and σ' only different in the product exponents, and the $||\sigma'||_{exp}$ is polynomially bounded in $||P||_{ext}$, and thus, by Lemma 22, in $||P||$. □

As in [13], one shows that INSECURE is NP-hard in presence of the DH intruder. As an immediate consequence of this Proposition 3, Proposition 2, and Theorem 1 we obtain:

Theorem 4 *The problem* INSECURE *is NP-complete for the DH intruder.*

7 The A-GDH.2 Protocol

We refer the reader to [12] for a detailed description of the A-GDH.2 protocol. Let $\mathrm{P} = \{1, \ldots, n, I\}$ be the set of principals that may be involved in a run of a A-GDH.2 protocol where I is the name of the intruder (who can be both a legitimate participant and a dishonest principal). Any two principals $i, j \in \mathrm{P}$ share a long-term secret key $K_{i,j} (= K_{j,i})$. In a protocol run, a group $G \subseteq \mathrm{P}$ of principals (membership to a group may vary from one run to another) establish a session key that at the end of the protocol run is only known to the members of the group as long as all members in G are honest (*implicit key authentication*). In a run, one principal plays the role of the so-called *master*. Assume for example that $A, B, C, D \in \mathrm{P}$ want to share a session key and that D is the master. Then, A sends a message to B, B sends a message to C, and C sends a message to the master D. Then, D computes the session key for himself and also broadcasts keying material to A, B, and C using the long-term secret keys shared with these principals who from this material can each derive the session key. We call A the first, B the second, and C the third member of the group.

We now give a formal specification of the protocol in our protocol model. We abbreviate terms $\langle t_1, \langle t_2 \cdots \langle t_{n-1}, t_n \rangle \cdots \rangle \rangle$ by $t_1, \ldots, t_n$. We will define protocol rules $\Pi^{p,j}_{i,l,p'}$ which describe the lth step, $l \in \{1, 2\}$, of principal $p \in \mathrm{P}$, in the jth instance of p, $j \geq 0$, acting as the ith member of the group in which $p' \in \mathrm{P}$ is the master. The relation $\Pi^{p,j}_{i,1,p'} < \Pi^{p,j}_{i,2,p'}$ is the only partial order relationship between protocol rules. By $r^{p,j}$ we denote a random number (an atomic message) generated by p in instance j, and $\mathsf{secret}^{p,j}$ denotes a secret (some atomic message) of p in instance j. We define $\Pi^{p,j}_{1,1,p'}$, i.e., the first step of p in instance j acting as the first member of the group (i.e., the initiator of the protocol) where p' is the master to be

$$1 \Rightarrow \alpha, Exp(\alpha, r^{p,j})$$

where α is a group generator (an atomic message), and for $i > 1$ we define $\Pi^{p,j}_{i,1,p'}$ to be

$$x_1^{p,j}, \ldots, x_i^{p,j} \Rightarrow Exp(x_1^{p,j}, r^{p,j}), \ldots, Exp(x_{i-1}^{p,j}, r^{p,j}), x_i^{p,j}, Exp(x_i^{p,j}, r^{p,j})$$

where the $x_k^{p,j}$ are variables. The second step $\Pi^{p,j}_{i,2,p'}$ of p in instance j as ith member, $i > 0$, is the protocol rule

$$y^{p,j} \Rightarrow \{\mathsf{secret}^{p,j}\}^s_{Exp(y^{p,j}, r^{p,j} \cdot K^{-1}_{p,p'})}.$$

Note that $Exp(y^{p,j}, r^{p,j} \cdot K_{p,p'}^{-1})$ is the session key computed by p and that implicit key authentication requires that no principal outside of the group can get hold of $\mathsf{secret}^{p,j}$. We now define the protocol rule $M^{p,j}_{p_1 \cdots p_h}$ which describes principal $p \in \mathrm{P}$ in the jth instance acting as master for the group $p_1, \ldots, p_h, p \in \mathrm{P}$ (in this order) where p is the last member of that group. We define $M^{p,j}_{p_1 \cdots p_h}$ to be

$$z_1^{p,j}, \ldots, z_{h+1}^{p,j} \Rightarrow Exp(z_1^{p,j}, r^{p,j} \cdot K_{p_1,p}), \ldots, Exp(z_h^{p,j}, r^{p,j} \cdot K_{p_h,p}), \{\mathsf{secret}^{p,j}\}^s_{Exp(z_{h+1}^{p,j}, r^{p,j})}$$

where the $z_k^{p,j}$ are variables, $Exp(z_k^{p,j}, r^{p,j} \cdot K_{p_k,p})$ is the keying material for p_k, and the message $Exp(z_{h+1}^{p,j}, r^{p,j})$ is the session key computed by the master p.

The following protocol P describes two sessions of the A-GDH.2 protocol one for the group $p, p', I, p'' \in \mathrm{P}$ and one for the group p, p', p'' where in both cases p'' is the master of the group. Note that in the first instance, the actions of the intruder I need not be defined. Formally, the set of protocol rules in P consists of the rules describing p in the first session $\Pi^{p,1}_{1,1,p''}, \Pi^{p,1}_{1,2,p''}$ (note that $\Pi^{p,1}_{1,1,p''} < \Pi^{p,1}_{1,2,p''}$), the rules for p' in the first session $\Pi^{p',1}_{2,1,p''}, \Pi^{p',1}_{2,2,p''}$, and the master $M^{p'',1}_{pp'I}$ of the first session. The protocol rules of the second session are $\Pi^{p,2}_{1,1,p''}, \Pi^{p,2}_{1,2,p''}, \Pi^{p',2}_{2,1,p''}, \Pi^{p',2}_{2,2,p''}, M^{p'',2}_{pp'}$. The initial intruder knowledge is $\{\alpha, r^{I,1}\} \cup \{K_{pI} \mid p \in \mathrm{P}\}$. Let secret be some of the secrets returned by p or p' in the second session. Note that since the intruder is not a member of the group of the second session, he should not be able to obtain secret. However, as shown in [12], there exists an attack on P. It is easy to verify that this attack will be found by our decision procedure.

8 Conclusion

We have shown that a class of protocols with Diffie-Hellman exponents can be automatically analyzed. We conjecture that RSA encryption can easily be integrated into this framework: We have to limit the intruder to only use non-negative product exponents when performing the exponentiation operation and to allow terms of the form t^{-1} (which stands for t's private key) outside of exponents. More generally, in future work we plan to consider the case of arbitrary products outside of exponents and to allow the intruder to perform abelian group operators outside of exponents.

References

[1] R.M. Amadio, D. Lugiez, and V. Vanackere. On the symbolic reduction of processes with cryptographic functions. *Theoretical Computer Science*, 290(1):695–740, 2002.

[2] A. Bockmayr and V. Weispfenning. Solving numerical constraints. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, chapter 12, pages 751–842. Elsevier Science, 2001.

[3] M. Boreale and M.G. Buscemi. On the Symbolic Analysis of Low-Level Cryptographic Primitives: Modular Exponentiation and the Diffie-Hellman Protocol. In *In Proceedings of the Workshop on Foundations of Computer Security (FCS 2003)*, 2003.

[4] Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. An NP Decision Procedure for Protocol Insecurity with XOR. In *Proceedings of LICS 2003*, 2003.

[5] H. Comon and V. Shmatikov. Is it possible to decide whether a cryptographic protocol is secure or not? *Journal of Telecommunications and Information Technolog*, 2002.

[6] H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *Proceedings of LICS 2003*, 2003.

[7] D. Dolev and A.C. Yao. On the Security of Public-Key Protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.

[8] D. Kapur, P. Narendran, and L. Wang. Analyzing protocols that use modular exponentiation: Semantic unification techniques. In *Proceedings of RTA 2003*, 2003.

[9] C. Meadows and P. Narendran. A Unification Algorithm for the Group Diffie-Hellman Protocol. In *Workshop on Issues in the Theory of Security (WITS 2002)*, 2002.

[10] J. Millen and V. Shmatikov. Symbolic Protocol Analysis with Products and Diffie-Hellman Exponentiation. In *Proceedings of the 16th IEEE Computer Security Foundations Workshop (CSFW 16)*, 2003.

[11] J. K. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *Proceedings of the 8th ACM conference on Computer and Communications Security*, pages 166–175. ACM Press, 2001.

[12] O. Pereira and J.-J. Quisquater. A Security Analysis of the Cliques Protocols Suites. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 73–81, 2001.

[13] M. Rusinowitch and M. Turuani. Protocol Insecurity with Finite Number of Sessions is NP-complete. In *14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 174–190, 2001.