Research Article

Alexei Myasnikov and Vitaliĭ Roman'kov

A linear decomposition attack

Abstract: We discuss a new attack, termed a *dimension* or *linear decomposition* attack, on several known group-based cryptosystems. This attack gives a polynomial time deterministic algorithm that recovers the secret shared key from the public data in all the schemes under consideration. Furthermore, we show that in this case, contrary to the common opinion, the typical computational security assumptions are not very relevant to the security of the schemes, i.e., one can break the schemes without solving the algorithmic problems on which the assumptions are based.

Keywords: Algebraic cryptography, Diffie–Hellman scheme, cryptanalysis, linear decomposition, linear representation

MSC 2010: 94A60, 20F10

DOI: 10.1515/gcc-2015-0007
Received January 8, 2015

1 Introduction

1.1 Motivation

In this paper, we discuss, following [37, 38], a new general attack, called the *dimension* or *linear decomposition* attack, on various group-based cryptosystems. We do cryptanalysis of ten principle protocols (see Section 4) while quite a few of others can be dealt similarly. The attack works when the platform groups (or semigroups, or algebras, or some other related objects) have efficient matrix representations (though we do not require here that the group embeds into the linear space homomorphically). Furthermore, the schemes themselves should be based on algorithmic problems of a particular type: the conjugacy search problem, the decomposition search problem, the factorization problem, automorphic actions – those inducing linear transformations on the underlying linear space. We show that, in this case, contrary to the common opinion (and some explicitly stated security assumptions), one does not need to solve the underlying algorithmic problems to break the schemes, i.e., there is another algorithm that recovers the private keys without solving the principal algorithmic problem on which the security assumptions are based. This changes completely our understanding of security of these schemes. Though the schemes are quite different the attack is uniform, it is based on the same idea from linear algebra related to linear decompositions and dimensions of the vector spaces at hands, hence the name. Our results are rather theoretical, we show that the attack breaks the keys in polynomial time in the size of the public data, but we do not try to get sharp estimates on the degrees of the polynomials, neither we do any implementations of the algorithms. The efficacy of the attack depends also on the platform groups, so sharp estimates would require a specific analysis in each particular case, which is out of the scope of the paper. We also discuss how to improve the schemes in order to fail the dimension attack. The easiest way would be to choose the platform groups which do not have efficient matrix representations. However, this would leave many interesting groups, in particular all finite groups, out of reach. On the other hand, the group might be linear, but the dimensions of its linear representations could be so large that the dimension attacks become inefficient. Not much in general is known about efficiency of linear representations

Alexei Myasnikov: Department of Mathematical Sciences, Stevens Institute of Technology, Castle Point, Hoboken, NJ 07030, USA, e-mail: amiasnikov@gmail.com
Vitaliĭ Roman'kov: Institute of Mathematics and Information Technologies, Omsk F. M. Dostoevsky State University, Mira, 55-A, Omsk 644077, Russia, e-mail: romankov48@mail.ru

of groups, most of the general problems are wide open. In particular, it would be very interesting to study minimal (and efficient) matrix representation of nilpotent and polycyclic groups, and their holomorphs, and especially finite groups.

1.2 Results and the structure of the paper

In Section 2, we describe some typical cryptosystems based on non-commutative groups (semigroups, algebras). Most of them are best understood as various generalizations of the classical Diffie–Hellman (DH) scheme. In particular, we discuss schemes based on the conjugacy search problem, the decomposition and factorization problems, and actions by automorphisms. It is worth while to mention here that the famous Anshel–Anshel–Goldfeld scheme [4] is not part of our discussion, since, on the one hand, it is not of Diffie–Hellman type, and on the other hand, it is currently under another powerful linear algebra attack from [47].

In Section 3, we explain the basic ideas of the linear decomposition attacks and provide some useful algorithms. The algorithms work in polynomial time in the size of the public data described in the schemes. All these results are rather theoretical: we did not try to improve neither on the efficacy of the algorithms nor on the complexity bounds, no doubts that both could be tighten up considerably.

In Section 4, we give cryptanalysis of a variety of group-based protocols of Diffie–Hellman type. There are two main results here. The first one claims that the linear decomposition attacks give a polynomial time deterministic algorithm that recovers the secret shared key from the public data in all the schemes we discuss in the section. The second result shows that in all these schemes the typical computational security assumptions are not very relevant to the security of the schemes, i.e., one can break the schemes without solving the algorithmic problems on which the assumptions are based. This is rather striking since the schemes together with the assumptions were known for some time and studied quite thoroughly.

1.3 Efficacy of linear decomposition attacks

As we have mentioned above, some of the DH-like schemes based on group-theoretic problems are susceptible to linear decomposition attacks, provided that the platform groups (or some other related groups) are linear. One of the possible ways to fail the attack is to use non-linear groups as platforms. However, this would leave behind all finite groups, which are known to be linear. Another way is to allow linear groups G as platforms, but only those whose smallest faithful linear representations $G \to \mathrm{GL}_n(\mathbb{F})$ have "prohibitively high" dimensions n, which makes dimension attacks implausible. Of course, if the platform group G is fixed then the dimension of any fixed linear representation of G is a constant, so the dimension attack still works in polynomial time (though the constants in the polynomials could be very large). To make the situation more clear one has to consider a class $\mathcal{C}$ of platform groups such that the dimension $n(G)$ of a smallest linear representation of G grows exponentially with respect to the size of the given description of G (for example, with respect to the size of the given finite presentation of G in terms of generators and relators). This approach accommodates naturally any infinite class $\mathcal{C}$ of finite groups.

Notice that the class of linear groups is very large. Besides finite groups it contains finitely generated nilpotent groups, arbitrary polycyclic groups, right angled Artin and braid groups, holomorphs of polycyclic groups, etc. Not all finitely generated metabelian groups are linear, but all of them admit faithful representations into finite direct products of general linear groups. Surprisingly, not much is known about the dimensions of smallest faithful linear representations of such infinite groups as nilpotent, or polycyclic, even less about metabelian groups. De Graaf and Nickel [15, 34] studied the classical linear representations of finitely generated torsion-free nilpotent groups G into $UT_n(\mathbb{Z})$ from the algorithmic view-point, showing in particular that there are polynomial time (in the Hirsh length of G) algorithms to construct the representations. Recently, Habeeb and Kahrobaei [17] showed, following [34], that the dimension of the classical representations above is $O(n^2)$ where n is the Hirsh length of the group. It seems the dimension is not prohibitively high in this case. However, there are nilpotent groups with presentations (in generators and relators) whose size is logarithmically smaller than their Hirsh length. Indeed, free nilpotent groups of class c and rank r have presentations with r generators and no relations in the variety of nilpotent groups of class at most c, but their Hirsh length is about r^{c+1}. On the other hand, to use a group G as a platform in cryptography requires a fast solution of the word problem in G, as well as fast algorithms for computing normal forms of group elements (see [33] for details). It seems the known algorithms for computing normal forms in nilpotent groups rely on so-called polycyclic presentations of nilpotent groups whose size is about the same as the Hirsh length of the group. The situation with finite nilpotent groups is even more interesting. Some upper bounds for dimensions of smallest linear representations of say finite p-groups are known: Janusz [22] showed that the minimal faithful representation of a finite p-group as a group of matrices over a finite field of characteristic p is $1 + p^{(e-1)}$ where e is the exponent of G, so the degree here is quite large (for some groups it is comparable with the order of the group).

The discussion above highlights several interesting open problems in algorithmic group theory, whose solution would shed some light on security of the corresponding cryptosystems.

2 Cryptoschemes under the dimension attack

Now we describe cryptosystems which to some extent are susceptible to the dimension attack. But first a few words on terminology. Most of the cryptoschemes (systems, protocols) discussed below were not originally stated within the formal rules of the current cryptographic practice (see, for example, the book [25] for definitions of a cryptosystem), usually some of the required algorithms or parameters are not completely described, or security assumptions are missing. In particular, it is hard to break such schemes because the precise description is lacking. However, they make perfect sense as general ideas or "general schemes" from which the concrete protocols should be worked out after some research on sorting out which parameters are strong and which are not. The research itself usually comes as a series of "attacks" on the scheme and the subsequent cryptanalysis. Our main intention in designing the dimension attack is not on breaking but on improving the generic schemes at hands. In what follows we are focusing on the choice of the so-called *platform group*, one of the main parameters in the group-based cryptosystems, and discuss how security of the scheme depends on the chosen platform. In particular, we shed some light on some known (computational) security assumptions.

Most of the schemes we discuss in this paper can be best seen in the light of the famous Diffie–Hellman (DH) key establishment protocol [12]. The main idea of DH is very simple and can be described as follows. Two users, say Alice and Bob, first choose the multiplicative group of integers $G = \mathbb{Z}_p^*$ modulo a prime number p, as the platform group, and some element $g \in G$ (all this data is public). Alice then selects a random integer $k \in \mathbb{N}$ (her private key) and sends g^k to Bob through a public channel. He in turn picks $l \in \mathbb{N}$ (his private key) and sends g^l to Alice through a public channel. Both Alice and Bob can then compute their secret shared keys $K = g^{kl}$. An adversary, say Eve, monitoring the transmission between Alice and Bob knows the public data G, g, g^k, g^l and her task is to recover the shared key K (i.e., to break the scheme). The *computational security assumption* claims that it is a time consuming task to recover the shared key from the public data. Namely, the claim is that for a fixed G, any probabilistic polynomial-time algorithm succeeds in breaking the scheme with only negligible probability. We refer to [25] for precise formulations. Observe that if Eve could compute k or l, then she could find K easily. Thus we arrive at the *underlying algorithmic problem*: recover Alice's (or Bob's) private key from the public data. In the case of Diffie–Hellman, the underlying algorithmic problem is the famous *discrete logarithm problem* for G, which asks whether or not one can compute (minimal non-negative) k from given g and g^k in probabilistic time. For other schemes the underlying algorithmic problems could be different. Notice that it is important how the group G and its elements are given: algebraically the same cyclic group of order p (and its elements) could be given as integers between 0 and $p - 1$, or as the group of an elliptic curve, or by a finite presentation. The scheme security depends on the presentation. Furthermore, the scheme still makes sense when the group $\mathbb{Z}_p^*$ is replaced by an arbitrary finite (or infinite) group G. This gives rise to a general DH scheme. Various platform groups G were suggested and studied, finite or not: the group of non-singular matrices over a finite field [11, 35], over a group algebra [24], over a semigroup [16], etc.

Now we discuss some group based cryptoschemes with respect to the algorithmic problems they are based on.

Schemes based on the conjugacy search problem. One of the possible generalizations of the general DH scheme to arbitrary non-commutative group is to use the conjugacy in the place of exponentiation. Recall that the conjugate of an element g by an element x in a group is defined by $g^x = xgx^{-1}$. The map $\phi_x : g \to g^x$ is an automorphism of G, called *conjugation*. One of the principal DH-type schemes based on the conjugation (instead of exponentiation) was introduced by Ko, Lee et al. in [26]. In this case, let G be a platform group and U, W be two finite subsets of G which are commuting element-wise, i.e., $uw = wu$ for any elements $u \in U$, $w \in W$. Denote by A and B the subgroups of G generated by U and W, respectively, and fix an element $g \in G$. All this data is assumed to be public. Then Alice picks a private element $a \in A$ and publishes g^a. Bob picks a private element $b \in B$ and publishes g^b. After that Alice computes a shared secret $K_A = (g^b)^a = g^{ab}$, while Bob computes the same element as $K_b = (g^a)^b = g^{ba} = g^{ab}$. In this scheme, the following is the underlying algorithmic problem (like the discrete log for DH scheme):

- *The conjugacy search problem (CSP) in a group G: given two elements $g, f \in G$ and information that $g^x = f$ for some $x \in G$, find at least one particular element x like that.*

The CSP plays a special role in group-based cryptography. Many protocols and cryptosystems based on groups use one or another variation of CSP. For example, the schemes in [16, 23, 26, 44, 48] use CSP; [4, 5] use the simultaneous CSP (when one has to solve in a group G a system of the type $g_1^x = f_1, \dots, g_k^x = f_k$).

Schemes based on the decomposition and factorization problems. Again, the DH scheme in a group G can be simulated by replacing exponentiation by right and left multiplication. One of the most typical schemes in this area is due to Shpilrain and Ushakov [42]. In the notation above, the idea of the scheme is as follows. A group G, two element-wise commuting subgroups A and B, and a fixed element $g \in G$ are given. Then Alice picks private elements $a, a' \in A$ and publishes the element aga'. Bob picks private elements $b, b' \in B$ and publishes the element bgb'. After that Alice computes a shared secret key $K_A = abgb'a'$, while Bob computes the same element as $K_B = baga'b' = abgb'a'$. In this scheme, the underlying algorithmic problem is the decomposition search problem.

- *The decomposition search problem (DSP) in a group G: given two subgroups $A, B \subseteq G$ and two elements $g, f \in G$, find elements $a \in A$ and $b \in B$ such that $a \cdot g \cdot b = f$, provided that at least one such pair of elements exists.*

There are two variations of DSP, which have been used in group-based cryptography:

- *The factorization search problem (FSP) in a group G: given an element $f \in G$ and two subsets (usually subgroups) A and B of G, find elements $a \in A$ and $b \in B$ such that $a \cdot b = f$.*
- *The power conjugacy search problem (PCSP) in a group G: given two elements $g, f \in G$ and information that $(g^k)^x = f$ for some $k \in \mathbb{N}$ and $x \in G$, find at least one particular pair (k, x) like that.*

The schemes in [1–3, 41–43, 45, 46] are based on DSP and FSP. The schemes in [23, 39] use PCSP in matrix groups.

Schemes using actions by automorphisms. One natural generalization of the DH scheme is to replace exponentiation by arbitrary commuting automorphisms of the group G. When the automorphisms are conjugations, one gets the scheme [26] described above. More precisely, let G be a group and $g \in G$. By $\mathrm{Aut}(G)$ we denote the group of automorphisms of G. For $g \in G$ and $\phi \in \mathrm{Aut}(G)$ we denote by g^ϕ the image of g under ϕ. Suppose that U, W are two finite subsets of $\mathrm{Aut}(G)$ commuting element-wise. Denote by A and B the subgroups in $\mathrm{Aut}(G)$ generated by U and W, respectively. Now Alice picks $a \in A$ and publishes g^a. Bob picks $b \in B$ and publishes g^b. Then Alice computes a secret shared key $K_A = (g^b)^a = g^{ba}$, while Bob computes the same element as $K_B = (g^a)^b = g^{ab} = g^{ba}$. The underlying algorithmic problem is the search automorphism problem:

- *The search automorphism problem in G: given a subgroup $A \leq \mathrm{Aut}(G)$ and two elements $g, h \in G$, find an automorphism $a \in A$ such that $g^a = h$, provided that such an automorphism exists.*

Schemes from [13, 16, 30] use the search automorphism (or endomorphism) problems in their design.

3 The principle idea

In this section, we describe the mathematical idea behind the linear decomposition attacks. Our exposition is closely linked to the "prototypical" schemes discussed in Section 2.

3.1 Finding a basis

Let V be a finite dimensional vector space over a field $\mathbb{F}$ with basis $\mathcal{B} = \{v_1, \dots, v_r\}$. Let $\mathrm{End}(V)$ be the semigroup of endomorphisms of V. We assume that elements $v \in V$ are given as vectors relative to $\mathcal{B}$, and endomorphisms $a \in \mathrm{End}(V)$ are given by their matrices relative to $\mathcal{B}$. For an endomorphism $a \in \mathrm{End}(V)$ and an element $v \in V$ we denote by v^a the image of v under a. Also, for any subsets $W \subseteq V$ and $A \subseteq \mathrm{End}(V)$ we put $W^A = \{w^a \mid w \in W,\ a \in A\}$, and denote by $\mathrm{Sp}(W)$ the subspace of V generated by W, and by $\langle A \rangle$ the submonoid generated by A in $\mathrm{End}(V)$.

The discussion below is about the time complexity of some algorithms. To this end we put some assumptions on computations in $\mathbb{F}$.

Computational assumption on the fields. *We assume that elements of the field $\mathbb{F}$ are given in some constructive form and the "size" of the form is defined. Furthermore, we assume that the basic field operations in $\mathbb{F}$ are efficient, in particular they can be performed in polynomial time in the size of the elements. In all the particular protocols considered in this paper the field $\mathbb{F}$ satisfies all these conditions.*

For an element $\alpha \in \mathbb{F}$ we write $\|\alpha\|$ for the size of α and put $\|v\| = \max \|\alpha_i\|$ for a vector $v = (\alpha_1, \dots, \alpha_r) \in V$, and $\|a\| = \max\{\|\alpha_{ij}\|\}$ for a matrix $a = (\alpha_{ij}) \in \mathrm{End}(V)$.

Lemma 3.1 (Principal lemma). *There is an algorithm that for given finite subsets $W \subseteq V$ and $U \subseteq \mathrm{End}(V)$ finds a basis of the subspace $\mathrm{Sp}(W^{\langle U \rangle})$ in the form $w_1^{a_1}, \dots, w_t^{a_t}$, where $w_i \in W$ and a_i is a product of elements from U. Furthermore, the number of field operations used by the algorithm is polynomial in $r = \dim_{\mathbb{F}} V$ and the cardinalities of W and U.*

Proof. Using Gauss elimination, one can effectively find a maximal linearly independent subset L_0 of W. Notice that $\mathrm{Sp}(L_0^{\langle U \rangle}) = \mathrm{Sp}(W^{\langle U \rangle})$. Adding to the set L_0 one by one elements v^a, where $v \in L_0$, $a \in U$ and checking every time linear independence of the extended set, one can effectively construct a maximal linearly independent subset L_1 of the set $L_0 \cup L_0^U$ which extends the set L_0. Notice that $\mathrm{Sp}(L_0^{\langle U \rangle}) = \mathrm{Sp}(L_1^{\langle U \rangle})$ and the elements in L_1 are of the form w^a, where $w \in W$ and $a \in \langle U \rangle$. It follows that if $L_0 = L_1$ then L_0 is a basis of $\mathrm{Sp}(W^{\langle U \rangle})$. If $L_0 \neq L_1$ then we repeat the procedure for L_1 and find a maximal linearly independent subset L_2 of $L_1 \cup L_1^U$ extending L_1. Keep going one constructs a sequence of strictly increasing subspaces $L_0 < L_1 < \cdots < L_i$ of V. Since the dimension r of V is finite, the sequence stabilizes for some $i \leq r$. In this case, L_i is a basis of $\mathrm{Sp}(W^{\langle U \rangle})$ and its elements are in the required form.

To estimate the upper bound of the number of the field operations used by the algorithm, observe first that the number of the field operations in Gauss elimination performed on a matrix of size $n \times r$ is $O(n^2 r)$. Hence it requires at most $O(n^2 r)$ steps to construct L_0 from W, where $n = |W|$ is the number of elements in W. Notice that $|L_j| \leq r$ for every j. So to find $L + j + 1$, it suffices to perform Gauss elimination on the matrix corresponding to $L_j \cup L_j^U$ which has size at most $r + r|U|$. Thus the upper estimate on this number is $O(r^3 |U|^2)$. Since there are at most r iterations of this procedure, one has the total estimate as $O(r^3 |U|^2 + r|W|^2)$. Of course, this estimate is very crude. □

Corollary 3.2. *With our assumptions on the field $\mathbb{F}$ the algorithm in Lemma 3.1 works in polynomial time in the size of the inputs, i.e., in $r = \dim_{\mathbb{F}} V$, $|W|$, $|U|$, and $\max\{\|w\|, \|u\| \mid w \in W,\ u \in U\}$.*

Notice that the algorithm described in Lemma 3.1 can be obviously adapted to noetherian modules over commutative rings.

3.2 The basic linear decomposition attack

Let as above V be a finite dimensional vector space over a field $\mathbb{F}$ with basis $\mathcal{B} = \{v_1, \dots, v_r\}$ and U and W be two finite subsets of $\mathrm{End}(V)$.

Commutativity assumption. *We assume that every element of U commutes with every element of W, i.e., for every $v \in V$, $u \in U$, $w \in W$ one has $v^{uw} = v^{wu}$.*

Let A and B be the submonoids of $\mathrm{End}(V)$ generated by U and W, respectively. Suppose that $a \in A$, $b \in B$ and $v \in V$. We assume that the field $\mathbb{F}$, the space V, the sets U, W and the vectors v, v^a, v^b are public, while the endomorphisms a and b are private. By the size of the public data we mean the total size of the following parameters: $r = \dim_{\mathbb{F}} V$ (given in unary, i.e., as 1^r), $|W|, |U|, \max\{\|w\|, \|u\| \mid w \in W,\ u \in U\}, \|v\|, \|v^a\|, \|v^b\|$.

Claim 1. *Given U, W, v, v^a, v^b, one can find in polynomial time (in the size of the public data) the vector $v^{ab} = v^{ba}$.*

Proof. Indeed, given U and v by Lemma 3.1 (and its corollary) one can find in polynomial time a basis of $\mathrm{Sp}(v^A)$ in the form $v^{a_1}, \dots, v^{a_t}$, where $a_i \in A$ given as some particular products of elements from U. Using Gauss elimination, one can decompose v^a as a linear combination in the given basis:

$$v^a = \sum_{i=1}^{t} \alpha_i v^{a_i}, \qquad \alpha_i \in \mathbb{F}.$$

This allows one to compute v^{ab} as follows:

$$v^{ab} = (v^a)^b = \left(\sum_{i=1}^{t} \alpha_i v^{a_i}\right)^b = \sum_{i=1}^{t} \alpha_i v^{a_i b} = \sum_{i=1}^{t} \alpha_i v^{b a_i} = \sum_{i=1}^{t} \alpha_i (v^b)^{a_i},$$

which is immediate, since the vector v^b and the matrices a_i are known.

The conclusion is that one does not need to find neither a nor b to compute the vector v^{ab}. □

3.3 A linear group acting by conjugation

Let G be a finitely generated group that comes equipped with an injective homomorphism $\phi : G \to \mathrm{GL}_n(\mathbf{A})$, where $\mathbf{A}$ is a finite dimensional associative algebra over a field $\mathbb{F}$.

Computational assumption on A. *As usual we assume that elements of $\mathbf{A}$ are given in some constructive form and the "size" of the form is defined. Furthermore, we assume that the basic algebra operations in $\mathbf{A}$ are efficient, so they can be performed in polynomial time in the size of the elements. In particular, the matrix multiplication in $\mathrm{Mat}_n(\mathbf{A})$ can be performed in polynomial time. Of course, all the conditions above obviously hold in the case when $\mathbf{A}$ is just the field $\mathbb{F}$.*

Since the group G is finitely generated and multiplication in $\mathrm{GL}_n(\mathbf{A})$ is efficient, one can compute in polynomial time the image g^ϕ for any element $g \in G$, given as a word in a fixed finite set of generators of G.

Notice that $V = \mathrm{Mat}_n(\mathbf{A})$ can be viewed as a finite dimensional vector space over $\mathbb{F}$, where matrices from $\mathrm{Mat}_n(\mathbf{A})$ are tuples of length n^2 over $\mathbf{A}$, i.e., elements from $\mathbf{A}^{n^2}$. If $\mathbf{A}$ has dimension r over $\mathbb{F}$ then $\mathbf{A}^{n^2}$ can be viewed as a vector space over $\mathbb{F}$ of dimension rn^2 in which addition naturally comes from the matrix addition in $\mathrm{Mat}_n(\mathbf{A})$. The group $\mathrm{GL}_n(\mathbf{A})$ acts on V by left as well as right multiplication. In both cases, the homomorphism ϕ gives a faithful representation $\phi : G \to \mathrm{End}(V)$. It follows that any two given elements $g, h \in G$ determine an endomorphism $E_{g,h} : \mathrm{Mat}_n(\mathbf{A}) \to \mathrm{Mat}_n(\mathbf{A})$ defined by $E_{g,h}(x) = \phi(g)x\phi(h)$. In particular, the conjugation by $g \in G$ in $\mathrm{Mat}_n(\mathbf{A})$ corresponds to the endomorphism $E_{g,g^{-1}}$.

Let U and W be two finite subsets of G satisfying the *commutativity assumption* as above, i.e., every element of U commutes with every element of W. Let A and B be the submonoids of G generated by U and W, respectively. Suppose that $a \in A$, $b \in B$, and $v \in G$. Put $v^a = ava^{-1}$, $v^b = bvb^{-1}$.

We also assume that the algebra $\mathbf{A}$, the group G, the embedding $\phi : G \to \mathrm{GL}_n(\mathbf{A})$, the sets U and W, as well as the elements v, v^a, v^b are public. As above the size of the public data is the total size of the following parameters: r and n (given in unary), the sizes of U and W, and the sizes of all public elements. Notice that, in this case, we may assume that elements of G are given as words in a fixed finite generating set. Due to our assumptions on $\mathbf{A}$ and G the embedding ϕ is computable in polynomial time in the length of the words representing elements in G, therefore the sizes of elements of G computed as lengths of the words or the norms of the corresponding matrices are within the polynomial bounds of each other. This implies that the time complexity estimates for our algorithms will be similar if we use representations of the elements as words or as the corresponding matrices.

Claim 2. *Given U, W, g, g^a, g^b, one can find in polynomial time (in the size of the public data) the element $g^{ab} = g^{ba}$.*

Proof. Indeed, the argument above shows that the embedding $\phi : G \to \mathrm{Mat}_n(\mathbf{A})$ gives, in fact, an embedding $\phi : G \to \mathrm{End}(V)$ in such a way that conjugation by an element $g \in G$ gives rise to an endomorphism $E_{g,g^{-1}} \in \mathrm{End}(V)$. Since the embedding ϕ is polynomial time computable, one finds himself in the situation of the basic linear decomposition attack. Now Claim 2 follows immediately from Claim 1. □

There are several possible variations or generalizations of the basic scheme described in this section, which also could be easily reduced to the basic model. We mention some of them below.

3.4 A linear group acting by right/left multiplication

We assume all the notation from Section 3.3. Beyond that assume also that $a, a' \in A$ and $b, b' \in B$.

Claim 3. *Given $U, W, g, agb, a'gb'$, one can find in polynomial time (in the size of the public data) the element $a'agbb' = aa'gb'b$.*

Proof. Indeed, an argument similar to the one in Section 3.3 reduces Claim 3 to the basic model. □

Obviously, Claim 2 is just a particular case of Claim 3. Observe also that Claim 3 holds if one replaces a group G by a semigroup G. The same argument works in this case as well.

3.5 Groups acting by automorphisms

Let G be a finitely generated group and $\mathrm{Aut}(G)$ the group of automorphisms of G. Let U and W be two finite subsets of $\mathrm{Aut}(G)$ satisfying the commutativity assumption as above: every element u of U commutes with every element w of W in $\mathrm{Aut}(G)$, i.e., for any $g \in G$ the equality $g^{uw} = g^{wu}$ holds.

Denote by A and B the submonoids (or subgroups) of $\mathrm{Aut}(G)$ generated by U and W, respectively. Now one can consider an analog of the situation described in Section 3.3, where the conjugations are replaced by arbitrary automorphisms. Namely, suppose some automorphisms $a \in A$, $b \in B$ are chosen. The question arises weather there is a polynomial time algorithm which when given $U \subseteq \mathrm{Aut}(G)$, $W \subseteq \mathrm{Aut}(G)$, $g \in G$, and the images g^a and g^b for some elements $a \in A$, $b \in B$ computes the element $g^{ab} = g^{ba}$ in G. Even if the group G is linear (and the embedding $\phi : G \to \mathrm{GL}_n(\mathbf{A})$ is given), there is no obvious reduction to Claims 1 or 2. Indeed, in this case, arbitrary automorphisms from $\mathrm{Aut}(G)$ do not in general induce endomorphisms on the linear space V (in the notation above). However, the reduction would be possible if one can interprete the automorphisms as conjugations in some (perhaps larger) linear group. Now we discuss one group-theoretic construction that can be useful here.

Recall that the *holomorph* $H(G)$ of a group G is a semidirect product $H(G) = G \rtimes \mathrm{Aut}(G)$ of G and $\mathrm{Aut}(G)$, where the multiplication on pairs from $G \times \mathrm{Aut}(G)$ is defined by $(g, a)(h, b) = (gh^a, ab)$. By construction the groups G and $\mathrm{Aut}(G)$ embed into $H(G)$ via injections $g \to (g, 1)$ and $a \to (1, a)$. Notice that every automorphism $a \in \mathrm{Aut}(G)$ acts on G by a conjugation in $H(G)$, since $(1, a)(h, 1)(1, a^{-1}) = (h^a, 1)$. It follows that if the holomorph $H(G)$ is a linear group, in particular if there is an injective homomorphism $\phi : H(G) \to \mathrm{GL}_n(\mathbf{A})$ for some n and $\mathbf{A}$ as above, then this case can be reduced to Claim 2. In particular, the following result holds.

Claim 4. *Suppose that $H(G)$ is a linear group. Then in the notation above, given U, W, g, g^a, g^b, one can find in polynomial time (in the size of the public data) the element $(g^a)^b = (g^b)^a$.*

Observe that the holomorph $H(G)$ is linear when the group G is finite or polycyclic-by-finite [31].

Suppose a platform group $G = \mathrm{gp}(g_1, \dots, g_r)$ is given by its generators and defining relations. To apply the linear decomposition we need an effective embedding μ of G into a linear group $\mathrm{GL}_n(\mathbb{F})$. Let $\mu(g)$ be a secret date that we get applying our approach to $\mu(G)$. We have to recover g as the result in the original language. Suppose there is an effective procedure of rewriting $\mu(g)$ as a word $w(\mu(g_1), \dots, \mu(g_r))$ in the images $\mu(g_i)$, $i = 1, \dots, r$, of generators. Then $g = w(g_1, \dots, g_r)$ and we succeed. There are several papers giving such algorithms; see [6] and references there. Let us to quote from [6]: "A constructive membership test not only answers the question whether or not a given element belongs to a given group but in the case of positive answer, it also provides a straight-line program that constructs the given element from the given generators of the group."

Recall that constructive membership is the problem of expressing an element in terms of the generators of the group.

Let G be a group and $S \subseteq G$. A straight-line program reaching some element $g \in G$ from S is a sequence $(w_1, \dots, w_m)$, $w_i \in G$, such that for each i either $w_i \in S$ or $w_i = w_j^{-1}$ for some $j < i$ or $w_i = w_j w_k$ for some $j, k < i$.

Let $G \leq H$ be groups; let G be given by a generating set S. The constructive membership problem for G in H is, given $g \in H$, to decide whether $g \in G$, and if so find a straight-line program over S reaching g.

There is a randomized polynomial-time algorithm which uses number theory oracles and, given a matrix group G of odd characteristic p, solves constructive membership in G.

Previously similar results were given by E. M. Luks [29] for solvable matrix groups only. Luk's algorithms are deterministic. Other algorithms and their analysis build on a large body of prior work and most notably on the papers [7, 19, 36].

4 Cryptanalysis of protocols

4.1 Protocols based on conjugation

(1) Ko, Lee et al. key establishment protocol [26]. Let G be a group and U, W be two finite subsets of G which are commuting element-wise. Denote by A and B the subgroups of G generated by U and W, respectively. Fix an element $g \in G$. We assume that all the data above is public.

Algorithm. Alice picks a private element $a \in A$ and publishes g^a. Bob picks a private element $b \in B$ and publishes g^b.

Key establishment. Alice computes $K_A = (g^b)^a = g^{ab}$. Bob computes $K_b = (g^a)^b = g^{ba} = g^{ab}$. The shared key is $K = K_A = K_B = g^{ab}$.

Cryptanalysis. If the group G is linear then by Claim 2 there exists an algorithm that, given the public data above, finds the shared key K in polynomial time.

In the original version of this cryptosystem [26], G was proposed to be the Artin braid group B_n on n strings. In 1990, R. Lawrence described a family of so-called Lawrence representations of B_n. Around 2001, S. Bigelow [8] and D. Krammer [27] independently proved that all braid groups B_n are linear. Their work used the Lawrence–Krammer representations $\rho_n : B_n \to \mathrm{GL}_{n(n-1)/2}(\mathbb{Z}[t^{\pm 1}, s^{\pm 1}])$ that has been proved faithful for every $n \in \mathbb{N}$. One can effectively find the image $\rho_n(g)$ for every element $g \in B_n$. Moreover, there exists an effective procedure to recover a braid $g \in B_n$ from its image $\rho_n(g)$. It was shown by J. H. Cheon and B. Jun in [10] that it can be done in $O(2m^3 \log d_t)$ multiplications of entries in $\rho_n(g)$. Here $m = n(n-1)/2$ and d_t is a parameter that can be effectively computed by $\rho_n(g)$; see [10] for details. Therefore, in this case, there is a polynomial time algorithm to find the shared key K from the public data. The algorithm presented here is more practical. Constructing a basis is off-line. In every session, we have to find on-line, by the Gauss elimination process, coordinates of elements in a given basis of the vector space.

Let Alg(CJ) denote Cheon–Jun's algorithm and Alg(LD) the linear decomposition attack. We can compare these two algorithms.

1. Alg(CJ) is not deterministic because it looks for invertible solutions of underlying sets of linear equations. Alg(LD) is completely deterministic.
2. Alg(CJ) works on-line. Alg(LD) works mostly off-line.
3. Alg(CJ) deals with bigger sets of variables and equations than Alg(LD) does.
4. Alg(CJ) uses the specific embedding μ, but Alg(LD) can work with all effective linear representations. Thus Alg(LD) can be used on different platforms.

(2) Wang et al. key establishment protocol [48]. Let G be a non-commutative monoid. Fix an element $g \in G$. Let x be an invertible element of G. It is assumed that G, g, x are public.

Algorithm. Alice picks a private number $s \in \mathbb{N}$ and publishes g^{x^s}. Bob picks a private number $t \in \mathbb{N}$ and publishes g^{x^t}.

Key establishment. Alice computes $K_A = (g^{x^t})^{x^s} = g^{x^{s+t}}$. Bob computes $K_B = (g^{x^s})^{x^t} = g^{x^{s+t}}$. The shared key is $K = K_A = K_b = g^{x^{s+t}}$.

Cryptanalysis. If the monoid G is linear, then by Claim 2 there exists an algorithm that, given the public data above, finds the shared key K in polynomial time.

However, in [48], Wang et al. used the semigroup G of 3×3 matrices of 1000-truncated polynomials in 10 variables over the ring (not a field) $\mathbb{Z}_{12}$, which does not reduce directly to Claim 2.

Nevertheless, a slight modification of the linear decomposition attack works in this case as well.

4.2 Protocols based on left/right multiplication

(3) B. Hurley and T. Hurley's authentication and digital signature protocols [20, 21]. Let G be a commutative subgroup of $\mathrm{GL}_n(\mathbb{F})$. These dates are public.

Algorithm.

1. Bob picks $y \in \mathbb{F}^n$ and $B \in G$, computes and publishes yB.
2. Alive wants to send a message $x \in \mathbb{F}^n$ to Bob. She picks $A_1, A \in G$, computes and sends (xA, yBA_1) to Bob.
3. Bob picks $B_1, B_2 \in G$, computes and sends (xAB_1, yA_1B_2) to Alice.
4. Alice computes (xB_1, yB_2) and sends $xB_1 - yB_2$ to Bob.
5. Bob computes $x - yB_2B_1^{-1}$ and recovers x.
   Bob may use yB in further transactions.

Cryptanalysis. Since the group G is linear, by Claim 3 there exists an algorithm that, given the public data above, finds the message x in polynomial time. Indeed, let us describe the recovering algorithm.

1. By Claim 3 we build a basis of the space $\mathrm{Sp}(yBA_1)G$. Let this basis be $\{yBA_1C_1, \dots, yBA_1C_r\}$, where $C_i \in G$, $i = 1, \dots, r$.
2. Then we obtain $yB = \sum_{i=1}^r \alpha_i yBA_1C_i$, $\alpha_i \in \mathbb{F}$.
3. Swap yBA_1 by yA_1B_2. We have $\sum_{i=1}^r \alpha_i yA_1B_2C_i = yB_2$.
4. Similarly, we construct a basis $xAB_1D_1, \dots, xAB_1D_t$, $D_j \in G$, $j = 1, \dots, t$, of $\mathrm{Sp}(xAB_1)G$.
5. Compute $xA = \sum_{i=1}^t \beta_i xAB_1D_i$, $\beta_i \in \mathbb{F}$.
6. Swap xAB_1 by $xB_1 - yB_2$. We have $\sum_{i=1}^t \beta_i(xB_1 - yB_2)D_i = x - yB_2B_1^{-1}$.
7. Swap again xAB_1 by yB_2 and get $\sum_{i=1}^t \beta_i yB_2D_i = yB_2B_1^{-1}$.
8. Compute x.

(4) Stickel's key exchange protocol [46]. Let G be a non-abelian finite group and let g and f be two non-commuting elements of G. Let k_0 and l_0 be the orders of g and f, respectively. It is assumed that G, g, f, k_0, l_0 are public.

Algorithm. Alice picks two private positive numbers k and l, $1 < k < k_0$, $1 < l < l_0$, and publishes $g^k f^l$. Bob picks two private positive numbers r and s, $1 < r < k_0$, $1 < s < l_0$, and publishes $g^r f^s$.

Key establishment. Alice computes the element $K_A = g^k(g^r f^s)f^l = g^{k+r}f^{l+s}$. Bob computes the element $K_B = g^r(g^k f^l)f^s = g^{k+r}f^{l+s}$. The shared key is $K = K_A = K_B = g^{k+r}f^{l+s}$.

Cryptanalysis. If $G \leq \mathrm{Mat}_n(\mathbf{A})$ (as is the case in [46]), then by Claim 3 there exists an algorithm that, given the public data above, finds the shared key K in time polynomial in n, $\dim_{\mathbb{F}}(\mathbf{A})$, k_0, l_0 and the sizes of g and f (we assume that the field $\mathbb{F}$ is fixed).

Remark. A similar cryptanalysis applies when G is an arbitrary (not necessary finite) linear group.

(5) Álvarez et al. key exchange protocol [1–3]. Given a prime number p and two positive numbers n, m, Alice chooses M_1 and Bob chooses M_2:

$$M_i = \begin{pmatrix} A_i & X_i \\ 0 & B_i \end{pmatrix}, \quad i = 1, 2.$$

Here, $A_i \in \mathrm{GL}_n(\mathbb{F}_p)$, $B_i \in \mathrm{GL}_m(\mathbb{F}_p)$, $X_i \in \mathrm{M}_{n\times m}(\mathbb{F}_p)$. Let $|M_i| = m_i$ be the order of M_i. For any positive number t one has

$$M_i^t = \begin{pmatrix} A_i^t & X_i^{(t)} \\ 0 & B_i^t \end{pmatrix}, \quad i = 1, 2.$$

Algorithm. Alice picks two private positive numbers k_i, $1 \leq k_i \leq m_i - 1$, $i = 1, 2$, and publishes

$$C = M_1^{k_1} M_2^{k_2} = \begin{pmatrix} A_C & X_C \\ 0 & B_C \end{pmatrix}.$$

Bob picks two private positive numbers l_i, $1 \leq l_i \leq m_i - 1$, $i = 1, 2$, and publishes

$$D = M_1^{l_1} M_2^{l_2} = \begin{pmatrix} A_D & X_D \\ 0 & B_D \end{pmatrix}.$$

Key establishment. Alice computes

$$K_A = A_1^{k_1} A_D X_2^{(k_2)} + A_1^{k_1} X_D B_2^{k_2} + X_1^{(k_1)} B_D B_2^{k_2}.$$

Bob computes

$$K_B = A_1^{l_1} A_C X_2^{(l_2)} + A_1^{l_1} X_C B_2^{l_2} + X_1^{(l_1)} B_C B_2^{l_2}.$$

The shared key is $K = K_A = K_B$. It is noted in [14] that K is the $(1, 2)$ entry of $M_1^{k_1+l_1} M_2^{k_2+l_2}$.

Cryptanalysis. By Claim 3 there exists an algorithm that, given the public data above, finds the shared key K in time polynomial in n, m, m_1, m_2 and the sizes of the matrices M_1 and M_2 (we assume that the field $\mathbb{F}_p$ is fixed).

(6) Shpilrain–Ushakov's key exchange protocol [42]. (See also [32].) Here we describe the general (non-twisted) version of the protocol and show that if the platform group is linear then there is an efficient attack to recover the shared key. There is also a twisted version of the protocol, to which our cryptanalysis applies as well, so we omit it here and we leave the details to the reader. Notice that in the original paper [42], the suggested platform is the Thompson group, which is non-linear, so our attack does not apply here.

Let $G \leq \mathrm{M}_n(\mathbf{A})$ be a group (or a submonoid) and let g be an element of G. Let A and B be two public finitely generated subgroups (or submonoids) of G commuting element-wise.

Algorithm. Alice picks private elements $a, a' \in A$ and publishes the element aga'. Bob picks private elements $b, b' \in B$ and publishes the element bgb'.

Key establishment. Alice computes $K_A = abgb'a'$. Bob computes $K_B = baga'b' = abgb'a'$. The shared key is $K = K_A = K_b = abgb'a'$.

Cryptanalysis. By Claim 3 there exists an algorithm that, given the public data above, finds the shared key K in time polynomial in n, $\dim_{\mathbb{F}}(\mathbf{A})$, and the sizes of the fixed generating sets of A and B and the sizes of the elements g, aga', and bgb' (we assume that the field $\mathbb{F}$ is fixed).

(7) Romanczuk–Ustimenko key exchange protocol [9]. Let $G = \mathrm{GL}_n(\mathbb{F})$, where $\mathbb{F}$ is a finite field. Suppose $C, D \in G$ are two commuting matrices. Fix a vector $g \in \mathbb{F}^n$. All this data is public.

Algorithm. Alice picks a polynomial $P = P(C, D) \in \mathbb{F}[x, y]$ and publishes the vector gP. Bob picks a polynomial $Q = Q(C, D) \in \mathbb{F}[x, y]$ and publishes the vector gQ.

Key establishment. Alice computes $K_A = (gQ)P = gQP$ and Bob computes $K_B = (gP)Q = gPQ$. The shared key is the vector $K = K_A = K_B$.

Cryptanalysis. By Claim 1 there exists an algorithm that, given the public data above, finds the shared key K in time polynomial in n and the sizes of C, D, P, and Q (we assume that the field $\mathbb{F}$ is fixed).

Remark. Another attack on this protocol, also based on linear algebra, was proposed earlier by Blackburn, Cid and Mullan [9]. The main idea of the attack is as follows.

Suppose an adversary Eve knows g, gP, gQ, C, and D. Let X be any matrix such that X commutes with C and with D, and such that $gQ = gX$. To find such an X it suffices to solve the corresponding system of linear equations. Now Eve can compute the shared key as $(gP)X = gXP = gQP = K$.

4.3 Protocols using automorphisms of groups

(8) Mahalanobis' key exchange protocol 1 [30]. Let G be a group and $g \in G$. Suppose U, W are two finite subsets of $\mathrm{Aut}(G)$ commuting element-wise. Denote by Φ and Ψ the subgroups in $\mathrm{Aut}(G)$ generated by U and W, respectively.

Algorithm. Alice picks $\phi \in \Phi$ and publishes $\phi(g)$. Bob picks $\psi \in \Psi$ and publishes $\psi(g)$.

Key establishment. Alice computes $K_A = \phi(\psi(g))$. Bob computes $K_B = \psi(\phi(g)) = \phi(\psi(g))$. The shared key is $K = K_A = K_B = \phi(\psi(g))$.

Cryptanalysis. If G is such that $\mathrm{Hol}(G)$ is linear (a subgroup of $\mathrm{Mat}_n(\mathbf{A})$) then by Claim 4 there exists an algorithm that, given the public data above, finds the shared key K in time polynomial in n, $\dim_{\mathbb{F}}(\mathbf{A})$ and the sizes of g and the elements in U, W (we assume that the field $\mathbb{F}$ is fixed).

In the original paper [30], Mahalanobis suggested a (finitely generated) non-abelian nilpotent group G as the platform group. It is known (see, for example, [28, 31, 40]) that the holomorph $\mathrm{Hol}(G)$ of every polycyclic group, in particular, every finitely generated nilpotent group, admits a faithful matrix representation. Hence the analysis above holds.

Observe that the efficacy of the attack depends on the size of the linear representation of $\mathrm{Hol}(G)$. Not much is known about the dimensions of minimal faithful representations of the holomorphs of nilpotent groups.

(9) Mahalanobis' key exchange protocol 2 [30]. We assume the notation above.

Algorithm. Alice picks $\phi \in \Phi$ and sends g^ϕ to Bob. Bob picks $\psi \in \Psi$ and sends $(g^\phi)^\psi = g^{\phi\psi}$ back to Alice. Alice computes ϕ^{-1} and gets $g^\psi = g^{\phi\psi\phi^{-1}}$. Then Alice picks another automorphism $\xi \in \Phi$ and sends $(g^\psi)^\xi = g^{\psi\xi}$ to Bob.

Key establishment. Bob computes ψ^{-1} and gets $((g^\psi)^\xi)^{\psi^{-1}} = g^\xi$ which is his session key.

Cryptanalysis. Similar to the case above.

Indeed, assume $\mathrm{Hol}(G) \leq \mathrm{Mat}_n(\mathbf{A})$. Put $v = g^{\phi\psi}$. As in Claim 1 one can find a basis of $\mathrm{Sp}(v^\Psi)$, say $v^{c_1}, \dots, v^{c_t}$. Notice that $g^{\psi\xi} \in \mathrm{Sp}(v^\Psi)$, so one can decompose

$$g^{\psi\xi} = \sum_{i=1}^{t} \alpha_i v^{c_i} = \Big(\sum_{i=1}^{t} \alpha_i (g^\phi)^{c_i} \Big)^\psi \quad \text{for } \alpha_i \in \mathbb{F}.$$

Hence, $(g^{\xi})^{\psi} = (\sum_{i=1}^{t} \alpha_i (g^{\phi})^{c_i})^{\psi}$, so we derive

$$g^{\xi} = \sum_{i=1}^{t} \alpha_i (g^{\phi})^{c_i}.$$

(10) Habeeb, Kahrobaei, Koupparis and Shpilrain's key exchange protocol [18]. Let G be a (semi)group, and $\mathrm{Aut}(G)$ be the automorphism group of G. Let $H(G)$ be the holomorph of G. Fix an element $g \in G$ and an automorphism $\phi \in \mathrm{Aut}(G)$. All this data is public.

In this paragraph, for $g \in G$ and $\mu \in \mathrm{Aut}(G)$ we write the image as $\mu(g)$ instead of g^{μ}.

Algorithm. Alice picks a private number $m \in \mathbb{N}$. Then she computes

$$(\phi, g)^m = (\phi^m, \phi^{m-1}(g) \cdot \ldots \cdot \phi^2(g) \cdot \phi(g) \cdot g)$$

and sends only the second component $a_m = \phi^{m-1}(g) \cdot \ldots \cdot \phi^2(g) \cdot \phi(g) \cdot g$ of this pair to Bob.

Bob picks a private $n \in \mathbb{N}$. Then he computes

$$(\phi, g)^n = (\phi^n, \phi^{n-1}(g) \cdot \ldots \cdot \phi^2(g) \cdot \phi(g) \cdot g)$$

and sends only the second component $a_n = \phi^{n-1}(g) \cdot \ldots \cdot \phi^2(g) \cdot \phi(g) \cdot g$ of this pair to Alice.

Key establishment. Alice computes $(*, a_n) \cdot (\phi^m, a_m) = (* \cdot \phi^m, \phi^m(a_n) \cdot a_m) = (* \cdot \phi^m, K_A)$. Note that she does not actually "compute" $* \cdot \phi^m$.

Bob computes $(**, a_m) \cdot (\phi^n, a_n) = (** \cdot \phi^n, \phi^n(a_m) \cdot a_n) = (** \cdot \phi^n, K_B)$. Note that he does not actually "compute" $** \cdot \phi^n$.

The shared key is $K = K_A = K_B = a_{m+n}$.

Cryptanalysis. Let $G \leq \mathbf{A}$, where $\mathbf{A}$ is a finite dimensional associative algebra over a field $\mathbb{F}$. Assume that the automorphism ϕ is extended to an automorphism of the underlying vector space of $\mathbf{A}$.

Using Gauss elimination, we can effectively find a maximal linearly independent subset L of the set $\{a_0, a_1, \ldots, a_k, \ldots\}$, where $a_0 = g$ and $a_k = \phi^{k-1}(g) \cdot \ldots \cdot \phi(g) \cdot g$ for $k \geq 1$. Indeed, suppose that the set $\{a_0, \ldots, a_k\}$ is linearly independent, but a_{k+1} can be presented as a linear combination of the form

$$a_{k+1} = \sum_{i=0}^{k} \lambda_i a_i \quad \text{for } \lambda_i \in \mathbb{F}.$$

Suppose by induction that a_{k+j} can be presented as above for every $j \leq t-1$. In particular,

$$a_{k+t-1} = \sum_{i=0}^{k} \mu_i a_i \quad \text{for some } \mu_i \in \mathbb{F}.$$

Then

$$a_{k+t} = \phi(a_{k+t-1}) \cdot g = \sum_{i=0}^{k} \mu_i \phi(a_i) \cdot g = \sum_{i=0}^{k} \mu_i a_{i+1} = \mu_k \lambda_0 a_0 + \sum_{i=0}^{k-1} (\mu_i + \mu_k \lambda_{i+1}) a_{i+1}.$$

Thus $L = \{a_0, \ldots, a_k\}$. In particular, we can effectively compute

$$a_n = \sum_{i=0}^{k} \eta_i a_i \quad \text{for some } \eta_i \in \mathbb{F}.$$

Then

$$a_{m+n} = \phi^m(a_n) \cdot a_m = \sum_{i=0}^{k} \eta_i \phi^m(a_i) \cdot a_m = \sum_{i=0}^{k} \eta_i \phi^i(a_m) \cdot a_i.$$

Note that all data on the right-hand side is known now. Thus we get the shared key $K = a_{m+n}$.

In the original version of this cryptosystem [18], G was proposed to be the semigroup of 3×3 matrices over the group algebra $\mathbb{F}_7[\mathbb{A}_5]$, where $\mathbb{A}_5$ is the alternating group on 5 elements. The authors of [18] used an extension of the semigroup G by an inner automorphism which is conjugation by a matrix $H \in \mathrm{GL}_3(\mathbb{F}_7[\mathbb{A}_5])$. Therefore, in this case, there is a polynomial time algorithm to find the shared key K from the public data.

Funding: The first author was partially supported by NSF grant DMS-1318716. The second author was supported by RFBR, project 13-01-00239a. Both authors were supported by Russian Research Fund, project 14-11-00085.

References

[1] R. Álvarez, F.-M. Martinez, J. F. Vicent and A. Zamora, A matricial public key cryptosystem with digital signature, *WSEAS Trans. Math.* **4**, (2008), 195–204.
[2] R. Álvarez, L. Tortosa, J. Vicent and A. Zamora, A non-abelian group based on block upper triangular matrices with cryptographic applications, in: *Proceedings of the 18th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes* (AAECC-18 '09), Springer, Berlin (2009), 117–126.
[3] R. Álvarez, L. Tortosa, J. Vicent and A. Zamora, Analysis and design of a secure key exchange scheme, *Inform. Sci.* **179** (2009), 2014–2021.
[4] I. Anshel, M. Anshel and D. Goldfeld, An algebraic method for public-key cryptography, *Math. Res. Lett.* **6** (1999), 287–291.
[5] I. Anshel, M. Anshel, D. Goldfeld and S. Lemieux, Key agreement, the algebraic eraser, and lightweight cryptography, in: *Algebraic Methods in Cryptography*, Contemp. Math. 418, American Mathematical Society, Providence (2006), 1–34.
[6] L. Babai, R. Beals and A. Seress, Polynomial-time theory of matrix groups, in: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing* (STOC'09), Association for Computing Machinery, New York (2009), 55–64.
[7] L. Babai, P. P. Pálfy and J. Saxl, On the number of p-regular elements in simple groups, *LMS J. Comput. Math.* **12** (2009), 82–119.
[8] S. Bigelow, Braid groups are linear, *J. Amer. Math. Soc.* **14** (2001), 471–486.
[9] S. R. Blackburn, C. Cid and C. Mullan, Cryptanalysis of three matrix-based key establishment protocols, *J. Math. Cryptol.* **5** (2011), 159–168.
[10] J. H. Cheon and B. Jun, A polynomial time algorithm for the braid Diffie–Hellman conjugacy problem, in: *Advances in Cryptology* (CRYPTO 2003), Lecture Notes in Comput. Sci. 2729, Springer, Berlin (2003), 212–225.
[11] D. Coppersmith, A. Odlyzko and R. Schroeppel, Discrete logarithms in GF(p), *Algorithmica* **1** (1986), 1–15.
[12] W. Diffie and M. E. Hellman, New directions in cryptography, *IEEE Trans. Inform. Theory* **22** (1976), 644–654.
[13] S. Y. Erofeev and V. A. Roman'kov, On constructing possibly one-way functions based on the non-decidability of the endomorphism problem in groups (in Russian), *Prikladnaya Discretnaya Matematika* **3** (2012), 13–24.
[14] M. I. González Vasco, A. L. Pérez del Poso and P. T. Duarte, Cryptanalysis of a key exchange scheme based on block matrices, preprint (2009), http://eprint.iacr.org/2009/553.
[15] W. A. De Graaf and W. Nickel, Constructing faithful representations of finitely-generated torsion-free nilpotent groups, *J. Symbolic Comput.* **33** (2002), 31–41.
[16] D. Grigoriev and V. Shpilrain, Authentication from matrix conjugation, *Groups Complex. Cryptol.* **1** (2009), 199–206.
[17] M. Habeeb and D. Kahrobaei, On the dimension of matrix representations of finitely generated torsion-free nilpotent groups, preprint (2013), http://arxiv.org/abs/1309.4514.
[18] M. Habeeb, D. Kahrobaei, C. Koupparis and V. Shpilrain, Public key exchange using semidirect product of (semi)groups, preprint (2013), http://arxiv.org/abs/1304.6572.
[19] P. E. Holmes, S. A. Linton, E. A. O'Brien, A. J. E. A. Ryba and R. A. Wilson, Constructive membership in black-box groups, *J. Group Theory* **11** (2008), 747–763.
[20] B. Hurley and T. Hurley, Group ring cryptography, preprint (2011), http://arxiv.org/abs/1104.1724.
[21] T. Hurley, Cryptographic schemes, key exchange, public key, preprint (2013), http://arxiv.org/abs/1305.4063.
[22] G. J. Janusz, Faithful representations of p groups at characteristic p, I, *J. Algebra* **15** (1970), 335–351.
[23] D. Kahrobaei and B. Khan, A non-commutative generalization of ElGamal key exchange using polycyclic groups, in: *Global Telecommunication Conference* (GLOBECOM'06), IEEE Computer Society (2006), 1–5.
[24] D. Kahrobaei, C. Koupparis and V. Schpilrain, Public key exchange using matrices over group rings, *Groups Complex. Cryptol.* **5** (2013), 97–115.
[25] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*, Chapman & Hall/CRC, 2007.
[26] K. H. Ko, S. J. Lee, J. H. Cheon, J. W. Han, J. Kang and C. Park, New public-key cryptosystem using braid groups, in: *Advances in Cryptology* (CRYPTO 2000), Lecture Notes in Comput. Sci. 1880, Springer, Berlin (2000), 166–183.
[27] D. Krammer, Braid groups are Linear, *Ann. Math.* **155** (2002), 131–156.
[28] J. C. Lennox and D. J. S. Robinson, *The Theory of Infinite Soluble Groups*, Oxford Math. Monogr., Oxford University Press, Oxford, 2004.
[29] E. M. Luks, Computing in solvable matrix groups, in: *Proc. 33rd FOCS*, IEEE Computer Society (1992), 111–120.
[30] A. Mahalanobis, The Diffie–Hellman key exchange protocol and non-abelian nilpotent groups, *Israel J. Math.* **165** (2008), 161–187.

[31] Y. I. Merzlyakov, Integral representation of holomorphs of polycyclic groups, *Algebra Logic* **9** (1970), 326–337.

[32] A. G. Miasnikov, V. Shpilrain and A. Ushakov, *Group-Based Cryptography*, Advanced Courses in Math., CRM Barcelona, Birkhäuser, Basel, 2008.

[33] A. G. Miasnikov, V. Shpilrain and A. Ushakov, *Non-commutative Cryptography and Complexity of Group Theoretic Problems*, Math. Surveys Monogr., American Mathematical Society, Providence, 2011.

[34] W. Nickel, Matrix representations for torsion-free nilpotent groups by deep thought, *J. Algebra* **300** (2006), 376–383.

[35] R. Odoni, V. Varadharajan and P. Sanders, Public key distribution on matrix rings, *Electronic Lett.* **20** (1984), 386–387.

[36] C. W. Parker and R. A. Wilson, Recognising simplicity of black-box groups, *J. Algebra* **324** (2010), 885–915.

[37] V. A. Roman'kov, *Algebraic Cryptography* (in Russian), Omsk State Dostoevsky University, 2013.

[38] V. A. Roman'kov, Cryptanalysis of some schemes applying automorphisms (in Russian), *Prikladnaya Discretnaya Matematika* **3** (2013), 35–51.

[39] L. Sakalauskas, P. Tvarijonas and A. Raulynaitis, Key agreement protocol (kap) using conjugacy and discrete logarithm problems in group representation level, *Informatica* **18** (2007), 115–124.

[40] D. Segal, *Polycyclic Groups*, Cambridge Tracts in Math. 82, Cambridge University Press, Cambridge, 1983.

[41] V. Shpilrain, Cryptanalysis of Stickel's key exchange scheme, in: *Computer Science in Russia 2008*, Lecture Notes in Comput. Sci. 4296, Springer, Berlin (2008), 283–288.

[42] V. Shpilrain and A. Ushakov, Thompson's group and public key cryptography, in: *Applied Cryptography and Network Security* (ACNS 2005), Lecture Notes in Comput. Sci. 3531, Springer, Berlin (2005), 151–164.

[43] V. Shpilrain and A. Ushakov, A new key exchange protocol based on the decomposition problem, in: *Algebraic Methods in Cryptography*, Contemp. Math. 418, American Mathematical Society, Providence (2006), 161–167.

[44] V. Shpilrain and A. Ushakov, The conjugacy search problem in public key cryptography: Unnecessary and unsufficient, *Appl. Algebra Engrg. Comm. Comput.* **17** (2006), 285–289.

[45] V. M. Sidelnikov, M. A. Cherepnev and V. Y. Yashenko, Systems of open distribution of keys on the basis of noncommutative semigroups, *Russian Acad. Sci. Dokl. Math.* **48-2** (1994), 384–386.

[46] E. Stickel, A new method for exchanging secret keys, in: *Proceeding of the Third International Conference on Information Technology and Applications* (ICITA 05), Contemp. Math. 2, IEEE Computer Society (2005), 426–430.

[47] B. Tsaban, Polynomial time solutions of computational problems in noncommutative-algebraic cryptography, preprint (2012), http://arxiv.org/abs/1210.8114.

[48] L. Wang, L. Wang, Z. Cao, E. Okamoto and J. Shao, New constructions of public-key encryption schemes from conjugacy search problems, in: *Information Security and Cryptology*, Lecture Notes in Comput. Sci. 6584, Springer, Berlin (2010), 1–17.