ARSPA 2004 Preliminary Version

Deciding the Security of Protocols with Commuting Public Key Encryption

Y. Chevalier[1], R. Küsters[2], M. Rusinowitch[1] and M. Turuani[1] ⋆,⋆⋆

[1] *LORIA-INRIA, France*
54506 Vandoeuvre-les-Nancy cedex, France
email: {chevalie,rusi,turuani}@loria.fr
fax: (33) 3.83.27.83.19

[2] *Department of Computer Science, Stanford University, USA*
Stanford University, Stanford CA 94305, USA
email: kuesters@theory.stanford.edu
fax: (1) 650 725-4671

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 (see e.g. [1,15,5,10]). However, it is an open question whether this result remains valid when the intruder model is extended to take into account even simple algebraic properties of low-level cryptographic primitives. 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., [16,14]).

Only recently the perfect encryption assumption for protocol analysis has been slightly relaxed. In [12], 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 [7,8], decidability of security has been proved for protocols that employ exclusive or. In [6], we have extended this result to protocols that are based on Diffie-Hellman exponentiation. Diffie-Hellman exponentiation has also been studied in [13] and [3]. However, in the former work no decision procedure is provided

⋆ This work was partially supported by IST AVISPA `http://www.avispa-project.org/`
⋆⋆This work was partially supported by PROCOPE and the DFG.

This is a preliminary version. The final version will be published in
Electronic Notes in Theoretical Computer Science
URL: `www.elsevier.nl/locate/entcs`

and in the latter severe restrictions are imposed on the protocol and intruder model.

In this paper, we show that the insecurity problem for protocols that use commuting public-key encryption operators (such as RSA encryption with common modulus) admits an NP decision procedure for a finite number of sessions (see the main result in Section 4). In Section 2, we present a very simple protocol illustrating that protocols and attacks on these protocols may rely on the commutativity of encryption.

This problem can be related to the analysis of Diffie-Hellman protocols as studied in [6] since Diffie-Hellman exponentiation and commuting public-key encryption, which in case of RSA also involves exponentiation, share algebraic properties. However, there are significant differences.

First, the intruder capabilities differ. In case of public-key encryption the intruder is not able to compute the inverse of exponents, e.g., given a public key (n, e) and a cipher text $c = m^e \bmod n$, the intruder can not compute the private key d and then by computing $c^d \bmod n$ obtain m. Conversely, in the Diffie-Hellman setting, exponentiation is done modulo a publicly known prime, and thus, it is computationally feasible to compute the inverse of exponents, e.g., given $m = g^{a \cdot b}$ and b where g generates the multiplicative group induced by the prime p, the intruder can easily invert b modulo $p - 1$ obtaining b^{-1} (in case an inverse exists) and by computing $m^{b^{-1}}$ obtain g^a.

Second, in [6] the intruder can not obtain inverses of messages, such as b^{-1} directly, but only use them in exponents. However, in the public-key setting we consider here, this would be unrealistic since inverses correspond to private keys, and of course, we need to allow the intruder to possess such keys (own private keys and private keys of dishonest principals).

As a result of these differences, the proofs differ as well. First, while roughly speaking in [6] we reduce the insecurity problem to solving linear equations in integers, we now obtain linear equations in non-negative integers. Also, we need to extend the intruder to allow private keys in her possession. To minimize the changes necessary compared to the proof in [6], we consider private keys as atomic messages and extend the normalization function to make sure that in exponents public and private keys cancel each other out. This allows us to lift the proofs presented in [6] to the setting considered here.

2 Examples of Protocols Relying on Commutative Encryption

Let us illustrate by two simple examples given in [17] how commutative properties of public key encryption schemes are employed in cryptographic protocols.

The first protocol is due to Shamir. The aim of this protocol is to permit secure communication between two agents who neither share a symmetric key nor know the public key of the other agent. The protocol uses the commutativity property of the RSA encryption scheme:

$$\begin{array}{ll} 1.\ A \rightarrow B: & \{\mathsf{secret}\}^p_{K_A} \\ 2.\ B \rightarrow A: & \{\{\mathsf{secret}\}^p_{K_A}\}^p_{K_B} \\ 3.\ A \rightarrow B: & \{\mathsf{secret}\}^p_{K_B} \end{array}$$

In this protocol, a common RSA modulus n is assumed. The public key of A is (n, K_A) and the one for B is (n, K_B). The message secret is some non-negative integer $< n$. The term $\{\mathsf{secret}\}^p_{K_A}$ stands for secret^{K_A} mod n. By the algebraic properties of exponentiation, we have that $\{\{\mathsf{secret}\}^p_{K_A}\}^p_{K_B} = \{\{\mathsf{secret}\}^p_{K_B}\}^p_{K_A}$. In step 3 of the protocol, A computes $\{\{\{\mathsf{secret}\}^p_{K_A}\}^p_{K_B}\}^p_{K'_A} = \{\mathsf{secret}\}^p_{K_B}$ where K'_A is A's private key. Thus, the protocol itself uses the commutativity of encryption. Since B is not authenticated in this protocol, it is obvious that the intruder I can impersonate B, by simply playing B's role while using her own public key K_I.

A commutative public key encryption scheme or signature scheme may also be relevant in the case of group protocols. Inspired by the protocol given in [17], Chapter 23, consider a group of l agents. A trusted server generates two large prime numbers p and q, computes $n = p \cdot q$, and $l + 1$ numbers $k_0, \ldots, k_l$ such that:

$$k_0 \cdots k_l \equiv 1 \bmod (p-1) \cdot (q-1)$$

Each agent A_i, $1 \leq i \leq l$, receives the public keys $K_j = k_0 \cdot \ldots \cdot k_l \cdot {k_j}^{-1}$ for every j and the private key k_i. Note that

$$\{M\}^p_{k_0 \cdots k_l} = M,$$

and in particular,

$$\{\{M\}^p_{k_i}\}^p_{K_i} = M.$$

Once the key distribution is completed, a message can be signed by a subset $\{A_i\}_{i \in I, I \subseteq [1,..,l]}$ of the members of the group. For example, suppose $l = 4$ and A_1 wants to sign a contract, say the message M, with A_2 and A_4. A possible message sequence is:

$$\begin{array}{ll} 1.\ A_1 \rightarrow A_2: & \{M\}^p_{k_1} \\ 2.\ A_2 \rightarrow A_4: & \{\{M\}^p_{k_1}\}^p_{k_2} \\ 3.\ A_4 \rightarrow A_1: & \{\{\{M\}^p_{k_1}\}^p_{k_2}\}^p_{k_4} \end{array}$$

On receiving the second message, A_4 can verify the signatures and identity the agents that have signed M by testing whether

$$\{\{\{\{M\}^p_{k_1}\}^p_{k_2}\}^p_{K_1}\}^p_{K_2} = \{\{\{\{M\}^p_{k_1}\}^p_{K_1}\}^p_{k_2}\}^p_{K_2} = M.$$

Agent A_4 can then also sign the contract using her private key k_4. The point here is that due to the commutativity property, A_4 does not need to know in what order the agents signed the message. Certainly, this protocol, when for instance used as a contract signing protocol, has many problems, which, however, we do not want to discuss here.

3 The Protocol and Intruder Model

The protocol and intruder model we describe here extend standard models for automatic analysis of security protocols in two respects. First, messages can be built using the operator $\{\_\}^p_\_$, which stands for encryption by a multiset of public/private keys described as a product of public/private keys. For instance, we have that $\{\{m\}^p_{K_A}\}^p_{K_B} = \{m\}^p_{K_A \cdot K_B} = \{m\}^p_{K_B \cdot K_A} = \{\{m\}^p_{K_B}\}^p_{K_A}$. In particular, we can model the commutativity of public key encryption. Second, in addition to the standard Dolev-Yao intruder capabilities, the intruder is equipped with the ability to perform this generalized encryption with any set of public or private keys she knows (we even allow arbitrary messages). For instance, if she happens to know A's private key K'_A and the message $c = \{m\}^p_{K_A}$, then she can compute $\{c\}^p_{K'_A} = \{m\}^p_{K_A \cdot K'_A} = \{m\}^p_1 = m$. In what follows, we provide a formal definition of our model by defining terms, messages, protocols, the intruder, and attacks.

3.1 Terms and Messages

The set of terms *term* is defined as the union of *roots* (also called *standard terms*) and *products* (also called *non-standard terms*) in the following grammar:

$$\begin{aligned} \mathit{root} &::= \mathcal{A} \mid \mathcal{V} \mid \langle \mathit{root}, \mathit{root} \rangle \mid \{\mathit{root}\}^s_{\mathit{root}} \mid \{\mathit{root}\}^p_{\mathit{product}} \\ \mathit{product} &::= \mathit{root}^{\mathbb{N}} \mid \mathit{root}^{\mathbb{N}} \cdot \mathit{product} \end{aligned}$$

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 $\mathbb{N}$ is the set of non-negative integers. We assume that there is a bijection $\cdot'$ on $\mathcal{K}$ which maps every public (private) key k to its corresponding private (public) key k'. 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 standard term and that for public key encryption the key can be any non-standard term (product). The non-negative integers occurring in products are called *product exponents*.

Envision a term t as a tree structure where each internal node is labelled by a constructor and each leaf is either a variable or a constant. A term u is a *subterm* if the tree representing u is a subtree of the tree representing t and if u is a root term. We note $\mathrm{Sub}(t)$ the set of subterms of a term t. By extension if E is a set of terms we note $\mathrm{Sub}(E)$ the union for $t \in E$ of the sets $\mathrm{Sub}(t)$.

The *size* of a term t is denoted $|t|$ and is the size of the representation of t by a labelled DAG (*Directed Acyclic Graph*). It is linear with respect to the cardinal of $\mathrm{Sub}(t)$ plus the space needed to represent the coefficients in binary.

A *ground term* (also called *message*) is a term without variables. Like a term, it can be *standard* or *non-standard.* 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 expected.

We now formulate the algebraic properties of terms. Besides commutativity and associativity of the product operator we consider the following properties where t is a standard term, M_1, M_2 are products, $k \in \mathcal{K}$, k' is the corresponding inverse key to k, and z, z' are non-negative integers:

$$
\begin{array}{lll}
t^1 = t & t \cdot 1 = t & \{t\}^p_1 = t \\
t^0 = 1 & t^z \cdot t^{z'} = t^{z+z'} & \{\{t\}^p_{M_1}\}^p_{M_2} = \{t\}^p_{M_1 \cdot M_2} \\
1^z = 1 & k \cdot k' = 1 &
\end{array}
$$

A *normal form* $\ulcorner t \urcorner$ of a term t is obtained by exhaustively applying these identities from left to right. Note that $\ulcorner t \urcorner$ is uniquely determined up to commutativity and associativity of the product operator. Two terms t and t' are *equivalent* if $\ulcorner t \urcorner = \ulcorner t' \urcorner$. The notion of normal form extends in the obvious way to sets of terms and substitutions. We illustrate the notion of a normal form by some examples: If $a, b, c, d \in \mathcal{K}$, then i) $\ulcorner (a^2 \cdot b^1) \cdot b'^2 \urcorner = a^2 \cdot b'$, ii) $\ulcorner \{\{a\}^p_{b^1 \cdot c^1}\}^p_{c' \cdot d'^2} \urcorner = \{a\}^p_{b \cdot d'^2}$, and iii) $\ulcorner \{\{a\}^p_{b^3 \cdot c'^6 \cdot b'^3}\}^p_{c^6} \urcorner = a$. Recall that, for instance, b' denotes the decryption key corresponding to b.

One easily shows:

3.2 Protocols

Protocols are defined as follows.

Definition 3.1 A *protocol rule* is of the form $R \Rightarrow S$ where R and S are standard terms.

A *protocol* P is a tuple $(\{R_i \Rightarrow S_i \mid i \in \mathcal{I}\}, <_\mathcal{I}, E)$ 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

(i) the (standard) terms R_i and S_i are normalized,

(ii) for all $x \in \mathcal{V}(S_i)$, there exists $j \leq_\mathcal{I} i$ such that $x \in \mathcal{V}(R_j)$, and

(iii) for every subterm $\{t_1\}^p_{t_2^{z_2} \cdots t_n^{z_n}}$ of R_i, there exists $r \in \{1, \ldots, n\}$ such that $\mathcal{V}(t_l) \subseteq \cup_{j <_\mathcal{I} i} \mathcal{V}(R_j)$ for every $l \in \{1, \ldots, n\} \setminus \{r\}$.

Condition 1. in the above definition is not an actual restriction. One obtains an equivalent protocol (in the sense that the same attacks are possible) by normalizing the R_i and S_i in the protocol rules in case they are not normalized already. Roughly speaking, Condition 2. guarantees that a principal can only output messages she has learned before. Finally, Condition 3. ensures that every single protocol rule can be applied deterministically to an input message. These conditions do not seem to exclude realistic protocols. See [6] for more detailed remarks on the above conditions. Note that in our model, a *protocol* corresponds to a specification in the Alice&Bob notation *together* with an instantiation. As a result, several sessions are modelled as only one protocol.

In our protocol model, the RSA protocol (Section 2) can formally be stated as follows where we assume that A runs one instance of the protocol as initiator and B runs one instance as responder. The protocol consists of three protocol rules denoted $(A, 1)$, $(A, 2)$, and $(B, 1)$ with

$$(A, 1):\ 1 \Rightarrow \{\mathsf{secret}\}^p_{K_A},$$
$$(A, 2):\ x \Rightarrow \{x\}^p_{K'_A}, \quad \text{and}$$
$$(B, 1):\ y \Rightarrow \{y\}^p_{K_B}$$

where $(A, 1)$ and $(A, 2)$ denote the first and second protocol step performed by A, respectively, and $(B, 1)$ denotes B's protocol step. The partial ordering is $<= \{((A, 1), (A, 2))\}$, i.e., it only satisfies $(A, 1) < (A, 2)$, and thus, makes sure that $(A, 1)$ must be performed before $(A, 2)$. The initial intruder knowledge is $\{1, K_I, K'_I\}$, i.e., besides the constant 1, the intruder knows her public and private key.

3.3 The Intruder Model and Attacks

Given a finite normalized set E of messages, the (infinite) set $\mathit{forge}(E)$ of messages the intruder can derive from E is the smallest set satisfying the following conditions:

(i) $E \subseteq \mathit{forge}(E)$.

(ii) If $\langle m, n\rangle \in \mathit{forge}(E)$, then $m \in \mathit{forge}(E)$ and $n \in \mathit{forge}(E)$. Conversely, if $m, n \in \mathit{forge}(E)$, then $\langle m, n\rangle \in \mathit{forge}(E)$.

(iii) If $\{m\}^s_n \in \mathit{forge}(E)$ and $n \in \mathit{forge}(E)$, then $m \in \mathit{forge}(E)$. Conversely, if $m, n \in \mathit{forge}(E)$, then $\{m\}^s_n \in \mathit{forge}(E)$.

(iv) If $m, m_1, \ldots, m_n \in \mathit{forge}(E)$, and $z_1, \ldots, z_n \in I\!N$ then $\ulcorner\{m\}^p_{m_1^{z_1} \cdots, m_n^{z_n}}\urcorner \in \mathit{forge}(E)$.

While the first three conditions are standard for Dolev-Yao intruders, the last condition, which gives the intruder the ability to perform commuting public key encryption, is new. We call this extended intruder the *RSA intruder*. Note that by performing pubic key encryption, the intruder can both encrypt and decrypt messages. Also note that if E is a set of normalized messages, then so is $\mathit{forge}(E)$.

We now define attacks. In an attack on a protocol P, the intruder (nondeterministically) chooses some execution order for P, i.e., a linearization of the protocol rules which is compatible with the partial ordering, 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.

Definition 3.2 Let $P = (\{R_j \Rightarrow S_j \mid j \in \mathcal{I}\}, <_{\mathcal{I}}, S_0)$ be a protocol. Then an *attack* on P consists of a linearization $\pi : R_1 \Rightarrow S_1, \ldots, R_k \Rightarrow S_k$ of the protocol rules in P assuming $k = \mathsf{Card}(I)$ which is compatible with $<_{\mathcal{I}}$ and a normalized ground substitution σ of the variables occurring in P such that

(i) $\ulcorner R_i\sigma \urcorner \in \mathit{forge}(\ulcorner S_0, S_1\sigma, ..., S_{i-1}\sigma \urcorner)$ for every $i \in \{1, \ldots, k\}$ and

(ii) $\mathsf{secret} \in \mathit{forge}(\ulcorner S_0, S_1\sigma, ..., S_k\sigma \urcorner)$.

In Definition 3.2 we restrict ourselves to $k = \mathsf{Card}(I)$ for simplifying notations. Considering $k < \mathsf{Card}(I)$ would amount to detect attack even with partial (i.e. unfinished) sessions. This kind of attacks can be captured too by analyzing protocols with some final steps removed.

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

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

It can easily be checked that the RSA-protocol as formally described in Section 3.2 is insecure according to our definition.

4 Main Theorem

The main result of this paper is the following:

Theorem 4.1 *For the RSA intruder, the problem* Insecure *is NP-complete.*

As mentioned in the introduction, the proof follows the same lines as the one for Diffie-Hellman exponentiation presented in [6]. Here we only provide a very brief proof sketch.

NP-hardness can easily be established (see for instance [1]). The decision procedure first guesses an execution order of the protocol rules and a ground substitution σ of size polynomially bounded in the size of the protocol, and then checks whether condition 1. and 2. of Definition 3.2 are met. This check can be done in polynomial time.

The completeness and the complexity of this procedure can be proved in two steps. First, we prove that it is possible to check in polynomial time w.r.t. the sizes of $\mathcal{P}$ and σ whether a couple (π, σ) is an attack. Then we prove it is sufficient to consider substitutions σ of polynomial size w.r.t. the size of the protocol in order to prove whether there exists attacks on a given protocol $\mathcal{P}$.

4.1 Deciding whether (π, σ) is an attack

We want to have a procedure with a time complexity polynomial w.r.t. the sizes of $\mathcal{P}$ and σ. Since in Definition 3.2 we have $k \leq |\mathcal{P}|$, it is sufficient to prove that:

(i) for all terms t and all substitutions σ, we have:

$$|\ulcorner t\sigma \urcorner| \leq |t| + |\sigma|$$

(ii) for every set of terms E and for all term m, it can be decided in polynomial time w.r.t. $|E, t|$ if m is in *forge*(E).

The first point is a consequence of the definition of the normalisation function. We give now a sketch of the proof of the second point.

The deduction power of the intruder can be modelled by an infinite set of ground deduction rules $l \to r$ where l is a set of terms and r is a term. A set L of such rules defines a transition relation between sets of terms as follows. Let E and E' be two sets of terms, then we have $E \to_L E'$ iff there exists $l \to r \in L$ such that $l \subseteq E$ and $E' = E, r$. The set *forge*(E) can then equivalently be defined as the closure of E by the rules of L.

Let $\mathcal{F}$ be the set of constructors $\{\langle \_, \_ \rangle, \{\_\}^s_\_, \{\_\}^p_\_\}$. The path followed in [4] is then to associate to each constructor f in $\mathcal{F}$ a system of rewrite rules L_f. The first result is that for each constructor $f \in \mathcal{F}$, the transition relation between two sets E and F for the set of rewrite rules L_f can be decided in polynomial time w.r.t. $|E, E'|$. We just give here the proof for the commutative encryption operator $\{\_\}^p_\_$.

Proposition 4.2 *Let E and E' be two sets of terms and $f = \{\_\}^p_\_$. Then $E \to_{L_f} E'$ can be decided in polynomial time w.r.t. $|E, E'|$.*

Proof. First, we check that $E \subseteq E'$ and $E' \setminus E = \{t\}$. Then let r be the root of t, and E_r be the subset of E of terms of root r. For each $u \in E_r$, compute p_u, the product of t divided by the product of u. If each root term v in p_u is in E, answer yes.

If no check succeeds, answer no. $\square$

Example.

Let $E = a, \{a\}^p_{k_a \cdot k'_c}, k_a, k'_a, k_b$ and $F = E \cup \{\{a\}^p_{k_b \cdot k'_c}\}$. We want to decide whether there exists a one-step transition from E to F. We have $E \subseteq F$ and $F \setminus E = \{\{a\}^p_{k_b}\} = \{t\}$. The root of t is the constant a. The subset of E of terms of root a is $E_a = a, \{a\}^p_{k_a \cdot k'_c}$.

- For a, we take the product equal to 1. Thus, $p_a = k_b \cdot k'_c$ and since $k'_c \notin E$, the check fails in this case;
- For $\{a\}^p_{k_a \cdot k'_c}$, we have:

$$p_{\{a\}^p_{k_a \cdot k'_c}} = (k_b \cdot k'_c)/(k_a \cdot k'_c) = k_b \cdot k'_c \cdot k'_a \cdot k_c = k_b \cdot k'_a$$

In this case, $k_b \in E$ and $k'_a \in E$, and thus the check succeeds. The procedure returns with the answer YES.

There is a transition from E to E' iff there exists $f \in \mathcal{F}$ such that $E \rightarrow_{L_f} E'$. Thus, the one-step transition problem can be decided in polynomial time for the set of rewrite rules $\mathrm{L} = \cup_{f\in\mathcal{F}} L_f$. We now consider the derivation problem:

$$\mathrm{DERIVE} := \{(E, m) \mid m \in \mathit{forge}(E)\}$$

where E is a finite set of normalized standard messages and m is a normalized standard message (both given as DAG). It is equivalent to decide DERIVE and to decide whether there exists n sets of terms $E_1, \ldots, E_n$ such that $E_1 = E$ and $m \in E_n$ and for all $i \in \{1, \ldots, n-1\}$ we have $E_i \rightarrow_{\mathrm{L}} E_{i+1}$. A fundamental result is that if there exists such a sequence of transitions, then there exists one such that $E_n \subseteq \mathrm{Sub}(E, m)$. The proof consists in considering a sequence of transitions of minimal length $E_1 \rightarrow_{\mathrm{L}} \ldots \rightarrow_{\mathrm{L}} E_n$ such that $m \in E_n$ and to prove that each transition adds a subterm of E or of m. Such a transition sequence is called a *normal proof* in [8] or a *well-formed derivation* in [7]. Combining this result with Proposition 4.2, one obtains the following proposition:

Proposition 4.3 *For the RSA intruder,* DERIVE *can be decided in deterministic polynomial time.*

Proof. (sketch) The procedure consists in computing $F = \mathit{forge}(E) \cap \mathrm{Sub}(E, m)$. We start with $F_0 = E$, and F_{i+1} is computed from F_i by adding all the terms t in $\mathrm{Sub}(E, m)$ such that there exists a transition between F_i and F_i, t. The computation stops as soon as $F_{i+1} = F_i$. The existence of well-formed derivation then implies $F_i = \mathit{forge}(E) \cap \mathrm{Sub}(E, m)$. We have to decide a polynomial number of times a one-step transition problem. Thus, the total running time is polynomial w.r.t. $|E, m|$. □

4.2 Bounds on $|\sigma|$

The involved part of the proof of Theorem 4.1 is to show that when there is an attack on a protocol P, then there exists an attack of size polynomially bounded in the size of the protocol. This proof is done in two steps. First, it is shown that the number of subterms occurring in σ can polynomially be bounded in the size of P. Note that this does not bound the size of product exponents in σ. Therefore, in a second step, it is shown that the size of product exponents can polynomially be bounded in the size of P. This is done as follows: Given an attack with substitution σ, the product exponents in σ are replaced by variables (taking non-negative integers) yielding a symbolic substitution σ^Z. Now, we associate a linear Diophantine equation system in non-negative integers (of polynomial size in P) with the attack which constraints the variables in σ^Z (and auxiliary variables) such that when instantiating σ^Z by a non-negative solution of the equation system this also gives an attack on P. By [2], the size of the solutions can polynomially be bounded in the size of the equation system, and thus, P. This shows that the size of product exponents can polynomially be bounded in the size of P.

5 Conclusion

We have shown that the security problem for a class of protocols with commuting public key encryptions is in NP. This result was obtained by a reduction to the satisfiability of linear diophantine equations on $I\!N$. The result generalizes easily to the more common case where the set of keys yielding commuting encryption is a subset of the set of all keys. It would be interesting to characterize a class of algebraic properties that can be captured by our approach.

References

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

[2] I. Borosh and L.B. Treybig. Bounds on positive integral solutions of linear Diophantine equations. In Proc. Amer. Math. Soc. 55, 299-304 (A6), 1976.

[3] M. Boreale and M.G. Buscemi. On the symbolic analysis of low-level cryptographic primitives: Modular exponentiation and the Diffie-Hellman protocol. In *Proc. of FCS 2003*.

[4] Y. Chevalier Résolution de problèmes d'accessibilité pour la compilat ion et la validation de protocoles cryptographiques Mémoire de thèse, Université Henri Poincaré Nancy 1, 2003.

[5] Y. Chevalier and L. Vigneron. Towards Efficient Automated Verification of Security Protocols. In Proceedings of the Verification Workshop (VERIFY'01) (in connection with IJCAR'01), Università degli studi di Siena, TR DII 08/01, pages 19–33, 2001.

[6] Y. Chevalier, R. Küsters, M. Rusinowitch, and M. Turuani. Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Products in Exponents. In *Proc. of FSTTCS 2003*.

[7] Y. Chevalier, R. Kuesters, M. Rusinowitch, and M. Turuani. An NP Decision Procedure for Protocol Insecurity with XOR. In *Proceedings of the Logic In Computer Science Conference LICS'0 3*, June 2003. Long version available as Technical Report RR-4697, INRIA, Fr ance.

[8] H. Comon-Lundh and V. Shmatikov. Intruder Deductions, Constraint Solving and Insecurity Decision in Presence of Exclusive or. In *Proceedings of the Logic In Computer Science Conference, LICS'03*, pages 271–280, 2003.

[9] R. Corin and S. Etalle An Improved Constraint-based system for the verification of security protocols, 9th Int. Static Analysis Symp. (SAS), LNCS 2477, M. V. Hermenegildo and G. Puebla (eds.) , published by Springer-Verlag, Berlin, held in Madrid, Spain, Sep. , 2002, pp. 326-341

[10] R. Küsters. On the Decidability of Cryptographic Protocols with Open-ended Data Structures. In *Proc. of CONCUR 2002*.

[11] R. Küsters and T. Wilke. Automata-based Analysis of Recursive Cryptographic Protocols 21st Symposium on Theoretical Aspects of Computer Science (STACS 2004), 2004, Lecture Notes in Computer Science, Springer-Verlag, pages 382–393.

[12] C. Meadows and P. Narendran. A unification algorithm for the group Diffie-Hellman protocol. In *Proc. of WITS 2002*.

[13] J. Millen and V. Shmatikov. Symbolic protocol analysis with products and Diffie-Hellman exponentiation. In *Proc. of CSFW 16*, 2003.

[14] O. Pereira and J.-J. Quisquater. A Security Analysis of the Cliques Protocols Suites. In *Proc. of CSFW-14*, pages 73–81, 2001.

[15] M. Rusinowitch and M. Turuani. Protocol Insecurity with Finite Number of Sessions is NP-complete. In *Proc. of CSFW-14*, pages 174–190, 2001.

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

[17] B. Schneier. *Applied Cryptography.* John Wiley & sons, New York, 1996.