6-2-2017

Rewriting Methods in Groups with Applications to Cryptography

Gabriel Zapata
The Graduate Center, City University of New York

REWRITING METHODS IN GROUPS WITH APPLICATIONS TO CRYPTOGRAPHY

By

Gabriel Zapata

A dissertation submitted to the Graduate Faculty in Mathematics in partial fulfillment for the requirements for the degree of Doctor of Philosophy,

The City University of New York

2017

THE CITY UNIVERSITY OF NEW YORK
DEPARTMENT OF
MATHEMATICS

This manuscript has been read and accepted for the Graduate Faculty in Mathematics in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

Vladimir Shpilrain

Date | Chair of Examining Committee

Ara Basmajian

Date | Executive Officer

Supervision committee:

Melvyn Nathanson

Alexey Ovchinnikov

Abstract

Rewriting Methods in Groups with Applications to Cryptography

by
Gabriel Zapata

Adviser: Professor Vladimir Shpilrain

In this thesis we describe how various rewriting methods in combinatorial group theory can be used to diffuse information about group elements, which makes it possible to use these techniques as an important constituent in cryptographic primitives. We also show that, while most group-based cryptographic primitives employ the complexity of search versions of algorithmic problems in group theory, it is also possible to use the complexity of decision problems, in particular the word problem, to claim security of relevant protocols.

Acknowledgements

I would like to thank Professor Vladimir Shpilrain, my advisor, for his many suggestions and invaluable support over the years. His unshakable enthusiasm and unwavering belief in me, gave me the impetus to persevere; his unique approach to mathematics has influenced me in the art of science and everyday life.

My cherished mother, Yolanda, deserves far more than mere thanks. For inspiring me not to give up: it took me a PhD to realize and appreciate what she has done.

I also would like to express my gratitude to the other committee members, professor Melvyn Nathanson and professor Alexey Ovchinnikov, for taking the time to read my thesis and serve on my defense committee.

Needless to say, my sincere thanks are sent to my friends: Alex D. Myasnikov, Elena Myasnikova, Alex Guzman, Garry Call, Jorgue Florez, Joseph Driscoll, Alberto Mantilla, Jayson Rome, Mila Radulavich, Tara Wingate, and Sebastian Tola.

Finally, I gratefully acknowledge the kindnesses of all my colleagues who have helped and inspired me with their work ethic, and lending time; in particular, Alexei G. Myasnikov, Michael Anshel, Olga Kharlampovich, Marco Ziman, Sasha Ushakov, Delaram Kahrobaei, Robert Gilman, Ben Fine, and Gilbert Baumslag.

New York Gabriel Zapata

Table of Contents

Introduction

The objective of this thesis is to explore how developments in non-commutative group theory can be applied in *public-key cryptography*, or asymmetric cryptography. In particular, we suggest three cryptographic primitives that employ rewriting methods and problems from combinatorial group theory.

Asymmetric cryptography differs from symmetric (classical) cryptography, which generally uses a single key that allows for both encryption and decryption of messages. Symmetric methods for secure transmission of information were the only publicly-known strategies used before the year 1976. Then after, W. Diffie and M. Hellman introduced in their seminal paper [13] an ingenious theoretical method of openly transmitting information securely, and started the development of what is now known as public-key cryptography, i.e., asymmetric cryptography. Their specific procedure was later given the name of the Diffie-Hellman protocol.

Behind the alluring appeal of many public-key protocols is their theoretical simplicity along with the alluding use of an inherent *one-way function* to encrypt messages: a function f easy to compute in its domain but its inverses $f^{-1}(y)$ are hard to compute for "most" y in its range. There is more to this, of course; for a cryptographic scheme to be secure, it is crucial that its keys are randomly selected from a "small" subset from the domain of the one-way function instead of the whole domain.

Detecting these hard to compute subsets, or computational black holes, is a very difficult issue and it is the core design topic in applications of a cryptographic primitive. Without these computational black holes any implementation of a cryptographic primitive would be impossible.

Finding these computational black holes for a one-way function is an important goal. For example, the time-consuming difficulty in factoring "some" large integers elicits the RSA protocol—due to Rivest, Shamir and Adleman. RSA is the most common public-key cryptosystem in use today and the most acclaimed implementation of one-way functions known to date. Therefore, every aspect of a one-way functions is central in the design of exchanging data securely.

The security of protocols like Diffie-Hellman and RSA did not appear to be compromised (albeit known weaknesses exist), however, a recent report from the NIST [27] claims that in the very near future quantum computer will bring about the end of canonical implementations of asymmetric cryptography. Besides the superposition and entanglement properties from quantum mechanics that allow for fast computing with a quantum computer, in essence, standard quantum computers work by exploiting the internal properties of the *gauge group*, or symmetry group, from quantum electrodynamics: the group of complex number of unit length $\mathbf{U}(1)$, which is abelian. By operating with $\mathbf{U}(1)$, a quantum computer is like a quick linear-pattern recognizer, where, metaphorically, left and right are indifferent to it. Hitherto, the general efficiency of all asymmetric cryptosystems is decided by one-way functions emerging from the computational complexity intrinsic to finite abelian (or commutative) groups, which quantum computers can universally handle efficiently.

Quantum computers with approximately 512 qubits are now claimed to be in use

by companies like Google, Lockheed Martin and government agencies, see e.g. [47]. For these reasons there is a strong interest in designing *Post-Quantum Key Exchange* protocols that could resist quantum computer algorithms. In search for these new cryptographic primitives, post-quantum cryptography has given rise to alternatives such as finite field, elliptic curve, and lattice based cryptography. However, because the nature of a quantum computer is deeply rooted in the structure of its gauge group, it seems reasonable to explore possible enhancements of protocols emanating from a non-abelian group structure.

In theory a quantum computer that could potentially handle the structure of some non-abelian groups efficiently would have to use a non-abelian gauge group. However, for example, a computer using the non-abelian gauge group from quantum cromodynamics would have to be made from stable quark-gluon plasma, which theoretically can only be found inside neutron stars. For these reasons we believe non-commutative groups can deliver the ambition of describing one of these sought-out, non-commuting asymmetric cryptographic protocols.

A path paving-way towards this desire focuses on the search for group platforms where the security of the protocol would have different premises from the canonical models. Take for example Wagner and Magyarik [48], who in 1985 thought of a public-key protocol based on the unsolvability of the word problem for finitely presented groups. Although the protocol seems somewhat naive now, it was pioneering. In particular, it shows that the idea of using the complexity of non-abelian groups in cryptography is not recent; however, the new interest in applications of non-abelian group theory to cryptography can be traced to [1, 26, 44].

Group-based cryptography is still in its early stages of development, although it

has been productively advancing over last decade (see [49] and [35]). The majority of group-theoretic protocols are based on search problems, which came from traditional *decision problems* in combinatorial group theory. In our applications, we employ both of these problems. Given a property $\mathcal{P}$ and an object $\mathcal{O}$, the decision problem $\mathcal{P}$ for $\mathcal{O}$ is the problem of deciding whether or not $\mathcal{O}$ has the property $\mathcal{P}$; while the search problem $\mathcal{P}$ is the problem of finding at least one particular object $\mathcal{O}$ with the property $\mathcal{P}$ from a pool $\mathcal{S}$ of objects whenever there is information that objects with the property $\mathcal{P}$ do exist. For example, the conjugacy search problem [1, 26]; the homomorphism search problem [20], [44]; the decomposition search problem [10, 26, 43] and the subgroup membership search problem [45] are some of the proposed search based primitives that have been suggested.

Employing a non-abelian group primitive based on a search problem in a non-abelian group introduces the very challenging problem of determining computational black holes for the chosen primitive—similar to the canonical model—as any protocol based on non-abelian groups has to eventually address this issue for practical implementations. On the other hand, employing decision problems allows for the development of cryptographic primitives with new properties, impossible in the canonical model. We address some of these issues by also considering primitives that only transmit partial information about its elements with the aid of rewriting methods from combinatorial group theory.

In Chapter 1 we use the decision problems known as the *word problem* to describe our cryptographic protocol [46]. This protocol has the following features: (1) Bob transmits to Alice an encrypted binary sequence which Alice decrypts correctly with probability "very close" to 1; (2) the adversary, Eve, who is granted arbitrarily high

(but fixed) computational speed, cannot positively identify (at least, in theory), by using a "brute force attack", the "1" or "0" bits in Bob's binary sequence. In other words: no matter what computational speed we grant Eve at the outset, there is no guarantee that her "brute force attack" program will give a conclusive answer (or an answer which is correct with overwhelming probability) about any bit in Bob's sequence.

So far, no protocol based on a search problem for a non-abelian group has been recognized as secure enough to be a viable alternative to other established protocols. However, by using the word problem with this probabilistic constraint, we believe it allows one to depart from the canonical paradigm. In particular, such protocols can potentially be secure against some "brute force" attacks by a computationally unbounded adversary.

In Chapter 2 we describe a cryptosystem [45] whose security is based on the computational hardness of the *subgroup membership search problem*: given a group G, a subgroup H generated by $h_1, \ldots, h_k$, and an element $h \in H$, find an expression of h in terms of $h_1, \ldots, h_k$. We believe that using the subgroup membership search problem also allows one to depart from most public key protocols based on non-abelian groups, which generally use the computational difficulty of either the conjugacy search problem or the word search problem. In our protocol, by using a subset of the Cartesian product of a subgroup H, we allow for use of partial information from our primitive, and not the whole. Also, in contrast to groups typically used in public-key cryptography, we suggest the class of *free solvable groups* as its platform, which are *infinitely presented* groups and have efficiently solvable word problem.

In Chapter 3 we derive a rewriting method for group elements that uses the

concept of a transversal set. A set of transversals T allows us to decompose an element $g \in G$ into two parts, say, $g := th$, where $t \in T$ and $h \in H$. By performing this decomposition we show how to rewrite the product of any two elements in G in terms of their respective parts from T and H. Essentially the group G will decompose into a set $T \times H$, which becomes a group isomorphic to G under the product we derive from our rewriting method.

Finally, we use the rewriting method developed in Chapter 3 to design a key exchange protocol in Chapter 4 that is analogous to the Kahrobaei and Shpilrain scheme [12]. More specifically, instead of applying the semidirect product as Kahrobaei and Shpilrain did, we employ the set $T \times H$ for a transversal T of G/H to design a key exchange primitive resembling the Diffie-Hellman protocol. Despite the difference between our primitives, both agree on a valuable innovative feature: One does not have to transmit the entire data, but only a part of it. We also emphasize that our protocol is less implementation-oriented. Instead, we focus in describing this new idea and research directions the primitive can open.

The development of the rewriting method of Chapter 3 for implementation in Chapter 4 illustrates an important feedback that exists between cryptography and group theory, which is not unexpected since problems motivated by cryptography are group-theoretical in nature. However, the emphasis of this thesis is to promote innovations in asymmetric cryptography, and the intuition behind it. Although we do *not* address security properties the way they are typically considered in traditional cryptography, we believe the theoretical work presented here can be promising for applications. These procedures or generalizations thereof could potentially be applied with greater efficiency than methods that are currently in use.

Chapter 1

Decision Problems in Public-Key Cryptography

In search for a more efficient and/or secure alternative to established cryptographic protocols (such as RSA), several authors have come up with public-key establishment protocols, as well as with complete public key cryptosystems, based on allegedly hard *search problems* from combinatorial (semi)group theory, including the conjugacy search problem [1, 26], the homomorphism search problem [20], [44], the decomposition search problem [10, 26, 43], the subgroup membership search problem [45]. All these are problems of the following nature: given a property $\mathcal{P}$ and the information that there are objects with the property $\mathcal{P}$, find at least one particular object with the property $\mathcal{P}$ from a pool $\mathcal{S}$ of objects. So far no cryptographic protocol based on a search problem in a non-commutative (semi)group has been recognized as secure enough to be a viable alternative to established protocols (such as RSA) based on commutative (semi)groups.

From the very nature of these problems one sees that the security of the corresponding cryptographic protocols relies heavily on the assumption that the adversary has only limited computational capabilities. Indeed, there is usually a natural way to

recursively enumerate elements of the pool $\mathcal{S}$; hereto, the adversary can just go over $\mathcal{S}$ one element at a time until he hits one with the property $\mathcal{P}$ (assuming that checking the latter can be done efficiently). This particular adversarial attack strategy is commonly know as a "brute force" attack.

In this chapter we use *decision problems* from combinatorial group theory as the core of a public key establishment protocol or a complete public-key cryptosystem. Decision problems are problems of the following nature: given a property $\mathcal{P}$ and an object $\mathcal{O}$, find out whether or not the object $\mathcal{O}$ has the property $\mathcal{P}$. We show that decision problems may be useful when one addresses the ultimate challenge of public-key cryptography: design a cryptosystem secure against brute-force attacks by an adversary with arbitrarily high computational speed.

In particular, we use the *word problem* to design a cryptosystem with the following features: (1) Bob transmits to Alice an encrypted binary sequence which Alice decrypts correctly with probability "very close" to 1; (2) the adversary, Eve, who is granted arbitrarily high computational speed (although limited storage space), can positively identify the "1" bits in those places of the binary sequence where Bob intended to transmit a 1. However, there is no way for Eve (at least, in theory) to positively identify the bits in those places where Bob intended to transmit a 0. In other words: granted an arbitrary speed of computation, Eve's "brute force attack" program will run without giving a conclusive answer in those places where Bob intended to transmit a 0 until it reaches the limit on the storage space size (albeit, we assume the adversary has arbitrary high speed of computation, we do *not* grant the adversary *unbounded* speed of computation.)

In the following sections we address the most relevant and typical questions of

this chapter; in particular, why encryption (by Bob) and decryption (by Alice) are efficient, why the receiver (Alice) and the adversary are separated in power, why the only obvious brute-force attack fail under our computational assumptions, etc.

1.1 Computational Descriptions

We assume an algorithm $\mathcal{A}$ to be a deterministic multi-tape Turing machine, adopting the notion of computational complexity from [37]. A *time-complexity* class $\mathcal{C}$ for $\mathcal{A}$ is a collection of functions bounding the time of the computation. A class $\mathcal{C}$ consists of *proper complexity functions* $f(n) \geq n$, such that for any $f \in \mathcal{C}$ and any integer $c \geq 1$, the function $cf(cn + c) + c$ is also in $\mathcal{C}$.

In the following sections of this chapter we assume that the adversary does not have any *externally imposed* limit on the *speed* of computation. More precisely, our computational model is as follows: we explain to the adversary how our cryptosystem works and allow him to choose any time-complexity class $\mathcal{C}$ for the of computation he wants to use to attack the cryptosystem; however, he cannot change his choice after he has made it—i.e., he cannot accelerate his computation beyond the limit he has chosen for himself.

On the other hand we impose a limit on the adversary's storage space; however, this limit is rather generous, so that he cannot run more than 10^{100} different programs at the same time. To have such a limit on the size of storage space seems to be a reasonable assumption because, conceivably, this size cannot possibly exceed the number of electrons in the observable universe.

Definition 1.1.1. The *word problem* for a recursive presentation of a group G is the

algorithmic problem of deciding whether or not $g = 1$ in G given any $g \in G$.

From the very description of the word problem we see that it consists of two parts: "whether" and "not". We call them the "yes" and "no" parts of the word problem, respectively. If a group is described by a recursive presentation in terms of generators and relators, then the "yes" part of the word problem has a recursive solution because one can recursively enumerate all products of defining relators, their inverses and conjugates. However, the number of factors in such a product required to express a word of length n that equals 1 in G, can be very large compared to n. In particular, there are groups G with efficiently solvable word problem and words w of length n equal to 1 in G, such that the number of factors in any factorization of w into a product of defining relators, their inverses, and its conjugates, is not bounded by any tower of exponents in n (cf. [38]).

Furthermore, if the word problem is recursively unsolvable in a group G, the length of a proof verifying that $w = 1$ is satisfied in G is not bounded by any recursive function on the length of w. Moreover the "no" part of the word problem in many groups is recursively unsolvable, and therefore a "brute force" attack will not be effective against this part. A group is said to have a *recursively enumerable word problem* if it has a recursively solvable answer of "yes" to the word problem and with a recursively unsolvable answer to the "no" part of the word problem.

Based on this general observation in the following section we design a cryptographic protocol with the following features:

1. Bob transmits to Alice an encrypted binary sequence that Alice decrypts correctly with probability "very close" to 1;

2. The adversary, Eve, who is granted arbitrarily high (but fixed) computational

> speed, cannot positively identify (at least, in theory), by using a brute force attack, the 1 or 0 bits in Bobs binary sequence. In other words: no matter what computational speed we grant Eve at the outset, there is no guarantee that her brute force attack program will give a conclusive answer (or an answer which is correct with overwhelming probability) about any bit in Bobs sequence.

Moreover we emphasize that there was an attempt to use the word problem in public key cryptography [48] a long-time ago, but it did not meet with success for several reasons. One of the reasons, which is relevant to the discussion above, was pointed out quite recently in [7], i.e., the problem that is actually used in [48] is not the word problem but the *word choice problem*: Given $g, w_1, w_2 \in G$, decide whether $g = w_1$ or $g = w_2$ is true in G whenever one of the two equations holds. This problem is recursively solvable for any recursively presented platform group G because one of these equations is the "yes" part of the word problem. Thus, the word choice problem cannot be utilized for our purpose; any comparison of it's implementation to our proposal is misleading; and our design seems to be the first based on an actual decision problem.

Finally, we note that a security model (named "bounded storage model") philosophically similar to ours has been considered in *private-key* cryptography, see [3] and references thereto. However, in [3] the "bounded storage model" utilizes a public bit string that is larger than the adversary's storage capacity. Therefore, their granted limit on the adversary's storage capacity is relatively small. In contrast, we grant the adversary "all storage space in the universe", so to speak, without losing efficiency in encryption or asking for a large storage space for legitimate parties.

1.2 The Word Problem Protocol

In this section we sketch of our cryptographic protocol and we leave the description of its details for the following sections.

Key Generation: Alice generates a group platform for anyone to encipher.

1. A pool of group presentations with efficiently solvable word problem is made public (e.g. is part of Alice's software).
2. Alice chooses randomly a particular presentation Γ from the pool, diffuses it by isomorphism-preserving transformations to obtain a diffused presentation Γ', discards some of the relators and publishes the abridged diffused presentation $\hat{\Gamma}$.

Encryption and Decryption: Let Bob encipher while Alice deciphers.

1. Bob transmits his private binary sequence to Alice by sending an element equal to 1 in $\hat{\Gamma}$ (and therefore also in Γ') in place of "1" and an element not equal to 1 in Γ' in place of "0".
2. Alice recovers Bob's binary sequence by first converting elements of $\hat{\Gamma}$ to the corresponding (under the isomorphism that she knows) elements of Γ, and then solving the word problem in Γ.

Each part of this protocol is rather nontrivial and opens several interesting research avenues. We discuss parts (1), (2), (3) in our Sections 1.3, 1.4, 1.5, respectively.

A priori it looks like the most nontrivial part is finding an element not equal to 1 in Γ' since Bob does not even know the whole presentation Γ'. We solve this problem

by "going with the flow", so to speak. More specifically, we just let Bob select a random (well, almost random) word of sufficiently big length and show that, with overwhelming probability, such an element is not equal to 1 in Γ'. We discuss this in more detail in Section 1.5.

We emphasize once again what is, in our opinion, the main advantage (at least, theoretical) of our protocol over the existing ones. The point is to deprive the adversary (Eve) from attacking the protocol by doing an "exhaustive search", which is the most obvious (although, perhaps, often "computationally infeasible") way to attack all existing public key protocol.

The way we plan to achieve our goal is relevant to the first part of the encryption and decryption part of the protocol, i.e., solving the word problem in $\hat{\Gamma}$. If Bob transmits an element g equal to 1 in $\hat{\Gamma}$, Eve will be able to detect it by going over all products of all conjugates of relators from $\hat{\Gamma}$ and their inverses. This set is recursive, but as we have pointed out in the Introduction, there are even groups G with efficiently solvable word problem and words w of length n equal to 1 in G, such that the length of a proof verifying that $w = 1$ in G is not bounded by any tower of exponents in n.

However, detecting an element g not equal to 1 in $\hat{\Gamma}$ is more difficult for Eve whenever Bob transmits it. In fact, it is impossible in general; Eve's only hope here is that she will be lucky to find a factor group of $\hat{\Gamma}$ where the word problem is solvable and $g \neq 1$ in that factor. This is what we call a *quotient attack*, (see Section 1.8.)

Remark. It may look like an encryption protocol with the features outlined in the Introduction cannot exist; in particular, the following attack by a computationally superior adversary may seem viable:

Eve can perform key generations over and over again, each time with

> fresh randomness, until the public key to be attacked is obtained—this will happen eventually with overwhelming probability. Already the correctness of the scheme (no matter if perfect or only with overwhelming probability) guarantees that the corresponding secret key (as obtained by Eve while performing key generation) allows to decrypt illegitimately.

This would be indeed viable if the correctness of the legitimate decryption by Alice was perfect. However, in our situation this kind of attack will fail for a general $\hat{\Gamma}$. Suppose Eve is building up two lists, corresponding to two possible encryptions "$0 \longrightarrow w \neq 1$ in $\hat{\Gamma}$" or "$1 \longrightarrow w = 1$ in $\hat{\Gamma}$" by Bob. Our first observation is that the list that corresponds to "$0 \longrightarrow w \neq 1$" is useless to Eve because it is simply going to contain all words in the alphabet $X = \{x_1, \ldots, x_n, x_1^{-1}, \ldots, x_n^{-1}\}$ since Bob is choosing such w simply as a random word. Therefore, Eve may just as well forget about this list and concentrate on the other one, which corresponds to "$1 \longrightarrow w = 1$".

Now the situation boils down to the following: if a word w transmitted by Bob appears on the list, then it is equal to 1 in Γ. If not, then not. The only problem is the following: how can Eve possibly conclude that w does not appear on the list if the list is infinite? Our opponent could say here that Eve can stop at some point and conclude that $w \neq 1$ with overwhelming probability, just like Alice does. The point is that this probability may not at all be as "overwhelming" as the probability of the correct decryption by Alice, however. Compare:

1. For Alice to decrypt correctly "with overwhelming probability", the probability $P_1(N)$ for a random word w of length N not to be equal to 1 should converge to 1 (reasonably fast) as N goes to infinity.

2. For Eve to decrypt correctly "with overwhelming probability", the probability $P_2(N, f(N))$ for a random word w of length N, which is equal to 1, generated by Bob's algorithm (see Section 1.5), to have a proof of length $\leq f(N)$ verifying that $w = 1$, should converge to 1 (reasonably fast) as N goes to infinity. Here $f(N)$ represents Eve's computational capabilities; this function can be arbitrary, but fixed. By "proof" here we mean a presentation of w as a product of conjugates of defining relators and their inverses.

We see that the functions $P_1(N)$ and $P_2(N)$ are of very different nature, and any correlation between them is unlikely. We note that the function $P_1(N)$ is generally well understood; in particular, it is known that in any infinite group G the probability $P_1(N)$ indeed converges to 1 as N goes to infinity; see our Section 1.5 for more details.

On the other hand, the functions $P_2(N, f(N))$ are more complex; they are currently subject of a very active research, and in particular, it appears likely that for any $f(N)$, there are groups in which $P_2(N, f(N))$ does not converge to 1 at all. Of course, $P_2(N, f(N))$ may depend on a particular algorithm used by Bob to produce words equal to 1, but we leave this discussion to another paper.

We also note, in passing, that if in a group G the word problem is recursively unsolvable, then the length of a proof verifying that $w = 1$ in G is not bounded by any recursive function of the length of w.

To conclude this section, we guide the reader to other sections of this paper where the questions of efficiency (for legitimate parties) are addressed. All steps of our protocol are shown to be quite efficient with the suggested parameters (the latter are summarized in Section 1.6). Step (2) of Key Generation of the protocol (Alice's algorithm for obtaining her public and private keys) is discussed in Section

1.4. Step (2) of the Encryption and Decryption part of the protocol (encryption by Bob) is discussed in Section 1.5. It turns out that encryption (of one bit) takes quadratic time in the length of a transmitted word; the latter is approximately 150 on average, according to our computer experiments. Step (2) of the Decryption part of the protocol (decryption by Alice) is discussed at the end of Section 1.4. It is straightforward to see that the time Alice needs to decrypt each transmitted word w is bounded by $C \cdot |w|$, where $|w|$ is the length of w and C is a constant which basically depends on Alice's private isomorphism between Γ and Γ'.

The fact that Alice (the receiver) and the adversary are separated in power is essentially due to Alice's knowledge of her private isomorphism between Γ and Γ' (note that Bob does *not* have to know this isomorphism for encryption!).

We have to admit here one disadvantage of our protocol compared to most well-established public-key protocols: we have encryption with a rather big "expansion factor". Computer experiments—with suggested parameters—show thatone bit in Bob's message gets encrypted into a word of length approximately 150 on average. This is the price we pay for granting the adversary vast computational power.

Finally, at the end of Section 1.5 we discuss semantic security.

1.3 Pool of Group Presentations

There are many classes of finitely presented groups with solvable word problem known by now, e.g. one-relator groups, hyperbolic groups, nilpotent groups, metabelian groups. Note however that Alice should be able to randomly select a presentation from the pool *efficiently*, which imposes some restrictions on classes of presentations that can be used in this context. The class of finitely presented groups that we suggest

to include in our pool is *small cancellation groups*.

Small cancellation groups have relators satisfying a simple (and efficiently verifiable) "metric condition" (we follow the exposition in [30]). More specificaly, let $F(X)$ be the free group with a basis $X = \{ x_i \mid i \in I \}$, where I is an indexing set. Let $\epsilon_k \in \{\pm 1\}$, where $1 \leq k \leq n$. A word $w(x_1, \ldots, x_n) = x_{i_1}^{\epsilon_1} x_{i_2}^{\epsilon_2} \cdots x_{i_n}^{\epsilon_n}$ in $F(X)$, with all x_{i_k} not necessarily distinct, is a *reduced X-word* if $x_{i_k}^{\epsilon_k} \neq x_{i_{k+1}}^{-\epsilon_{k+1}}$, for $1 \leq k \leq n-1$. In addition, the word $w(x_1, \ldots, x_n)$ is *cyclically reduced* if it is a reduced X-word and $x_{i_1}^{\epsilon_1} \neq x_{i_n}^{-\epsilon_n}$. A set R containing cyclically reduced words from $F(X)$ is *symmetrized* if it is closed under cyclic permutations and taking inverses.

Let G be a group with presentation $\langle X; R \rangle$. A non-empty word $u \in F(X)$ is called a *piece* if there are two distinct relators $r_1, r_2 \in R$ of G such that $r_1 = uv_1$ and $r_2 = uv_2$ for some $v_1, v_2 \in F(X)$, with no cancellation between u and v_1 or between u and v_2. The group G belongs to the class $C(p)$ if no element of R is a product of fewer than p pieces. Also, the group G belongs to the class $C'(\lambda)$ if for every $r \in R$ such that $r = uv$ and u is a piece, then $|u| < \lambda |r|$.

In particular, if G belongs to the class $C'(\frac{1}{6})$, then Dehn's algorithm solves the word problem for G efficiently. This algorithm is very simple: in a given word w, look for a "large" piece of a relator from R (that means, a piece whose length is more than a half of the length of the whole relator). If no such piece exists, then $w \neq 1$ in G. If such a piece does exist, call it u, then $r = uv$ for some $r \in R$, where the length of v is smaller than that of u. Then replace u by v^{-1} in w. The length of the resulting word is smaller than that of w; therefore, the algorithm will terminate in a finite number of steps with quadratic-time complexity relative to the length of w.

We also note that a generic finitely presented group is a small cancellation group

(cf. [2]). Then, to randomly select a small cancellation group, Alice can just take a few random words and check whether the corresponding symmetrized set satisfies the $C'(\frac{1}{6})$ condition; otherwise, repeat.

We conclude this section by providing a procedure for Alice to determine a presentation Γ from the protocol of Section 1.2.

1. Alice fixes a number k, $10 \leq k \leq 20$, of generators in her presentation Γ. Her Γ will therefore have generators $x_1, \ldots, x_k$.

2. Alice selects m random words $r_1, \ldots, r_m$ in the generators $x_1, \ldots, x_k$. Here $1 < m < k - 2$ and the lengths l_i of r_i are random integers from the interval $L_1 \leq l_i \leq L_2$. Particular values that we suggest are: $L_1 = 12$, $L_2 = 20$.

3. After Alice obtains the abridged presentation $\hat{\Gamma}$, she adds a relation

$$x_i' = \prod_{j=1}^{M} [x_i', w_j]$$

to it, where x_i' is a (randomly chosen) generator from $\hat{\Gamma}$, w_j are random elements of length 1 or 2 in the generators $x_1', x_2', \ldots$, and $M = 10$. (Our commutator notation is $[a, b] := a^{-1}b^{-1}ab$.) This relation is needed to foil *quotient attacks*, see Section 1.8. Then Alice finds the preimage of this relation under the isomorphism between Γ and $\hat{\Gamma}$ and adds this preimage to the defining relators of Γ. Thus, Γ finally has k generators and $m + 1 < k - 1$ defining relators. This ensures, in particular, that the group G defined by Γ is infinite.

4. Finally, Alice checks whether her private presentation Γ satisfies the small cancellation condition $C'(\frac{1}{6})$ (which it will with overwhelming probability, see [2]). If not, she has to restart.

1.4 Elementary Isomorphisms

In this section, we explain how Alice can implement the remaining steps of part (2) in the Key Generation part of our Protocol. First we introduce Tietze transformations; these are "elementary isomorphisms": any isomorphism between finitely presented groups is a composition of Tietze transformations. What is important to us is that every Tietze transformation is easily invertible, and therefore Alice can compute the inverse isomorphism that takes Γ' to Γ.

Tietze introduced isomorphism preserving elementary transformations that can be applied to groups presented by generators and relators. They are of the following types.

Tietze Transformations: Given $X := \{ x_1, x_2, \dots \}$, let $F(X)$ denote the free group generated by X.

(T1) *Introducing a new generator.* Replace $\langle x_1, x_2, \dots \, ; \, r_1, r_2, \dots \rangle$ as follows:

$$\langle x_1, x_2, \dots \, ; \, r_1, r_2, \dots \rangle \overset{\mathbf{T1}}{\longmapsto} \langle y, x_1, x_2, \dots \, ; \, ys^{-1}, r_1, r_2, \dots \rangle \,,$$

where $s = s(x_1, x_2, \dots) \in F(X)$.

(T2) *Canceling a generator.* Replace $\langle y, x_1, x_2, \dots \, ; \, q, r_1, r_2, \dots \rangle$ as follows:

$$\langle y, x_1, x_2, \dots \, ; \, q, r_1, r_2, \dots \rangle \overset{\mathbf{T2}}{\longmapsto} \langle x_1, x_2, \dots \, ; \, r_1, r_2, \dots \rangle \,,$$

provided $q = ys^{-1}$ and $s, r_1, r_2, \cdots \in F(X)$.

(T3) *Applying an automorphism.* Replace $\langle x_1, x_2, \dots \, ; \, r_1, r_2, \dots \rangle$ as follows:

$$\langle x_1, x_2, \dots \, ; \, r_1, r_2, \dots \rangle \overset{\mathbf{T3}}{\longmapsto} \langle x_1, x_2, \dots \, ; \, \varphi(r_1), \varphi(r_2), \dots \rangle \,,$$

where φ is an automorphism of $F(X)$.

(T4) *Changing defining relators.* Replace $\langle x_1, x_2, \dots \,;\, r_1, r_2, \dots \rangle$ as follows:

$$\langle x_1, x_2, \,;\, r_1, r_2, \dots \rangle \xmapsto{\textbf{T4}} \langle x_1, x_2, \,;\, r'_1, r'_2, \dots \rangle ,$$

where the normal closure of the set $\{ r'_1, r'_2, \dots \}$ is the same as the normal closure of the set $\{ r_1, r_2, \dots \}$.

Tietze proved that two group presentations of the form $\langle x_1, x_2, \dots \,;\, r_1, r_2, \dots \rangle$ and $\langle x_1, x_2, \dots \,;\, s_1, s_2, \dots \rangle$ are isomorphic if and only if one can get from one of the presentations to the other by a sequence of transformations (T1)–(T4), cf. [30].

For each Tietze transformation of the types (T1)–(T3), it is easy to obtain an explicit isomorphism (as a mapping on generators) and its inverse. For a Tietze transformation of the type (T4), the isomorphism is just the identity map. We would like here to make Tietze transformations of the type (T4) recursive, because *a priori* it is not clear how Alice can actually apply these transformations. Thus, Alice is going to use the following recursive version of (T4):

(T4′) In the set $r_1, r_2, \dots$, replace some r_i by one of the: r_i^{-1}, $r_i r_j$, $r_i r_j^{-1}$, $r_j r_i$, $r_j r_i^{-1}$, $x_k^{-1} r_i x_k$, $x_k r_i x_k^{-1}$, where $j \neq i$, and k is arbitrary.

We suggest that in part (2) of the protocol in Section 1.2, Alice should first apply several transformations of the type (T4′) to "mix" the presentation Γ. (This does not add complexity to the final isomorphism since for a Tietze transformation of the type (T4), the isomorphism is just the identity map, as we have noted above.) In particular, if Γ was a small cancellation presentation (see Section 1.3) to begin with, then after applying several transformations (T4′) it will, most likely, no longer be. As a result, Eve's chances of augmenting the public presentation $\hat{\Gamma}$ to a small cancellation presentation (see Section 1.7) become slimmer.

After Alice has mixed Γ by transformations (T4′), we suggest that she should aim for breaking down some of the defining relators into "small pieces". Formally, she can replace a given presentation by an isomorphic presentation where most defining relators have length at most 4. (Intuitively, diffusion of elements should be easier to achieve in a group with shorter defining relators). This is easily achieved by applying transformations (T1) (see below), which can be "seasoned" by a few elementary automorphisms (type (T3)) of the form $x_i \longmapsto x_i x_j^{\pm 1}$ or $x_i \longmapsto x_j^{\pm 1} x_i$, for better diffusion.

The procedure of breaking down defining relators is quite simple. Let Γ be a presentation $\langle x_1, \ldots, x_k; r_1, ..., r_m \rangle$. We are going to obtain a different, isomorphic, presentation by using Tietze transformations of types (T1). For example, let $r_1 = x_i x_j u$ where $1 \leq i, j \leq k$. We introduce a new generator x_{k+1} and a new relator $r_{m+1} = x_{k+1}(x_i x_j)^{-1}$. The new presentation $\langle x_1, \ldots, x_k, x_{k+1}; r_1, \ldots, r_m, r_{m+1} \rangle$ is isomorphic to Γ. Now if we replace r_1 with $r_1' = x_{k+1} u$, then the presentation $\langle x_1, \ldots, x_k, x_{k+1}; r_1', \ldots, r_m, r_{m+1} \rangle$ will again be isomorphic to Γ, but now the length of one of the defining relators (r_1) has decreased by 1. Continuing in this manner, Alice can eventually obtain a presentation where many relators have length at most 3, at the expense of introducing more generators. In fact, relators of length 4 are also good for the purpose of diffusing a given word, so we are not going to "cut" the relators into too small pieces —i.e., to avoid pieces of length 1 or 2— but rather settle with relators of length 3 or 4. Most of the longer relators can be discarded from the presentation Γ' to obtain the abridged presentation $\hat{\Gamma}$.

We conclude this section with a simple example, just to illustrate how Tietze transformations can be used to cut relators into pieces. In this example, we start

with a presentation having two relators of length 5 in 3 generators, and end up with a presentation having 4 relators of length 3 or 4 in 5 generators. The symbol $\simeq$ below means "is isomorphic to".

Example. Let G be a group with presentation $\langle x_1, x_2, x_3 \ ; \ x_1^2 x_2^3, \ x_1 x_2^2 x_1^{-1} x_3 \rangle$. Then

$$\begin{aligned} G &\simeq \langle x_1, x_2, x_3, x_4 \ ; \ x_4 = x_1^2, \ x_4 x_2^3, \ x_1 x_2^2 x_1^{-1} x_3 \rangle \\ &\simeq \langle x_1, x_2, x_3, x_4, x_5 \ ; \ x_5 = x_1 x_2^2, \ x_4 = x_1^2, \ x_4 x_2^3, \ x_5 x_1^{-1} x_3 \rangle \\ &\simeq \langle x_1, x_2, x_3, x_4, x_5 \ ; \ x_5 = x_2^2, \ x_4 = x_1^2, \ x_4 x_2^3, \ x_1 x_5 x_1^{-1} x_3 \rangle . \end{aligned}$$

The last isomorphism illustrates applying a transformation of type (T3), namely, the automorphism $x_5 \longmapsto x_1 x_5$ and $x_i \longmapsto x_i$ for $i \neq 5$.

1.5 Generating Random Elements in FPG

In this section, we explain how to implement the crucial step (1) for key generation of the protocol given in Section 1.2. We have to say up front that under the assumption that the adversary, Eve, has arbitrarily high computational speed, it does not matter how Bob is going to diffuse an element equal to 1 in $\hat{\Gamma}$; Eve will be able to detect triviality of his element anyway, see discussion in Section 1.2. Still, we include here a discussion of Bob's possible diffusion strategy because we believe it might be useful in real life, where Eve cannot enjoy arbitrarily high computational speed.

When Bob wants to transmit an element equal to 1 in $\hat{\Gamma}$, he should construct a word, looking "as random as possible" (for semantic security), in the relators $\hat{r}_1, \ldots, \hat{r}_l$ and their conjugates. When he wants to transmit an element not equal to 1 in $\hat{\Gamma}$, he just selects a random word of sufficiently big length; it turns out that, with

overwhelming probability, such an element is not equal to 1 in Γ' (we explain it in the end of this section).

It is rather straightforward to produce a random word of a given length in generators $x_1, \dots, x_k$, so we are not going to discuss it here. On the other hand, when Bob transmits a word w equal to 1 in $\hat{\Gamma}$, he wants to diffuse it so that pieces of defining relators would not be visible in w. In some specific groups (e.g. in braid groups) a diffusion is provided by a "normal form", which is a collection of symbols that uniquely corresponds to a given element of the group. The existence of such normal forms is usually due to some special algebraic or geometric properties of a given group.

However, since Bob does not know any meaningful properties of the group defined by the presentation $\hat{\Gamma}$ which is given to him, he cannot employ normal forms in the usual sense. The only useful property that the presentation $\hat{\Gamma}$ has is that most of its defining relators have length 3 or 4, see Section 1.3. We are going to take advantage of this property as follows. We suggest the following procedure which is probably best described by calling it "shuffling".

1. Make a product of the form $u = s_1 \cdots s_p$, where each s_i is randomly chosen among defining relators $r_1, r_2, \dots$, their inverses, and their conjugates by one- or two-letter words in $x_1, x_2, \dots$. The number p of factors should be sufficiently big, at least 10 times the number of defining relators.

2. Insert approximately $\frac{2p}{k}$ expressions of the form $x_j x_j^{-1}$ or $x_j^{-1} x_j$ in random places of the word u (here k is the number of generators x_i), for random values of j.

3. Going left to right in the word u, look for two-letter subwords that are parts of defining relators r_i of length 3 or 4. When you spot such a subword, replace it

by the inverse of the augmenting part of the same defining relator and continue. For example, suppose there is a relator $r_i = x_1x_2x_3x_4$, and suppose you spot the subword x_1x_2 in u. Then replace it by $x_4^{-1}x_3^{-1}$ (obviously, $x_1x_2 = x_4^{-1}x_3^{-1}$ in your group). If you spot the subword x_2x_3 in w, replace it by $x_1^{-1}x_4^{-1}$. If there is more than one choice for replacement, choose randomly between them.

4. Cancel remaining subwords (if any) of the form $x_jx_j^{-1}$ or $x_j^{-1}x_j$.

Steps (2)–(4) should be repeated approximately p times for good mixing. Finally, after Bob has obtained a word u this way, he sets $w = [x'_i, u]$ and applies steps (2)–(4) to w approximately $\frac{|w|}{2}$ times, where $|w|$ is the length of w. This final step is needed to make this w (which is equal to 1 in $\hat{\Gamma}$, and therefore also in Γ') indistinguishable from $w \neq 1$, which is constructed in the same form $w = [x'_i, u]$, see below. Here x'_i is the same as in the relator $x'_i = \prod_{j=1}^{M}[x'_i, w_j]$ published by Alice, see Section 1.3. Having $w \neq 1$ in this form is needed, in turn, to foil "quotient attacks", see the end of Section 1.8.

When Bob wants to transmit an element not equal to 1 in Γ', he should first choose a random word u from the commutator subgroup of the free group generated by $x'_1, x'_2, \ldots$. To select a random word from the commutator subgroup is easy; Bob can select an arbitrary random word v first, and then adjust the exponents on the generators in v so that the exponent sum on every generator in v is 0. The length of u should be in the same range as the lengths of the words u equal to 1 in $\hat{\Gamma}$ constructed by Bob before. Then Bob lets $w = [x'_i, u]$, where x'_i is the same as in the relator $x'_i = \prod_{j=1}^{M}[x'_i, w_j]$ published by Alice, see Section 1.3. Finally, to hide u, he applies "shuffling" to w (steps (2)–(4) above) approximately $\frac{|w|}{2}$ times.

Now we explain why a random word of sufficiently big length is not equal to 1 in

Γ with overwhelming probability, provided Γ is a presentation described in the end of Section 1.3.

Like any other group, the group G given by the presentation Γ is a factor group $G = F/R$ of the ambient free group F generated by $x_1, x_2, \ldots$. Therefore, to estimate the probability that a random word in $x_1, x_2, \ldots$ would not belong to R (and therefore, would not be equal to 1 in G), one should estimate the *asymptotic density* (see e.g. [24]) of the complement to R in the free group F. It makes notation simpler if one deals instead with the asymptotic density of R itself, which is

$$\rho_{_F}(R) = \limsup_{n\to\infty} \frac{\#\{u \in R : l(u) \le n\}}{\#\{u \in F : l(u) \le n\}}.$$

Here $l(u)$ denotes the usual lexicographic length of u as a word in $x_1, x_2, \ldots$. Thus, the asymptotic density depends, in general, on a free generating set of F, but we will not go into these details here because all facts that we are going to need are independent of the choice of basis. One principal fact that we can use here is due to Woess [50]: if the group $G = F/R$ is infinite, then $\rho_{_F}(R) = 0$. Since the group G given by the presentation Γ is infinite (see our Section 1.3), this already tells us that the probability for a random word of length n in $x_1, x_2, \ldots$ not to be equal to 1 in G is approaching 1 when $n \to \infty$. However, if we want words transmitted by Bob to be of reasonable length (on the order of 100–200, say), we have to address the question of *how fast* the ratio in the definition of the asymptotic density converges to 0 if R is the normal closure of the relators described in the end of Section 1.3. It turns out that for *non-amenable* groups the convergence is exponentially fast; this is also due to Woess [50]. We are not going to explain here what amenable groups are; it is sufficient for us to know that small cancellation groups are not amenable (because

they have free subgroups, see e.g. [30]). Thus, small cancellation groups are just fine for our purposes here: the probability for a random word of length n in $x_1, x_2, \ldots$ not to be equal to 1 in G is approaching 1 exponentially fast as $n \longrightarrow \infty$.

Finally, we touch upon semantic security (see [16]) of the words transmitted by Bob. We do not give any rigorous probabilistic estimates since this would require at least defining a probability measure on an infinite group, which is a very nontrivial problem by itself (cf. [8]). Instead, we offer here a "hand-waving" type of argument which we hope to be convincing, at least, to some extent. The nice thing about Bob's encryption procedure is that when he selects a word $u \neq 1$, he simply selects a random word. Thus, $u \neq 1$ is indistinguishable from a random word just because it is random! Then, the element $w = [x_i', u]$, which is transmitted by Bob, looks like it is no longer random because it is of a special form. However:

1. What is actually transmitted by Bob is a *word in the alphabet* $x_1, x_2, \ldots$ representing the element $w = [x_i', u]$ of the group defined by $\hat{\Gamma}$. This word is *not* of the form $[x_i', u]$ because Bob has applied a "shuffling" to w.

2. Given the specifics of our protocol, what really matters is that transmitted words equal to 1 in $\hat{\Gamma}$ are indistinguishable from transmitted words not equal to 1. This is why we require Bob's elements representing 1 in $\hat{\Gamma}$ to be of the form $[x_i', u]$ as well.

Thus, the question of semantic security of Bob's transmissions boils down to the following question of independent interest: is a word u representing 1 in $\hat{\Gamma}$ indistinguishable from a random word (of approximately the same length)? As we have admitted above, we do not have a rigorous proof that it is, but computer experiments

imply that when most of the relators in $\hat{\Gamma}$ have length at most 4, then words u representing 1 in $\hat{\Gamma}$, obtained as described earlier in this section, pass at least the equal frequency test for 1-, 2-, and 3-letter subwords, thus making it appear likely that the answer to the question above is affirmative for such $\hat{\Gamma}$.

1.6 Suggested Parameters

In this section, we summarize all suggested parameters of our protocol for the reader's convenience, although most of these parameters were already discussed in previous sections.

1. The number of generators x_i in Alice's private presentation Γ is k, a random integer from the interval $10 \leq k \leq 20$.

2. The relators $r_1, \ldots, r_m$ in Alice's private presentation Γ are random in the $x_1, \ldots, x_k$. The integer m is random from $1 < m < k - 2$, and the lengths l_i of r_i are also random from $12 \leq l_i \leq 20$. There is one other special relator in Γ, which is obtained as described at the end of Section 1.3.

3. Alice's private isomorphism (between the presentation Γ and Γ') is a product, in random order, of s_1 elementary transformations of type (T1) and s_2 elementary transformations of type (T3) (see Section 1.4). Neither of the parameters s_1, s_2 is specified, but their sum should be in the interval $20 \leq s_1 + s_2 \leq 30+$, where "30+" means that as soon as $s_1 + s_2$ becomes equal to 30, only transformations of type (T1) are applied, targeted at making all relators having length at most 4, as described in Section 1.4.

4. Bob encrypts his secret bits by words in the given alphabet, as described in Section 1.5. Here we specify the length of those words. Recall that Bob starts building a word $w = 1$ in Γ' as a product of p words randomly chosen among published defining relators, their inverses, and their conjugates by one- or two-letter words in the published generators. We specify p as a random integer from the interval $5 \leq p \leq 12$, thus making the whole w a word of length approximately 150 on average. Computer experiments show that subsequent "shuffling", as described in Section 1.5, only slightly increases the length of w.

   Finally, we recall that Bob selects a word $w \neq 1$ in Γ' in the form $[x'_i, u]$, where u is a random word from the commutator subgroup of the free group generated by the published generators $x'_1, x'_2, \ldots$. We therefore specify u as a random word of length l, where l is a random integer from the interval $65 \leq l \leq 85$. Then the length of $w = [x'_i, u]$ (which is $2l + 2$) is going to be approximately 150 on average, just as in the case of $w = 1$ considered above.

1.7 Isomorphism Attack

In this section, we discuss a "brute force" attack on the protocol from Section 1.2 and explain why arbitrarily high computational speed alone is insufficient to make this attack successful.

Knowing the pool of group presentations from which Alice selects her private presentation Γ, Eve can try to augment the public presentation $\hat{\Gamma}$ to a presentation that would be isomorphic to one from the pool. Theoretically, this is possible because the pool is recursive and because the set of finite presentations isomorphic to a given

one is recursive as well. Now let us take a closer look at this recursive procedure.

Eve can add to $\hat{\Gamma}$ one element at a time and check whether the resulting presentation, call it $\hat{\Gamma}_+$, is isomorphic to one of the presentations from Alice's pool. The latter is done the following way. Suppose Eve wants to check whether $\hat{\Gamma}_+$ is isomorphic to some Γ_i. She goes over mappings from Γ_i to $\hat{\Gamma}_+$, one at a time, defined on the generators of Γ_i. At the same time, she also goes over mappings from $\hat{\Gamma}_+$ to Γ_i defined on the generators of $\hat{\Gamma}_+$. She composes various pairs of those mappings and checks whether she gets the identical mapping on Γ_i. Having the word problem in Γ_i solvable makes the latter checking more efficient, but it is, in fact, not necessary because what matters here is the "yes" part of the word problem, which is always recursive.

Now let us focus on the part of this procedure where Eve works with a particular presentation Γ_i from Alice's pool. Suppose Γ_i is not isomorphic to $\hat{\Gamma}_+$. Since the "no" part of the isomorphism problem between $\hat{\Gamma}_+$ and Γ_i is not recursive, Eve would have to try out various pairs of mappings between $\hat{\Gamma}_+$ and Γ_i (see above) indefinitely. Therefore, she will have to allocate (indefinitely) some memory resources to checking this particular Γ_i. Since the number of Γ_i grows exponentially with the size of the presentation (which is the total length of relators), Eve would require essentially unlimited storage space and, in fact, she will reach physical limits on the storage space very quickly, e.g., the number of presentations on 6 generators with the total length of relators bounded by 100 is already more than 10^{100}.

We note that there seems to be no way to bypass these memory size requirements, even assuming that Eve enjoys arbitrarily high computational speed. She basically has two options: (1) to run in parallel subroutines corresponding to each Γ_i, until

one of the subroutines finds Γ_i isomorphic to $\hat{\Gamma}_+$; (2) to enumerate all steps in all subroutines the same way that one enumerates rational numbers (so that Step i in the Subroutine j corresponds to the rational number $\frac{i}{j}$).

The first option obviously requires unlimited storage space. With the second option, it may look like just having unlimited computational speed would be sufficient for Eve. However, since under this arrangement Eve would have to interrupt each subroutine and then return to it, she would have to at least store the information indicating where she left off each particular subroutine. Therefore, since the number of subroutines is unlimited, she would still need unlimited storage space.

1.8 Attack

In this last section for this chapter we discuss an attack that is, in general, more efficient (especially in real life) than the "brute force" attack described in Section 1.7. We use here some group-theoretic terminology not supported by formal definitions when we feel it should not affect the reader's understanding of the material. Some of the basic terminology however has to be introduced.

Recall that a group G is called *abelian* (or commutative) if $[a, b] = 1$ for any two elements $a, b \in G$, where $[a, b] := a^{-1}b^{-1}ab$ as described in page 18. This can be generalized in different ways. A group G is called *metabelian* if $[[x, y], [z, t]] = 1$ for any $x, y, z, t \in G$. A group G is called *nilpotent of class* $c \geq 1$ if $[y_1, y_2, \ldots, y_{c+1}] = 1$ for any $y_1, y_2, \ldots, y_{c+1} \in G$, where $[y_1, y_2, y_3] = [[y_1, y_2], y_3]$, etc.

We note that, in the definition of an abelian group, it is sufficient to require that $[x_i, x_j] = 1$ for all *generators* x_i, x_j of the group G. Thus, *any finitely generated abelian group is finitely presented.* The same is true for all finitely generated nilpotent

groups of any class $c \geq 1$, but not for all metabelian groups. In particular, it is known that finitely generated *free metabelian groups* are infinitely presented, cf. [6]. (A free metabelian group is the factor group $F/[[F,F],[F,F]]$ of a free group by the second commutator subgroup.)

Now we get to quotient attacks. One way for Eve to try to positively identify those places in Bob's binary sequence where he intended to transmit a 0 is to use a *quotient test* (see e.g. [24] for a general background). That means the following: Eve tries to add finitely or infinitely many relators to the given presentation $\hat{\Gamma}$ to obtain a presentation defining a group H with solvable word problem (more accurately, a group H that Eve can *recognize* as having solvable word problem).

It makes sense for Eve to only try recognizable quotients, such as, for example, abelian or, more generally, nilpotent ones. This amounts to adding specific relators to $\hat{\Gamma}$; for example, for an abelian quotient, Eve can add relators $[x'_i, x'_j]$ for all pairs of generators x'_i, x'_j in $\hat{\Gamma}$. For nilpotent quotients, Eve will have to add commutators of higher weight in the generators. For a metabelian quotient, Eve will have to add infinitely many relators (because, as we have already mentioned, free metabelian groups are infinitely presented), but this is not a problem since she does not have to "actually add" those relators; she can just consider $\hat{\Gamma}$ as a presentation *in the variety of metabelian groups* and apply the relevant algorithm for solving the word problem which is universal for all groups finitely presented in the variety of metabelian groups.

Note that this trick will *not* work with hyperbolic quotients, say. This is because there is no way, in general, to add specific relators (finitely or infinitely many) to $\hat{\Gamma}$ to make sure that the extended presentation defines a hyperbolic group. This deprives Eve from using a (rather powerful, cf. [24]) hyperbolic quotient attack.

Classes of groups with solvable word problem are summarized in the survey [25]. It appears that a quotient attack can essentially employ either a nilpotent or a metabelian quotient of $\hat{\Gamma}$. This is why, to foil such attacks, Alice adds a relator $x'_i = \prod_{j=1}^{M}[x'_i, w_j]$ to $\hat{\Gamma}$ (see our Section 1.3). This is also the reason why Bob should choose a word of the form $[x'_i, u]$ when he wants to transmit an element not equal to 1 in Γ' (see Section 1.5). Indeed, a metabelian quotient attack on an element of the form $[x'_i, u]$ will not work because this element belongs to the second commutator subgroup of the group defined by $\hat{\Gamma}$ since in this group, $x'_i = \prod_{j=1}^{M}[x'_i, w_j]$, so x'_i belongs to the commutator subgroup of the given group. Furthermore, an element of the form $[x'_i, u]$ belongs to every term of the lower central series of the given group since in this group, $[x'_i, u] = [\prod_{j=1}^{M}[x'_i, w_j], u] = [\prod_{j=1}^{M}[\prod_{j=1}^{M}[x'_i, w_j], w_j], u]$, etc. This foils nilpotent quotient attacks as well.

Chapter 2

The membership search problem in public-key cryptography

The majority of public-key protocols based on non-abelian groups use the computational hardness of either the *conjugacy search problem* [1, 26] or the word (search) problem [7, 15, 48]. In this second chapter we describe a cryptosystem whose security is based on the following computational hardness:

Definition 2.0.1. Let G be a group, H a subgroup of G generated by $\{h_1, \ldots, h_k\}$, and let $h \in H$. The *subgroup membership (search) problem* is the problem of finding an expression of h in terms of generators from $\{h_1, \ldots, h_k\}$.

It should be mentioned that, in fact, the Anshel-Anshel-Goldfeld protocol which may seem to rely on the computational hardness of the conjugacy search problem alone, actually relies (perhaps somewhat implicitly) on the hardness of the subgroup membership (search) problem as well, as explained in [43]. We also note that there is some similarity, at a philosophical level, between our cryptosystem and homomorphic public-key cryptosystems of [18] and [19]. However, there are also essential differences, which are explained in Section 2.2.

First we outline the ideas behind our cryptosystem. These ideas can be traced

back to [34], where a similar approach was used in a commutative situation but it was not accepted by the cryptographic community as secure [17]. We believe that employing a non-abelian group instead of a polynomial algebra as the platform does make a difference in both the efficiency and security components.

2.1 Automorphisms of Relatively-Free Groups

Definition 2.1.1. Let F be a free group and let R be a normal subgroup of F. The factor group F/R is called *relatively free* if R is fully invariant, i.e., if $\alpha(R) \leq R$ for any endomorphism α of F. If $x_1, \ldots, x_n$ are free generators of F, then $x_1R, \ldots, x_nR$ are called relatively free generators of F/R.

We are going to denote the relatively free generators of F/R simply by $x_1, \ldots, x_n$ when there is no ambiguity. Let $\mathcal{F}_n$ denote a relatively free group of rank n, i.e., $\mathcal{F}_n = F_n/R$ for some fully invariant R. Then any map on its generators into $\mathcal{F}_n$ can be extended to an endomorphism of $\mathcal{F}_n$. Because of this property, any relatively free group $\mathcal{F}_n$ can be a candidate platform of our cryptosystem.

Definition 2.1.2. Let $X := \{x_1, \ldots, x_n\}$ be a set of generators of F_n. A set of *Nielsen automorphisms* is a set of automorphisms of F_n engendered by the maps $\alpha_j, \beta_{jk} : X \longrightarrow F_n$, where $1 \leq i, j, k \leq n$, such that

$$\alpha_j : x_i \longmapsto \begin{cases} x_i^{-1} & \text{if } i = j \\ x_i & \text{if } i \neq j \end{cases} \quad \text{and} \quad \beta_{jk} : x_i \longmapsto \begin{cases} x_i x_j & \text{if } i = k \\ x_i & \text{if } i \neq k\,. \end{cases}$$

The set of Nielsen automorphisms $\{\, \alpha_j, \beta_{jk} \mid 1 \leq i, j, k \leq n \,\}$ generate the whole automorphism group of F_n. Similar to free groups, Nielsen automorphisms can be defined for any relatively free group $\mathcal{F}_n$ in the same manner, generating a subgroup

of the automorphism group $\mathrm{Aut}\,(\mathcal{F}_n)$ of $\mathcal{F}_n$ called the group of *tame* automorphisms of $\mathcal{F}_n$. In some cases, it is equal to the whole $\mathrm{Aut}\,(\mathcal{F}_n)$ (see e.g. [4]); in other cases, it is a proper subgroup of $\mathrm{Aut}\,(\mathcal{F}_n)$ (see e.g. [9, 41]). For our purposes, it is important that *free solvable groups* of derived length ≥ 3 (see Section 2.3) have many non-tame automorphisms by [41]:

Theorem 2.1.3 (Shpilrain [41])**.** *Let $\mathcal{F}_n = F_n/[R,R]$, and let $u \in R$. If $R \leq \gamma_3(F_n)$ and $n \geq 2$, then the following $\alpha_{u,j} \in \mathrm{Aut}\,(\mathcal{F}_n)$ is not tame:*

$$\alpha_{u,j} : x_i \longmapsto \begin{cases} x_j[x_j, u, x_j] & \text{if } i = j \\ x_i & \text{if } i \neq j\,. \end{cases}$$

Moreover, $\alpha_{u,j}^{-1} : x_j \longmapsto x_j[x_j, u^{-1}, x_j]$ for $i = j$, and $\alpha_{u,j}^{-1} : x_i \longmapsto x_i$ for $i \neq j$.

Now suppose $\langle x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n+m}\,;\, R\rangle$ is a presentation for a relatively free group $\mathcal{F}_{n+m}$, and let $\varphi \in \mathrm{Aut}\,(\mathcal{F}_{n+m})$ be defined as

$$\varphi\ : x_i \longmapsto y_i\,, \quad \text{where} \quad y_i = y_i(x_1, \ldots, x_{n+m})\,.$$

Notice that φ acts on the set $\mathcal{F}_{n+m}^{n+m}$, the direct product of $n+m$ copies of $\mathcal{F}_{n+m}$, in a natural way:

$$\varphi : \langle x_1, \ldots, x_{n+m}\rangle \longmapsto \langle y_1, \ldots, y_{n+m}\rangle\,,$$

and, more generally, it operates on $\mathcal{F}_{n+m}^{n+m}$ as

$$\varphi : \langle u_1, \ldots, u_{n+m}\rangle \longmapsto \langle\, y_1(u_1, \ldots, u_{n+m}), \ldots, y_{n+m}(u_1, \ldots, u_{n+m})\,\rangle$$

for any $u_i = u_i(x_1, \ldots, x_{n+m}) \in \mathcal{F}_{n+m}$. In particular we can restrict the operation of φ on $\mathcal{F}_{n+m}^{n+m}$ to an operation $\hat{\varphi}$ on subgroup of $\mathcal{F}_{n+m}^{n+m}$ consisting of elements of the

form $\langle u_1(x_1,\ldots,x_n),\ldots,u_n(x_1,\ldots,x_n),1,\ldots,1\rangle$, which is a subgroup isomorphic to $\mathcal{F}_n^n$ so that

$$\hat{\varphi}:\langle u_1,\ldots,u_n,1,\ldots,1\rangle \longmapsto \langle \hat{y_1}(u_1,\ldots,u_n),\ldots,\hat{y}_{n+m}(u_1,\ldots,u_n)\rangle$$

and where $\hat{y}_i(u_1,\ldots,u_n)$ denotes $y_i(u_1,\ldots,u_n,1,\ldots,1)$.

We note that $\hat{\varphi}$ is a one-to-one map because the restriction of a one-to-one map to a subgroup is itself one-to-one. This property will be important to us in Section 2.2. At the same time, there is no visible way of recovering φ from $\hat{\varphi}$.

2.2 The protocol

In this section we present our protocol, alluding to the specifics of the group platform while discussing the specifics for the choice of platform in the following Section 2.3.

Key Generation: Alice generates a public key for Bob to encipher by using the class of relatively free groups $\mathcal{F}_n$.

1. Alice privately chooses an automorphism $\varphi \in \mathrm{Aut}\,(\mathcal{F}_{n+m})$ as a random product of tame automorphisms and some easily invertible automorphisms of the type given in the theorem of Section 2.1, i.e., $\varphi = \tau_1 \cdots \tau_k$, and she sets φ^{-1} as her private-decryption key.

2. Let $y_i := y_i(x_1,\ldots,x_{n+m}) = \varphi(x_i)$. Alice publishes the collection of words

$$\hat{y}_i(x_1,\ldots,x_n) := y_i(x_1,\ldots,x_n,1,\ldots,1)$$

as her public-encryption key, where $i = 1,...,n+m$.

Encryption and Decryption Bob's plaintext is an element w from the subgroup of $\mathcal{F}_{n+m}^{n+m}$ and of the form

$$w = \langle\, w_1, \ldots, w_n \,\rangle := \langle w_1(x_1, \ldots, x_n), \ldots, w_n(x_1, \ldots, x_n), 1, \ldots, 1 \rangle\,.$$

1. Encryption is defined by

$$w \longmapsto \hat{\varphi}(w) = \langle\, \hat{y}_1(w_1, \ldots, w_n), \ldots, \hat{y}_{n+m}(w_1, \ldots, w_n) \,\rangle\,.$$

   Bob then transmits $\hat{\varphi}(w)$ as a tuple of words in the alphabet X.

2. Decryption is defined by

$$\hat{\varphi}(w) \longmapsto \varphi^{-1} \circ \hat{\varphi}(w) = w \quad (\text{modulo } x_{n+1} = ... = x_{n+m} = 1\,).$$

   This w, or more precisely, a *normal form* of w (see the end of Section 2.3), is Alice's and Bob's common secret key.

Remark. the encryption of w above is *not* a homomorphic image (or a preimage) of w, in contrast to homomorphic public-key cryptosystems of [18] and [19].

Observe that our adversary, Eve, can recover the plaintext w if she finds an expression for each $x_1, \ldots, x_n$ in terms of $\hat{y}_1, \ldots, \hat{y}_{n+m}$, i.e., if she solves the membership-search problem for the subgroup generated by $\hat{y}_1, \ldots, \hat{y}_{n+m}$ (this subgroup is actually $\mathcal{F}_n$). Then from

$$x_i = u_i(\hat{y_1}(x_1, \ldots, x_n), ..., \hat{y_n}(x_1, \ldots, x_n))$$

she can get

$$w_i = u_i(\hat{y_1}(w_1, \ldots, w_n), ..., \hat{y_n}(w_1, \ldots, w_n)).$$

However, if Eve wants to completely break the cryptosystem, i.e., to get the private decryption key, then she has to find the automorphism φ (along with its

inverse) based on the restriction $\hat{\varphi}$. This problem looks unapproachable to us by any deterministic method other than trying out products of "elementary" automorphisms of $\mathcal{F}_{n+m}$ until the one with the right restriction is found, which is computationally infeasible for large n and m. We discuss specific values of the parameters of our cryptosystem in Section 2.4.

We now give a "toy example" just to illustrate how our protocol works. The example is not platform-specific because here we only use Nielsen automorphisms as building blocks.

Example 2.2.1. Let $n = 2, m = 1$, and let β_{jk} be Nielsen automorphisms of $\mathcal{F}_3$ defined in the Introduction. Now if Alice chooses $\varphi = \beta_{23}^2\beta_{12}\beta_{31}\beta_{32}$, its action on the generators is

$$\varphi : \langle x_1, x_2, x_3\rangle \longmapsto \langle x_3x_2^2,\ x_2x_1x_3x_2^2,\ x_3x_2^2\rangle$$

Then $\langle y_1, y_2, y_3\rangle = \langle x_1x_3x_2^2,\ x_2x_1x_3x_2^2,\ x_3x_2^2\rangle$, and $\langle \hat{y_1}, \hat{y_2}, \hat{y_3}\rangle = \langle x_1x_2^2,\ x_2x_1x_2^2,\ x_2^2\rangle$. Therefore Bob's private message $(u_1(x_1, x_2), u_2(x_1, x_2))$ is encrypted as

$$\langle u_1u_2^2,\ u_2u_1u_2^2,\ u_2^2\rangle = \langle v_1, v_2, v_3\rangle\,.$$

However, in this basic example, eve can easily obtain x_1 and x_2 from the public $\hat{y_i}$'s, i.e., $x_1 = \hat{y_1}\hat{y_3}^{-1}$ and $x_2 = \hat{y_2}\hat{y_1}^{-1}$, which she then uses to recover $u_1 = v_1v_3^{-1}$ and $u_2 = v_2v_1^{-1}$.

As a comment to this example, we note that in a free group, the subgroup membership search problem can be efficiently solved by Nielsen's method (see e.g. [30]). However, adding automorphisms of the type $\alpha_{u,j}$ described in the Theorem from Section 2.1 makes a difference, since it makes working in a free group pointless.

We illustrate this fact with the previous example above: if we compose the automorphism φ with the map

$$\alpha : x_i \longmapsto \begin{cases} x_1[x_1,\,[x_1,x_2],\,x_1] & \text{if } i=1 \\ x_i & \text{if } i\neq 1\,, \end{cases}$$

again, where $[u,v] := u^{-1}v^{-1}uv$ denotes the commutator of u and v. The resulting automorphism takes the form

$$\langle x_1,x_2,x_3\rangle \longmapsto \langle y_1,y_2,y_3\rangle = \langle x_1[x_1,[x_1,x_2],x_1]x_3x_2^2,\; x_2x_1[x_1,[x_1,x_2],x_1]x_3x_2^2,\; x_3x_2^2\rangle\,,$$

where we get $\langle \hat{y_1},\hat{y_2},\hat{y_3}\rangle = \langle x_1[x_1,[x_1,x_2],x_1]x_2^2,\; x_2x_1[x_1,x_2],x_1]x_2^2,\; x_2^2\rangle$. Now in the free group F_2, the elements x_1 and x_2 no longer belong to the subgroup generated by $\hat{y_1},\hat{y_2},\hat{y_3}$. At the same time, since α is an automorphism of the free metabelian group F_3/F_3'' (see the next section), both x_1 and x_2, considered as elements of F_3/F_3'', belong to the subgroup of F_3/F_3'' generated by $\hat{y_1},\hat{y_2},\hat{y_3}$. Therefore, our adversary Eve will have to solve the subgroup membership-search problem in a free metabelian group, which is much more difficult than in a free group.

2.3 Free Solvable Groups

In this section we suggest the class of relatively free groups known as *free solvable groups* as an implementable pool of group platforms for the protocol described in the previous section.

Given a group G, the commutator subgroup of G, denoted by G', is the group generated by all its commutators. Furthermore, inductively, we can define the k^{th} term of the *derived series* of G by setting

$$G^{(1)} := G' \quad \text{and} \quad G^{(k+1)} := [G^{(k)},\, G^{(k)}]$$

for $k \geq 1$. Notice that one has $\alpha([u,v]) = [\alpha(u), \alpha(v)]$ for any endomorphism α of G; hence, $G^{(k)}$ is a fully invariant subgroup of G for any $k \geq 1$.

Definition 2.3.1. Given a the free group F_n of rank n, the *free metabelian group* of rank n, denoted by M_n, is the relatively free group $F_n/F_n'' := F_n/F_n^{(2)}$. Moreover, without loss of generality, the *free solvable group* of rank n and derived length k, denoted by $S_n^{(k)}$, is the relatively free group $F_n/F_n^{(k)}$.

Our choice of a free solvable group for platform is motivated by the following facts:

(1) The word problem in $S_n^{(k)}$ is solvable in time $O(m^k n)$ with respect to the length m of a given word. Moreover, every element of $S_n^{(k)}$ has a unique associated "normal form", which makes it possible to use our protocol for an efficient encryption/decryption of actual messages without prior common-key establishment.

(2) $S_n^{(k)}$ has exponential growth (if $k \geq 2$) which provides for an exponential (with respect to the key size) key space.

(3) Subgroup membership (search) problem in $S_n^{(k)}$ has no known polynomial-time solution if $k \geq 2$.

Fact (2) is a well-known (cf. [33]), so we leave it without any further comments. As for (3), there are two different solutions of the membership (decision) problem in M_n known by now [11, 39], but none of them is "even close" to yielding a polynomial-time algorithm for solving the membership search problem. No algorithm is known for solving the membership (decision) problem in $S_n^{(k)}$ for $k \geq 3$. We give more details in Section 2.4; here we concentrate on the word problem in M_n.

It is also interesting to note that free solvable groups $S_n^{(k)}$ are infinitely presented if $k \geq 2$ (see e.g. [6]), i.e., they cannot be defined by finitely many relations, in

contrast with groups typically used in public-key cryptography. Nevertheless, as we shall see in this section, these groups have efficient (and, in fact, very easy) solvable word problem. In order to demonstrate this statement we introduce *Fox derivatives*, which are noncommutative analogs of usual Leibniz derivatives.

Definition 2.3.2. Let $\mathbb{Z}F$ be the group ring of a free group F generated by the set $\{x_1, x_2, ...\}$. A *Fox derivative*[1] with respect to x_i is a map $\partial_{x_i} : F \longrightarrow \mathbb{Z}F$ such that for any $v, w \in F$ satisfies $\partial_{x_i}(vw) = \partial_{x_i}(v) + v\partial_{x_i}(w)$ and $\partial_{x_i}(x_j) = \delta_{ij}$, where δ_{ij} denotes the Kronecker delta.

Example 2.3.1. Let $g \in F$ and let 1 be the identity of F. Since $\partial(1) = \partial(1) + \partial(1)$, it follows that $\partial(1) = 0$. Therefore $\partial(gg^{-1}) = \partial(g) + g\partial(g^{-1}) = 0$, which implies $\partial(g^{-1}) = -g^{-1}\partial(g)$.

Example 2.3.2. Let x and y be generators of $F(x, y)$. Then $\partial_x\big([x, y]\big)$ is given as

$$\begin{aligned}\partial_x(x^{-1}y^{-1}xy) &= \partial_x(x^{-1}) + x^{-1}\Big(\partial_x(y^{-1}) + x^{-1}y^{-1}\partial_x(x) + x^{-1}y^{-1}x\partial_x(y)\Big)\\ &= -x^{-1} + x^{-1}y^{-1}\,.\end{aligned}$$

Now let F_{ab} denote the abelianization of a free group F, i.e., the factor group F/F', and let $\alpha : F \longrightarrow F_{ab}$ be the natural epimorphism—while noting that it can be extended to the group ring as $\alpha : \mathbb{Z}F \longrightarrow \mathbb{Z}F_{ab}$ by linearity.

Proposition 2.3.3. *Let $w \in F_n$. Then $w \in F_n''$ if and only if $\alpha(\partial_{x_i}(w)) = 0$ for each generator x_i of F_n.*

For a proof of proposition 2.3.3 see [21]. This Proposition allows us to determine a simple algorithm for solving the word problem in a free metabelian group M_n. In

[1] Fox derivatives extended to the entire groupring $\mathbb{Z}F$ by linearity.

other word, given $w \in M_n$, as a word in relatively free generators x_i, one considers w as an element of the free group F_n with the same set of free generators, computes $\partial_{x_i}(w)$ for each x_i, and checks whether or not all of them abelianize to 0. The latter is straightforward since the word problem in the free abelian group F_{ab} is easily solvable. This algorithm is not only simple but efficient as well:

Theorem 2.3.4. *The algorithm for solving the word problem in M_n based on Proposition 2.3.3 has at most quadratic time complexity with respect to the length of the input word.*

Proof. Let $w \in F_n$ and let $|w| = m$ denote the usual lexicographic length of the word w. The computation of $\partial_{x_i}(w)$, for any generator x_i, produces at most m summands in the free group ring $\mathbb{Z}F_n$. Thus, the computation of ∂_x has at most linear time complexity with respect to m. Then, deciding whether or not the abelianization of $\partial_{x_i}(w)$ is 0 amounts to collecting summands of the form $c \cdot h_i,\ c \in \mathbb{Z}, h_i \in F_n$, such that all h_i have the same abelianization. This is achieved by rewriting every h_i in the form $x_1^{a_1}x_2^{a_2} \cdot \ldots \cdot x_n^{a_n} \cdot u_i$, where $u_i \in F_n'$. Since any h_i has length $\leq m$ and the number of h_i is at most m, this part of the algorithm takes time $O(m^2)$, completing the proof. □

We also note that to any element w of the free metabelian group M_n, one can associate a unique "normal form", which is a vector of n abelianized partial Fox derivatives of the word w considered an element of the free group F_n. By Proposition 2.3.3, two vectors corresponding to two elements $w_1, w_2 \in M_n$ are equal if and only if $w_1 = w_2$ in M_n. The presence of a unique normal form makes it possible for Alice and Bob to make the exchange of information more efficient than it would be if it was based simply on the solvability of the word problem. (In the later case, if Alice

and Bob have arrived at a point where Alice has an element, say, u, and Bob has an element v such that $u = v$ in the platform group G, they can communicate as follows. Bob chooses, privately, a finite binary sequence $b_1, b_2, ...$, which is going to be his secret message to Alice. He then transmits a sequence of group elements $u_1, u_2, ...$ such that $u_i = v$ in G if and only if $b_i = 1$. Alice recovers the sequence $b_1, b_2, ...$ by comparing $u_1, u_2, ...$ to her u.)

To conclude this section, we mention that the word problem in any free solvable group of derived length k can be solved along the same lines, but the complexity would depend on k; namely, it would have time complexity $O(m^k)$.

2.4 Cryptanalysis

There are two visible ways to attack our cryptosystem: (1) trying to find the automorphism φ (and its inverse) based on the restriction $\hat{\varphi}$ and (2) trying to solve the subgroup membership (search) problem in the platform group. As we have already pointed out in Section 2.2, the first way looks intractable to us, so we focus on the subgroup membership problem here, but first we make a relevant remark about the word problem.

In the previous section, we obtained the solvability of the word problem in free metabelian groups M_n by using Fox derivatives. We indicate that the solvability of the word problem in M_n can also be obtained by a much less efficient but nevertheless interesting method, using the fact that M_n are *residually finite groups* [23].

Definition 2.4.1. A group G is said to be *residually finite* if for every $g \in G - \{1\}$ there exists a finite group H_g and a homomorphism $\varphi : G \longrightarrow H_g$ such that $\varphi(g) \neq 1$.

If a group $G = F/R$ is residually finite, then it has the word problem solvable as follows. Given $g \in G$ as a word in the generators of F, two algorithms run in parallel. One of them goes over all finite homomorphic images of G (those are recursively enumerable) and checks whether $\varphi(g) \neq 1$ in any of them (note that the word problem is solvable in any finite group). Thus, if $g \neq 1$ in G, this algorithm will eventually detect that. The other algorithm just recursively enumerates all elements of R and compares them to g one at a time. Thus, if $g = 1$ in G, this algorithm will eventually detect it.

This way of solving the word problem is "cute", but it is obviously useless in applications, albeit, we mentioned because there is a similar (equally useless) way of solving the membership problem in a special class of groups.

Definition 2.4.2. A group G is said to be *locally extended residually finite* (LERF) if for every subgroup $K \in G$ and $g \in G - K$, there exists a finite group H and a homomorphism $\varphi : G \longrightarrow H$ such that $\varphi(g) \in \varphi(G) - \varphi(K)$.

Coulbois showed that free metabelian groups are LERF (cf. [11].) If a group G is LERF, it follows that G has solvable subgroup membership (decision) problem by way of an algorithm similar to what was described above. However, this algorithm is anything but practical, as well as the algorithm for solving the word problem based on residual finiteness of a given group.

An alternative algorithm for solving the subgroup membership (both decision and search) problem in M_n was offered in [39], which is somewhat more "down-to-earth". However it is still far from being practical because it involves, among other things, computing a Gröbner basis of an ideal of the ring of Laurent polynomials in n variables.

In the absence of feasible deterministic attacks, one might try heuristic ("length based") attacks in the spirit of [14]. We point out, however, that groups that were typically used as proving ground for "length based" attacks are very different in nature from free solvable groups; the length function for free solvable groups is, informally speaking, much less predictable than it is, say, for braid groups, which probably means that "length based" attacks are going to be less successful.

2.5 Suggested Parameters

We culminate this chapter by specifying the parameters of our cryptosystem. First, we suggest the class of free metabelian group as the choice a platform group, albeit, there is an approach to solving the subgroup membership problem in such a group mentioned above. We believe that this approach has only a theoretical advantage over the exhaustive search, and, therefore, it is not a threat to the security of the cryptosystem. Another property of free metabelian groups that distinguishes them from other free solvable groups is that they only have tame automorphisms [4]. However, automorphisms described in the Theorem in Section 2.1, although tame in the case of a free metabelian group, are tame in a very nontrivial ("nonmonotonic") way, i.e., to factor them into a product of Nielsen automorphisms one has to first increase the length of $\alpha(x_i)$ before it could be decreased. This is supposed to foil "length based" attacks on φ.

Thus, in the interests of efficiency, we suggest to use a free metabelian group of a fairly small rank as platform, where the rank satisfies $r = n + m = 10$ for $n = 8,\ m = 2$.

An important parameter is the number of "elementary" automorphisms in the factorization $\varphi = \tau_1 \cdots \tau_k$ of the key automorphism φ. One has to be careful here because, say, the automorphism group of a free group has exponential growth with respect to the (finite) generating set that consists of all Nielsen automorphisms. This implies, in particular, that the length of $\varphi(x_i)$ grows exponentially with k. However, in the free group of rank $r = 10$ the base of this exponent is so small (the higher the rank the smaller the base) that, according to our experiments, for a random product of 30 Nielsen automorphisms the average length of $\varphi(x_i)$ is under 30, which is quite reasonable.

For these reasons we suggest $k = 30$, of which 90% should be random Nielsen automorphisms and 10% automorphisms of the type described in Theorem from Section 2.1. That means, when building the automorphism φ as a product of "elementary" automorphisms one factor at a time, Alice selects, at each step, a Nielsen automorphism with probability 90% and an automorphism of type $\alpha_{u,j}$ with probability 10%.

This provides for sufficiently large key space because there are approximately 100 Nielsen automorphisms in the free (or relatively free) group of rank 10, so products of 30 Nielsen automorphisms already give us approximately $100^{30} = 10^{60}$ choices for φ. On top of that, there are automorphisms $\alpha_{u,j}$ with arbitrary $u \in [F_r, F_r]$. The length of u has to be bounded by, say, 10, to keep the complexity of φ within reasonable limits. This leaves us with the pool of approximately 10^7 elements to choose u from.

With these parameters at hand the average total length of $\varphi(x_i)$ is under 1000, where $i \in \{1, ..., 10\}$, according to our experiments.

Chapter 3

Rewriting with Transversals

In this chapter describe a method of rewriting elements in a group that will be essential to Chapter 4. We accomplish this aim by capitalizing from the canonical action a group G has on itself, as an auxiliary operation of G, to develop the theory we call *diffracted groups*: a pseudo-vector decomposition of G in terms of a choice of elements from the cartesian product $T \times H$, in the category Set, where T will be a *transversal* of G of a subgroup H of G. In return the diffracted decomposition $T \times H$ bequeaths an isomorphic block decomposition of G, which makes it a group per se. The decomposition $T \times H$ of G will allow for an explicit description of a rewriting procedure for the product of its ordered pairs.

In the following section we show how the product of a group G decomposes into another auxiliary operation for G, with the basic concept of representations, that will be used in Section 3.5 to describe the rewriting, group-product operation for the decomposition $T \times H$ of G.

3.1 The Cayley Representation

In order to formally accomplish our intention we begin by denoting Set and Grp as the category of sets and groups, respectively, and we set $\mathsf{Sym} : \mathsf{Set} \longrightarrow \mathsf{Grp}$ to be the symmetric functor, which is described by the flow

$$\begin{array}{ccc} X & \xrightarrow{\ \psi\ } & Y \\ \vdots & \overset{\mathsf{Sym}}{\vdots} & \vdots \\ \mathrm{S}_X & \xrightarrow[\psi^{\,\mathsf{Sym}}]{} & \mathrm{S}_Y \end{array}$$

for any objects $X, Y \in \mathsf{Set}$ and any map $\psi : X \longrightarrow Y$, where S_X denotes the symmetric group of X.

Definition 3.1.1. Given a group G, the category $G|\mathsf{Sym}$ of *permutation representations of* G is the category consisting of the following objects and morphisms:

• Objects in $G|\mathsf{Sym}$ are pairs $\langle X, \varphi \rangle$ such that φ is in the exponential $\mathrm{Hom}_{\mathsf{Grp}}(G, \mathrm{S}_X)$ equipped with evaluation

$$v : \mathrm{Hom}_{\mathsf{Grp}}(G, \mathrm{S}_X) \times G \longrightarrow S_X \quad \text{defined by} \quad \forall_{\,g \,\in G} \ \left[\ \langle g, \varphi \rangle \overset{v}{\longmapsto} g^{\varphi}\ \right],$$

where $X \in \mathsf{Set}$, and $\langle g, \varphi \rangle$ are generalized constants from $\langle X, \varphi \rangle$. We refer to objects from $G|\mathsf{Sym}$ as *representations*.

• Morphisms in $G|\mathsf{Sym}$ are arrows $\langle X, \varphi_{_1} \rangle \overset{\psi_*}{\longrightarrow} \langle Y, \varphi_{_2} \rangle$ where ψ_* is a group homomorphism $\psi_* : G^{\varphi_1} \longrightarrow G^{\varphi_2}$ associated to a map ψ from $\mathrm{Hom}_{\mathsf{Set}}(X, Y)$ such that

i. $\quad \forall_{\,g \,\in G}\ \forall_{\,x \,\in X}\ \left[\ \psi_*\big(g^{\varphi_1}\big) : \psi(x) \longmapsto g^{\varphi_2} \circ \psi(x)\ \right]$

ii. $\quad \forall_{\,g \,\in G}\ \forall_{\,x \,\in X}\ \left[\ g^{\varphi_2} \circ \psi(x) = \psi \circ g^{\varphi_1}(x)\ \right]$

i.e., $\psi_* \circ \varphi_{_1} = \varphi_{_2}$ and $\psi \circ g^{\varphi_1} = g^{\varphi_2} \circ \psi$ is satisfied in Grp and Set, respectively.

Any group can be viewed as "operating" on itself by means of permutations through its fundamental operation: by considering the product operation in a group G, we can permute any element h in G simply by multiplying it by another element g in G. In this case, we only have to choose whether to multiply the element g to the right of h or to the left of h. Once we have made this choice we can describe it in the category $G|\mathsf{Sym}$ of permutation representations of G:

Definition 3.1.2. The (left) *regular representation* of a group G is the assignment

$$\varrho : G \longrightarrow \mathrm{S}_G \;\text{ such that }\; \forall_{g,h \in G} \left[\; \varrho : g \longmapsto \left(\, g^{\varrho} : h \longmapsto gh \, \right) \; \right]. \qquad (3.1.1)$$

This apparently simple assignment stores enough information to make ϱ an injective permutation representation of G into the symmetric group S_G.

Definition 3.1.3. Given $G \in \mathsf{Grp}$, the category G-Set of *group actions* of G on Set consists of the following objects and morphisms:

• Objects in G-Set are pairs $\langle \rho, X \rangle$, where $X \in \mathsf{Set}$ and $\rho \in \mathrm{Hom}_{\mathsf{Set}}(G \times X, X)$ satisfies

i. $\quad \forall_{x \in X} \left[\; \rho(1, x) = x \; \right]$

ii. $\quad \forall_{g_1, g_2 \in G} \; \forall_{x \in X} \left[\; \rho(\, g_1 g_2, x \,) \, = \, \rho\big(\, g_1, \rho(g_2, x) \big) \; \right]$

We refer to objects in G-Set as left *actions* of G.

• Morphisms in G-Set are arrows $\psi : \langle \rho_1, X \rangle \longrightarrow \langle \rho_2, Y \rangle$ such that $\psi \in \mathrm{Hom}_{\mathsf{Set}}(X, Y)$ and $\rho_2 \circ (\mathbb{1} \times \psi) \, = \, \psi \circ \rho_1$, i.e., the following diagram commutes:

$$\begin{array}{ccc} G \times X & \xrightarrow{\;\rho_1\;} & X \\ {\scriptstyle \mathbb{1} \times \psi}\downarrow & & \downarrow{\scriptstyle \psi} \\ G \times Y & \xrightarrow{\;\rho_2\;} & Y \end{array}$$

A morphisms ψ is referred as a G-Set *homomorphism*, and ψ is referred as a G-Set *isomorphism* if ψ is also bijective. (By symmetry we can also define a category G-Set whose objects are right actions of G.)

In particular there is an isomorphic adjunction G-$\mathsf{Set} \cong G|\mathsf{Sym}$ making any representation $\varphi : G \longrightarrow S_X$ envision its associated action $\rho : G \curvearrowright X$ via φ, i.e.,

$$\forall_{\, g \,\in\, G} \;\; \left[\;\; \rho : \langle g, x \rangle \longmapsto g x \;\; := \;\; g^{\varphi}(x) \;\; \right].$$

where, by convention, $gx := \rho(g, x)$ indicates ρ acts on the left of X.

The case of Cayley's representation ϱ shows us a glimpse into the decomposition of G. From ϱ we get its associated adjoined $\hat{\varrho} : G \times G \longrightarrow G$, a left action satisfying the commutative diagram

$$\begin{array}{ccc} G \times G & \xrightarrow{\;\bullet\;} & G \\ {\scriptstyle \mathbb{1}_G \,\times\, \mathbb{1}_G}\downarrow & & \uparrow{\scriptstyle \mathbb{1}_G} \\ G \times G & \xrightarrow{\;\hat{\varrho}\;} & G\,, \end{array} \tag{3.1.2}$$

where the symbols "$\bullet$" and $\mathbb{1}_G$ denote the product operation and the identity automorphism of G, respectively. We can think of diagram 3.1.2 as living in the category of sets, since everything involved in its concoction occurs in the underlying set of G. In this scenario, albeit, the bottom left-side part of the diagram $G \times G$ is interpreted as a set, the diagram's commutativity infers that the induced set $\mathrm{S}_{G} \times G$ can be thought as an auxiliary decomposition of the product of G from which the action $\hat{\varrho}$ mimics the product of elements in G.

3.2 The Frobenius Representation

The group G acts on itself via the symmetric group S_G, by way of Cayley's representation, in a regular way—from the left by multiplication—and offers an intuitive way of describing a structural decomposition of G by codifying it with respect to a subgroup H and its cosets.

Definition 3.2.1. Let G be a group, $H \leq G$ and G/H the set of left cosets of of the subgroup H. Given the exponential

$$\Lambda \;:=\; \{\, \tau : G/H \longrightarrow \bigcup_{g \,\in\, G} gH \;\mid\; \forall_{\,g \,\in\, G} \;\; [\, \tau(gH) \in gH \,] \,\}\,, \tag{3.2.1}$$

of G/H with its self, a *set of* (*left*) *representatives* T of G/H is the image of a coordinate $\tau \in \Lambda$, i.e., $T \;:=\; \mathrm{Im}\,\tau$. Moreover a set of representatives T is a (left) *transversal* of G/H whenever the identity element 1 in G is also in T. (If we consider the set commutes of right cosets $H\backslash G$ we can also define a *set of right representatives* R of $H\backslash G$.)

The existence of a set of representatives T emanates from a map $\tau \in \Lambda$, where a "choice" determines a single element from each coset of G/H. Thus, each coordinate point $\tau \in \Lambda$ contains a codification of G, modulo H. Given τ and the canonical projection map $p : G \longrightarrow G/H$, there is a unique map $\bar{\tau} : G \longrightarrow T$ such that the diagram

$$\begin{array}{ccc} G & \overset{\bar{\tau}}{\dashrightarrow} & T \\ & {\scriptstyle p}\searrow \quad \uparrow{\scriptstyle \tau} & \\ & G/H & \end{array} \qquad \text{, i.e.,} \qquad \begin{array}{ccc} g & \overset{\bar{\tau}}{\dashmapsto} & \tau(\,g\,H\,) \\ & {\scriptstyle p}\searrow \quad \uparrow{\scriptstyle \tau} & \\ & g\,H & \end{array}$$

commutes for any $g \in G$. In particular $\bar{\tau} : G \longrightarrow T$ is a systematic map of G onto the representative set T.

Definition 3.2.2. Let $p : G \longrightarrow G/H$ be natural and $\tau \in \Lambda$. Given a set of representatives T of G/N, a *representative map* is the map

$$\overline{\tau} : G \longrightarrow T \quad \text{such that} \quad \forall_{g \in G} \left[\; g \overset{\bar{\tau}}{\longmapsto} \overline{g} := \tau(gH) \; \right].$$

Moreover any element of the form $\overline{g}$ in T is termed a *representative of* g.

Notice that a representative element $\bar{g} \in T$ of g in G depends only on the chosen system of representatives T of G/H.

Lemma 3.2.3. *If T is a set of representatives of G/H of G, then*

i. $\forall_{g \in G} \left[\; gH = \overline{g}H \; \right]$

ii. $\forall_{g_1, g_2 \in G} \left[\; \overline{g_1 \overline{g_2}} = \overline{g_1 g_2} \; \right]$

Proof: Let T be a set of representative of G/H and τ a map of its representatives.

i. Given $g \in G$, then $gH = \bar{g}H$ since its representative $\bar{g}$ satisfies

$$g \in \bar{g}H \iff \exists!_{h \in H} \left[\; \bar{g} = g h^{-1} \; \right] \iff \bar{g} \in gH .$$

ii. By the above argument $g_2 H = \overline{g_2} H$ for any $g_2 \in G$. Then $g_1 g_2 H = g_1 \overline{g_2} H$ for any $g_1 \in G$. Therefore $\overline{g_1 \overline{g_2}} = \overline{g_1 g_2}$ since $\overline{\tau}$ is well defined. □

Definition 3.2.4. Given a set of representatives T of G/H, the *Frobenius representation* is the assignment

$$\gamma : G \longrightarrow \mathrm{S}_T \quad \text{definied as} \quad \forall_{g \in G} \; \forall_{t \in T} \left[\; g \longmapsto (\; g^{\gamma} : t \longmapsto \overline{g\,t} \;) \; \right]. \qquad (3.2.2)$$

Proposition 3.2.5. *The Frobenius representation for a set of representatives T of G/H is a permutation representation.*

Proof: Let T set of representatives of G/H. If $t \in T$, then $\gamma : 1 \longmapsto 1^\gamma$ satisfies $1^\gamma(t) = \overline{1t} = \bar{t} = t$ by definition, i.e., $1^\gamma = 1$ in $\operatorname{Sym} T$. Also, letting $g_1, g_2 \in G$,

$$(g_1\, g_2)^\gamma(t) = \overline{g_1\, g_2\, t} = \overline{g_1\, \overline{g_2\, t}} = \overline{g_1\, g_2^\gamma(t)} = g_1^\gamma \circ g_2^\gamma(t).$$

Therefore $(g_1\, g_2)^\gamma = g_1^\gamma \circ g_2^\gamma$; hence γ is a permutation representation. □

3.3 Diffractions and T-Fibrations

A set of representatives T of G/H is not a subgroup of G, in general; however, with the Frobenius representation we can begin to directly partition G into Set-isomorphic cells of elements from its representative set T and subgroup H.

Proposition 3.3.1. *If T be a set of representatives of G/H and*

$$\nabla : G \longrightarrow T \times H \quad \textit{is defined by} \quad \nabla : g \longmapsto \langle t, h\rangle \tag{3.3.1}$$

for any $g \in G$ such that $g = th$, where $\langle t, h\rangle \in T \times H$, then ∇ is a Set *isomorphism and g can be uniquely decomposed as $g = th$.*

Proof. Given a representative system T of G/H, setwise, G can be rewritten as

$$G = \bigsqcup_{t\,\in\,T} tH \simeq_{\mathsf{Set}} \bigsqcup_{t\,\in\,T} \{t\} \times H = T \times H$$

Therefore for any $g \in G$ there is a unique $t \in T$ such that $g \in tH$. Moreover, if $g = t\,h$ and $t\,h = t\,h'$ for $h, h' \in H$, then $h' = h$, i.e., $\nabla : g \longmapsto \langle t, h\rangle$ is a well-defined bijection. □

Definition 3.3.2. Given a set of representatives T of G/H, the *diffractor* of G by T is the Set-isomorphism $\nabla : G \longrightarrow T \times H$, described in proposition 3.3.1. We refer to its image $T \times H$ as the *diffraction* of G by T, and we refer to an element $\langle t, h\rangle \in T \times H$ as the *spectrum* of g by T.

If we think of the above process metaphorically, then any element g from its source G can be seen with a prism T as having a unique spectrum $\langle t, h\rangle$ from its diffraction $T \times H$. In essence ∇ "diffracts" the group G into the set $T \times H$.

Our goal is to diffract G as the Set product $T \times H$ with an effective procedure for any of its elements. Therefore, in order to do achieve this objective, we consider the actions $G \curvearrowright T$ we have determined from our representations:

From the Frobenius representation $\gamma : G \longrightarrow \mathrm{S}_T$ we get its associated group action

$$\frac{\hat{\gamma} : G \times T \longrightarrow T}{\gamma : G \longrightarrow \mathrm{S}_T} \quad \text{described as} \quad \forall_{\langle g,t\rangle \in G\times T} \left[\ \hat{\gamma} : \langle g, t\rangle \longmapsto g^{\gamma}(t) = \overline{gt} \ \right]$$

In particular the set of representatives T is invariant under $G \curvearrowright T$ modulo γ, by definition. Now consider $G \curvearrowright T$ modulo cayley's representation ϱ, and rewrite its action as

$$\hat{\varrho} \ : \ \langle g, t\rangle \longmapsto \overline{gt}\,(\,\overline{gt}\,)^{-1} gt \ := \ gt \tag{3.3.2}$$

given any of $g \in G$ and $t \in T$. On the right side of equation (3.3.2) we have rewritten the effect of Cayley's action over $\langle g, t\rangle$ using Frobenius representation γ. Notice $\overline{gt} \in T$ by definition, and $(\,\overline{gt}\,)^{-1} gt \in H$, because for any $t \in T$ and $g \in G$

$$\overline{1} = \overline{(\,\overline{g\,t}\,)^{-1}(\,\overline{g\,t}\,)} = \overline{(\,\overline{g\,t}\,)^{-1} g\,t} \quad \Longrightarrow \quad H = \overline{1}\,H = \overline{(\,\overline{g\,t}\,)^{-1} g\,t}\,H = (\,\overline{gt}\,)^{-1} gt\,H\,,$$

which follows from lemma 3.2.3. And, by proposition 3.3.1, Cayley's action of g on t can be uniquely express as the product of $\overline{gt} \in T$ and $(\,\overline{gt}\,)^{-1} gt \in H$. In particular $gt \notin T$ if and only if $gt \neq \overline{gt}$, which verifies the obvious fact that T is not invariant under cayley's action. But, most important, equation (3.3.2) allows also us to rewrite the effect of the diffractor with Cayley's action restricted to the set T as follows:

$$\nabla \circ \hat{\varrho}\,\restriction_{T} : G \times T \longrightarrow T \times H \quad \text{such that} \quad \forall_{\langle g,t\rangle \in G\times T} \left[\ \nabla \circ \hat{\varrho}\,(g, t) = \langle\, \overline{gt},\, (\,\overline{gt}\,)^{-1} gt \,\rangle \ \right].$$

Definition 3.3.3. A *T-fibration* of G, for a representative T of G/H, is the map

$$\delta : G \times T \longrightarrow H \quad \text{defined by} \quad \forall_{\langle g,t\rangle \in G\times T}\left[\ \delta(g,t) := (\overline{g\,t}\,)^{-1} g\,t\ \right].$$

A T-fibration δ is well defined by construction, and it allows for an effective description of the diffraction $\nabla(G) = T \times H$, as follows:

Theorem 3.3.4. *If T is a transversal of G/H and $\delta : G \times T \longrightarrow H$ its T-fibration, then the diffractor $\nabla : G \longrightarrow T \times H$ is uniquely decomposable into the pair*

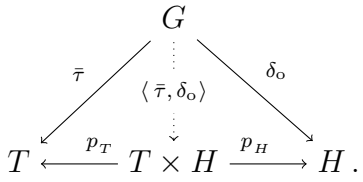
$$\nabla \;=\; \langle\, \bar{\tau}\,,\, \delta_{\mathrm{o}}\,\rangle \quad \textit{such that} \quad \forall_{\,g\,\in G}\left[\ \langle\, \bar{\tau}\,,\, \delta_{\mathrm{o}}\,\rangle\,(g) \;=\; \langle\, \bar{g}\,,(\,\overline{g}\,)^{-1} g\,\rangle\ \right],$$

where δ_{o} is $\delta(-,1) : G \longrightarrow H$ and $\bar{\tau}$ is the representative map (cf. definition 3.2.2).

Proof: Let $\delta : G \times T \longrightarrow H$ be the T-fibration of G/H and let $\nabla : G \longrightarrow T \times H$ be the diffractor of G by T. If T is a transversal, $1 \in T$ and Cayley's action of any $g \in G$ on the identity is just $\hat{\varrho}(g,1) = g$. Then

$$\nabla(g) \;=\; \nabla \circ \hat{\varrho}\,(g,1) \;=\; \langle \bar{g}, (\,\overline{g}\,)^{-1} g\rangle \;=\; \langle\, \bar{\tau}(g), \delta(g,1)\,\rangle \;=\; \langle \bar{\tau}, \delta_{\mathrm{o}}\rangle\,(g)\,,$$

where δ_{o} is $\delta(-,1) : G \longrightarrow H$. Now given the projection maps $p_{_H} : T \times H \longrightarrow H$ and $p_{_T} : T \times H \longrightarrow T$, the pair $\langle\, \bar{\tau}, \delta_{\mathrm{o}}\rangle$ satisfies the commutative diagram



Therefore ∇ is uniquely decomposable into the pair $\langle\, \bar{\tau}, \delta_{\mathrm{o}}\rangle$ by uniqueness of the universal property of direct products. □

3.4 The Diffracted Representation

To explicitly construct a group product in the set $T \times H$, we mimic the auxiliary product determined by the Cayley representation of G into S_G, cf. diagram (3.1.2). In this section we gather tools from the previous section to describe an isomorphic representation of Cayley's representation of G into $\mathrm{S}_{T\times H}$, the symmetric group of $T \times H$; then, in the following section, this representation will enable us to describe the product of G as a product in its canonical set $T \times H$ for a transversal T of G/H from the standard set-theoretic isomorphism between the elements of G and $T \times H$. We do this by first examining the set of maps $\mathrm{Hom}_{\mathsf{Set}}(T, H)$.

To be more precisely, given a concrete category C and any two objects $A, B \in \mathsf{C}$, recall the exponential of A by B is the set $B^A := \mathrm{Hom}_{\mathsf{C}}(A, B)$, which denotes the collection of morphisms from A to B. For a set of representatives T of G/H, the exponential explicitly takes the form

$$H^T = \{\, f : T \longrightarrow H \,\}$$

and is a set that becomes a group under point-wise multiplication, e.g.,

$$\forall_{f_1, f_2 \in H^T} \left[\, f_1 \cdot f_2 \;:=\; \{\, \langle\, t\,, f_1(t)\, f_2(t) \;\rangle \;\mid t \in T \,\} \,\right]. \tag{3.4.1}$$

Now the natural isomorphism $\mathrm{Hom}_{\mathsf{set}}(G \times T, H) \cong \mathrm{Hom}_{\mathsf{set}}(G, H^T)$ describes the dual $\tilde{\delta} : G \longrightarrow H^T$ of a T-fibration $\delta : G \times T \longrightarrow H$ as follows:

$$\frac{\tilde{\delta} : G \longrightarrow H^T}{\delta : G \times T \longrightarrow H} \quad \text{such that} \quad \forall_{g \in G}\ \forall_{t \in T} \left[\, \tilde{\delta} : g \longmapsto \big(\, \delta_g :\ t \longmapsto \delta(g, t) \,\big) \,\right],$$

where δ_g denotes the map $\delta(g, -) : T \longrightarrow H$.

Lemma 3.4.1. *A T-fibration δ, for a set of representatives T of G/H, satisfies*

$$\forall_{t \in T}\ \forall_{g_1, g_2 \in G} \left[\; \delta(g_1\, g_2\,, t) \;=\; \delta\big(g_1\,,\, g_2^{\gamma}\,(t)\big)\, \delta(g_2, t) \;\right]$$

Proof: If $\delta : G \times T \longrightarrow H$ is the T-homotopy associated to the representative, then

$$\delta\big(g_1\,,\, g_2^{\gamma}\,(t)\big)\,\delta(g_2,t) \;=\; (\,\overline{g_1\,\overline{g_2\,t}}\,)^{\text{-}1}\,g_1\,\overline{g_2\,t}\;\;(\,\overline{g_2\,t}\,)^{\text{-}1}\,g_2\,t \;=\; (\,\overline{g_1\,g_2\,t}\,)^{\text{-}1}\,g_1\,g_2\,t \;=\; \delta(g_1\,g_2\,,t)$$

for any $g_1, g_2 \in G$ since $\overline{gt} := g^{\gamma}(t)$. □

Observe that the dual map $\tilde{\delta} : G \longrightarrow H^T$ of the T-fibration is in Set and it does not extend to a homomorphism, by lemma 3.4.1 above, since

$$\delta_{g_1\,g_2}(t) \;:=\; \delta(g_1\,g_2\,,t) \;\neq\; \delta(\,g_1,t\,)\,\delta(\,g_2,t\,) \;=:\; \delta_{\,g_1}\,(t)\;\delta_{\,g_2}\,(t)\,.$$

However, letting $\tilde{\delta}(G) := \{\,\delta_g \mid g \in G\,\}$ denote the image of dual $\tilde{\delta} : G \longrightarrow H^T$, the set $\tilde{\delta}(G)$ becomes a group.

Lemma 3.4.2. *Given the dual $\tilde{\delta}$ of a T-fibration δ, its image $\tilde{\delta}(G)$ becomes a group with identity element δ_1 under point-wise multiplication .*

Proof. The set $\tilde{\delta}(G) = \{\,\delta_g \mid g \in G\,\}$ is clearly a group under formulation (3.4.1). Moreover, given the identity $1 \in G$, the identity of $\tilde{\delta}(G)$ is δ_1 since

$$\delta_{_1}(t) \;=\; \delta(1,t) \;=\; 1 \;\;\Longrightarrow\;\; \big(\delta_1 \cdot \delta_g\big)\,(t) \;=\; \delta(1,t)\,\delta(g,t) \;=\; \delta_g(t)$$

for any $\delta_g \in \tilde{\delta}(G)$ and $t \in T$. □

The group structure of the image $\tilde{\delta}(G)$ of the dual of δ acts on the diffraction $T \times H$, i.e., any element $f \in H^T$ can act as a permutation on $T \times H$ as

$$\forall_{\,\langle t,h\rangle\,\in T\times H}\;\;\Big[\;f : \langle t,h\rangle \longmapsto \langle\, t,\, f(t)\,h\,\rangle\;\Big]\,. \tag{3.4.2}$$

Lemma 3.4.3. *Given the group structure $\tilde{\delta}(G)$ of the image of the dual $\tilde{\delta}$ of a T-fibration δ, the assignment $\beta : \tilde{\delta}(G) \longrightarrow \mathrm{S}_{\,T\times H}$ defined as*

$$\forall_{\,\delta(g)\,\in\,\tilde{\delta}(G)}\;\;\forall_{\,\langle t,h\rangle\,\in\,T\times H}\;\Big[\;\;\delta_{\,g}\;\overset{\beta}{\longmapsto}\;\Big(\,(\delta_g)^{\beta}\;:\langle t,h\rangle \longmapsto \langle t,\,\delta_g(t)\,h\,\rangle\Big)\;\;\Big] \tag{3.4.3}$$

is an injective permutation representation of $\tilde{\delta}(G)$ into $\mathrm{S}_{\,T\times H}$.

Proof. The assignment $\beta : \tilde{\delta}(G) \longrightarrow \mathrm{S}_{T\times H}$ is a well-defined map by way of δ. Now *a priori* $(\delta_1)^{\beta}(t,h) = \langle t, \delta_1(t)\, h\rangle = \langle t, h\rangle$ for any $\langle t,h\rangle \in T\times H$. Then β is a homomorphism since

$$\left(\delta_{g_1}\cdot \delta_{g_2}\right)^{\beta}(t,h) = \langle t, \delta_{g_1}(t)\delta_{g_2}(t)\, h\rangle = \left(\delta_{g_1}\right)^{\beta}(\, t, \delta_{g_2}(t)\, h\,) = \left(\delta_{g_1}\right)^{\beta}\circ \left(\delta_{g_2}\right)^{\beta}(t,h)$$

for any $g_1, g_2 \in G$. To see that β is injective let $\delta_g \in \operatorname{Ker}\beta$. Then

$$(\delta_g)^{\beta}(t,h) = \langle t,h\rangle = \langle t, \delta_g(t) h\rangle\,. \tag{3.4.4}$$

Equation (3.4.4) above is satisfiable if and only if $\delta_g(t) = 1$, i.e., $g = 1$. Therefore β is an injective permutation representation of $\tilde{\delta}(G)$ into $\mathrm{S}_{T\times H}$. □

From lemma 3.4.3 it follows that the group structure of $\tilde{\delta}(G)$, the dual $\tilde{\delta}$ of a T-fibration δ, can act on the diffraction $T\times H$. This result, along with the assistance of the Frobenius representation and the dual $\tilde{\delta}$ of δ, grants us an extension of the Frobenious representation to a permutation representation of G into $\mathrm{S}_{T\times H}$.

Definition 3.4.4. The *diffracted representation* of G by a set of representatives T of G/H is the assignment $\alpha : G \longrightarrow \mathrm{S}_{T\times H}$ defined by

$$\forall_{\, g\,\in G}\ \forall_{\,\langle t,h\rangle\,\in T\times H}\left[\ g \overset{\alpha}{\longmapsto} \Big((g^{\gamma}\times \mathbb{1}_H)\circ(\delta_g)^{\beta} : \langle t,h\rangle \longmapsto \langle\, g^{\gamma}(t)\,,\, \delta_g(t)\, h\,\rangle\Big)\ \right],$$

where β is the permutation representation described in the previous lemma 3.4.3.

Theorem 3.4.5. *The diffracted representation $\alpha : G \longrightarrow \mathrm{S}_{T\times H}$ is an injective permutation representation of G into $\mathrm{S}_{T\times H}$.*

Proof: Let $g\in G$ and $\langle t,h\rangle \in T\times H$. Then $(g^{\gamma}\times \mathbb{1}_H) \in \mathrm{S}_{T\times H}$ since it acts as

$$\langle t,h\rangle \overset{g^{\gamma}\times \mathbb{1}_H}{\longmapsto} \langle g^{\gamma}(t), h\rangle\,,$$

which is permutation induced by the Frobenius representation γ. From lemma 3.4.3 $\delta_g \in \tilde{\delta}(G)$ acts on the diffraction $T \times H$ as $(\delta_g)^\beta : \langle t, h\rangle \longmapsto \langle\, t,\, \delta(t,g)\, h\rangle$. Thus $(g^\gamma \times \mathbb{1}_H) \circ (\delta_g)^\beta \in \mathrm{S}_{T\times H}$ is a permutation on the set $T \times H$ described as

$$(g^\gamma \times \mathbb{1}_H) \circ (\delta_g)^\beta : \langle t, h\rangle \longmapsto \langle\, g^\gamma(t)\,,\, \delta_g(t)\, h\,\rangle\,.$$

Now the action of the identity $1 \in G$ induced by α on $\langle h, t\rangle$ is given as

$$\alpha(1)\,\langle t, h\rangle \;:=\; (1^\gamma \times \mathbb{1}_H) \circ (\delta_g)^\beta\,\langle t, h\rangle \;=\; \langle\, 1^\gamma(t)\,,\, \delta_1(t)\, h\,\rangle \;=\; \langle\, \bar{t}\,,\, (\,\bar{t}\,)^{-1} t\, h\,\rangle \;=\; \langle t, h\rangle$$

since $\bar{t} = t$. Therefore $\alpha(1) \;=\; 1$ in $\mathrm{S}_{T\times H}$. Moreover, given $g_1, g_2 \in G$, the action of g_2 followed by the action of g_1 on $\langle t, h\rangle$ by way of α is

$$\begin{aligned}
\alpha(g_1) \circ \alpha(g_2)\,\langle t, h\rangle \;&=\; \big((g_1^\gamma \times \mathbb{1}_H) \circ (\delta_{g_1})^\beta\big) \circ \big((g_2^\gamma \times \mathbb{1}_H) \circ (\delta_{g_2})^\beta\big)\,\langle t, h\rangle \\
&=\; \big((g_1^\gamma \times \mathbb{1}_H) \circ \delta_g\,\big)\,\langle\, g_1^\gamma(t)\,,\, \delta(g_2, t)\, h\,\rangle \\
&=\; \langle\, g_1^\gamma \circ g_2^\gamma\,(t)\,,\, \delta(g_1\,,\, g_2^\gamma\,(t)\,)\,\delta(g_2, t)\, h\,\rangle\,.
\end{aligned}$$

But $g_1^\gamma \circ g_2^\gamma\,(t) \;=\; (\,g_1\, g_2\,)^\gamma\,(t)$ and $\delta\big(g_1\,,\, g_2^\gamma\,(t)\big)\,\delta(g_2, t) \;=\; \delta(g_1\, g_2\,, t) \;:=\; \delta_{\,g_1 g_2}\,(t)\,,$ which follows by the Frobenius representation γ and lemma 3.4.1, respectively. Therefore, putting these two facts together, we get

$$\begin{aligned}
\alpha(g_1) \circ \alpha(g_2)\,\langle t, h\rangle \;&=\; \langle\, (g_1\, g_2)^\gamma\,(t)\,,\, \delta_{\,g_1\, g_2}\,(t)\; h\,\rangle \;=\; \big((g_1\, g_2)^\gamma \times \mathbb{1}_H\big) \circ (\delta_{\,g_1\, g_2})^\beta\,\langle t, h\rangle \\
&=\; \alpha(\,g_1\, g_2\,)\,\langle t, h\rangle\,,
\end{aligned}$$

i.e., $\alpha(\,g_1\, g_2\,) \;=\; \alpha(g_1) \circ \alpha(g_2)$. Hence α is a representation of G into $\mathrm{S}_{\,T\times H}$. Last, α is injective if $\langle\, g^\gamma(t)\,,\, \delta_g(t)\, h\,\rangle \;=\; \langle t, h\rangle$, i.e., $\overline{gt} = t$ and $\delta_{\,g}(t) = 1$ if and only if $g = 1$. Thus α is injective as well. $\square$

3.5 The Diffracted Group

In this section we use the bijection of the diffractor $\nabla : G \longrightarrow T \times H$ and the difracted representation $\alpha : G \longrightarrow \mathrm{S}_{T\times H}$ to the describe the group structure for the difracted set of G, i.e, the cartesian set $T \times H$, for a transversal set T of G/H. Moreover, given the group structure of $T \times H$, we also show that ∇ is a group isomorphism as well. We do this by applying the following G-Set isomorphism:

Theorem 3.5.1. *The diffractor $\nabla : G \longrightarrow T \times H$ for a transversal T of G/H is a G-Set isomorphism between the actions induced by Cayley's representation $\varrho : G \longrightarrow T$ and the diffracted representation $\alpha : G \longrightarrow T \times H$.*

Proof: Let $\hat{\varrho} : G \times T \longrightarrow T$ and $\hat{\alpha} : G \times (T \times H) \longrightarrow T \times H$ be the action induce by Cayley's representation ϱ of G and the diffracted representation α of G by T, respectively. Then our bijection $\nabla : G \longrightarrow T \times H$ satisfies the commutative diagram

$$\begin{array}{ccc} G \times G & \xrightarrow{\hat{\varrho}} & G \\ {\scriptstyle \mathbb{1}_G \times \nabla}\downarrow & & \downarrow{\scriptstyle \nabla} \\ G \times (T \times H) & \xrightarrow{\hat{\alpha}} & (T \times H) \end{array} \qquad \text{since} \qquad \begin{array}{ccc} \langle g,\, k \rangle & \xmapsto{\hat{\varrho}} & \overline{gt}\,(\overline{g\,t})^{-1}\, g\, t\, h \\ {\scriptstyle \mathbb{1}_G \times \nabla}\downarrow & & \downarrow{\scriptstyle \nabla} \\ \langle g\,, \langle t, h\rangle \rangle & \xmapsto{\hat{\alpha}} & \langle \overline{gt}\,,\, (\overline{g\,t})^{-1} g\, t\, h\rangle \end{array}$$

for any $g, k \in G$ where $k = t\, h$ and $\langle t, h\rangle \in T \times H$, i.e., ∇ is a G-Set isomorphism. □

Theorem 3.5, above, not only stipulates that G and $T \times H$ are G-Set isomorphic, as sets, but that the diffracted representation α expresses the Cayley representation when G is diffracted as $T \times H$ as a set. Fortunately, this relationship is more than happenstance: the diffraction $T \times H$ of G borrows the group structure of G through α, since Cayley's representation manifests the product structure of G. As consequence,

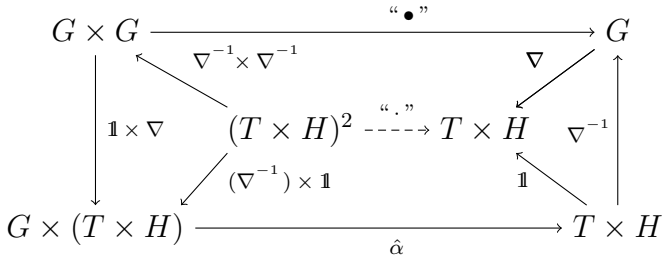
similar to the commutativity of diagram (3.1.2), the product "$\bullet$" in G splits as

$$\begin{array}{ccc} G \times G & \xrightarrow{\text{“}\bullet\text{”}} & G \\ {\scriptstyle \mathbb{1}\times\nabla}\downarrow & & \uparrow{\scriptstyle \nabla^{-1}} \\ G \times (T \times H) & \xrightarrow{\hat{\alpha}} & T \times H \end{array} \tag{3.5.1}$$

Theorem 3.5.2. *If δ is the T-fibration associated to Frobenius representation γ for a transversal T of G/H, then $T \times H$ becomes a group under the operation defined as*

$$\forall_{\,\langle t_1,h_1\rangle,\,\langle t_2,h_2\rangle\,\in T\times H}\ \Big[\ \langle t_1,h_1\rangle\cdot\langle t_2,h_2\rangle\ :=\ \langle\, t_1^{\gamma}\circ h_1^{\gamma}\,(t_2)\,,\,\delta(t_1h_1,t_2)\,h_2\,\rangle\ \Big] \tag{3.5.2}$$

Proof. Given the diffractor $\nabla : G \longrightarrow T \times H$ of G with respect to a transversal T and the diffracted representation $\alpha : G \longrightarrow T \times H$ of G by T, by diagram (3.5.1) the following diagram commutes as well:



Moreover, the central dashed arrow describes the product in $T \times H$ induced by G as

$$\Big\langle\, \langle t_1,h_1\rangle, \langle t_2,h_2\rangle \,\Big\rangle \ \overset{\cdot}{\longmapsto}\ \langle\, t_1^{\gamma}\circ h_1^{\gamma}\,(t_2)\,,\,\delta(t_1h_1,t_2)\,h_2\,\rangle$$

for any $\langle t_1,h_1\rangle$, $\langle t_2,h_2\rangle \in T \times H$. Therefore, since G is a group, $T \times H$ is a group. □

This theorem also allows the Frobenius representation γ and its T-fibration δ to interpret the product of any two elements $g_1, g_2 \in G$ of the form $g_1 = t_1h_1$ and $g_2 = t_2h_2$, for $\langle t_1,h_1\rangle, \langle t_2,h_2\rangle \in T \times H$, by the following "normal" form:

$$g_1g_2 = t_1^{\gamma}\circ h_1^{\gamma}\,(t_2)\,\delta(t_1h_1,t_2)\,h_2 = \overline{t_1h_1t_2}\,\delta(t_1h_1,t_2)\,h_2\,,$$

where $\overline{t_1 h_1 t_2} \in T$ and $\delta(t_1 h_1, t_2)\, h_2 \in H$. Hence, allowing the product of elements in a group G to be rewritten in terms of a transversal T and its associated subgroup H.

Definition 3.5.3. Given a transversal T of G/H for a group G, the (internal) *diffracted group of* G by T, denoted by $T \cdot H$, is the group structure on $T \times H$ inherited from formulation (3.5.2).

Theorem 3.5.4. *The diffractor* $\nabla : G \longrightarrow T \cdot H$ *of a transversal* T *of* G/H *is a group isomorphism.*

Proof: Given the identity $1 \in G$, then $\nabla(1) = \langle 1, 1 \rangle$. Now let g_1, g_2 be elements in G such that each has a unique decomposition $g_1 = t_1\, h_1$ and $g_2 = t_2\, h_2$, where $t_1, t_2 \in T$ and $h_1, h_2 \in H$. Then ∇ is an epimorphism since

$$\begin{aligned}
\nabla(g_1\, g_2) &= \langle\, \overline{t_1 h_1\, t_2 h_2}\,, \delta(t_1 h_1\, t_2 h_2, 1)\,\rangle \;\; = \langle\, \overline{t_1 h_1\, t_2 \overline{h_2}}\,,\, \delta(t_1 h_1,\, \overline{t_2 \overline{h_2}})\, \delta(t_2 h_2, 1)\,\rangle \\
&= \langle\, t_1^{\gamma} \circ h_1^{\gamma}(t_2)\,, \delta(t_1 h_1,\, t_2)\, h_2\,\rangle \\
&= \langle t_1, h_1 \rangle \cdot \langle t_2, h_2 \rangle \\
&= \nabla(g_1) \cdot \nabla(g_2)\,.
\end{aligned}$$

Moreover ∇ is an isomorphism since any $g \in \operatorname{Ker} \nabla$ must satisfy $g = 1$. □

Chapter 4

Using Transversals in Public-key Cryptography

In this last Chapter we present a key-exchange protocol based on the idea of Kahrobaei and Shpilrain [12]. In their description, Kahrobaei and Shpilrain implement the semidirect product $G = K \ltimes H$, where K is a subgroup of G and H is normal in G. In our description we replace $K \ltimes H$ with a set $T \times H$, where T is a transversal of a group G of a subgroup H of G, which we described in the Chapter 3. Despite differences between our protocols, the two systems agree on a main advantage factor: both do not transmit all of the information but rather part of it.

Our protocol also utilizes parts of the standard Diffie-Hellman cryptographic protocol, similar to [12]. Ordinarily the Diffie-Hellman protocol uses the cyclic group C_n, the multiplicative group of integers modulo some natural number n, as follows:

1. Alice and Bob agree on a cyclic group C_n and a generating element g in C_n.

2. Alice picks a random natural number a and sends g^a to Bob.

3. Bob picks a random natural number b and sends g^b to Alice.

4. Alice and Bob compute $K_A = (g^b)^a = g^{ba}$ and $K_B = (g^a)^b = g^{ab}$, respectively.

Here Alice and Bob end up with the same element $K = K_A = K_B$ since $ab = ba$, which stands as the shared secret key. Notice that an eavesdropper Eve has to solve the *Diffie-Hellman problem* to recover the shared secret key, i.e., Eve must recover g^{ab} from g, g^a and g^b to obtain K. This is currently considered a hard problem for an "ideal" choice of parameters (cf. [32] for details).

The Diffie-Hellman key exchange protocol initiated the modern field of public-key cryptography and asymmetric encryption methods in the seminal paper [13]. At the present time the Diffie-Hellman protocol—along with other key-exchange algorithms and asymmetric encryption—has become centrally ingrained in the cryptographic community because of its simplicity, clarity, and practical security that alludes its elegance, as a fundamental tool for exchanging sensitive data over the electronic channels. However, with the rise of post-quantum cryptography, algorithmic problems such as the Diffie-Hellman problem, can potentially be solved with an adequate quantum computer.

As we mentioned in the intructory chapter, quantum computers with approximately 512 qubits are claimed to be available now to companies like Google, Lockheed Martin and government agencies, see e.g. [47]. For these reasons there is a strong interest in designing a *Post-Quantum Key Exchange* protocol that could be resilient against quantum computers. One particular road paving way towards this ambition focuses on the search for platforms where the security of the Diffie-Hellman protocol would have different premises, or where a similar algorithm could be implemented with greater efficiently.

In search for these new cryptographic primitives, post-quantum cryptography has given rise to a number of alternatives. These include the development of finite field

based cryptography; elliptic curve based cryptography; lattice based cryptography; and, in our context, the nascent development of primitives based on non-commutative groups (see e.g. [49] and [35],) which has been the subject of this thesis.

4.1 Mimicking the Kahrobaei-Shpilrain Protocol

The Kahrobaei and Shpilrain protocols [12] and [22] rely on the idea of semidirect products, or more generally, on extension of (semi)groups by endomorphisms.

Definition 4.1.1. Let G and H be groups, and let $\rho : H \longrightarrow \operatorname{Aut} G$ be a homomorphism from H into the automorphism group of G. Then the semidirect product of G by H is the set $\Gamma = G \rtimes_\rho H = \{\langle g, h\rangle : g \in G,\ h \in H\}$ with the product given by

$$\forall_{n \in \mathbb{N}} \left[\langle g, h\rangle\langle g', h'\rangle = \langle g^{\rho(h')} \cdot g',\ h \cdot h'\rangle \right],$$

where $g^{\rho(h')}$ denotes the image of g under the automorphism $\rho(h')$.

To explain our protocol we first describe the Kahrobaei-Shpilrain protocol, which resembles the classical Diffie-Hellman procedure; however, there are several distinctive features that give their new protocol a theoretical advantage that can potentially serve in practical implementations. For example, as we alluded in the introduction of this chapter, even though Alice and Bob compute a large power of a public element of the group, they transmit only part of the element. More specifically, verbatim, their protocol is described as follows:

A (semi)group G and an element $g \in G$ is chosen and made public, as well as an arbitrary endomorphism $\phi \in \operatorname{End} G$. Both Alice and Bob are going to compute elements of the form $\langle g, \phi^r\rangle$, where $\langle g, \phi^r\rangle \cdot \langle h, \phi^s\rangle = \langle \phi^s(g) \cdot h,\ \phi^{r+s}\rangle$ describes the product for $g, h \in G$ and $r, s \in \mathbb{N}$. Then,

1. Alice computes $\langle g, \phi \rangle^m = \langle \phi^{m-1}(g) \cdots \phi^2(g) \cdot \phi(g) \cdot g,\ \phi^m \rangle$ for an $m \in \mathbb{N}$ she privately chooses. She then sends **only the first component** to Bob. Thus, she sends Bob **only** $a = \phi^{m-1}(g) \cdots \phi^2(g) \cdot \phi(g) \cdot g$ of the (semi)group G.

2. Bob computes $\langle g, \phi \rangle^n = \langle \phi^{n-1}(g) \cdots \phi^2(g) \cdot \phi(g) \cdot g,\ \phi^n \rangle$ for an $n \in \mathbb{N}$ he privately chooses. He then sends **only the first component** to Alice. Thus, he sends Alice **only** $b = \phi^{n-1}(g) \cdots \phi^2(g) \cdot \phi(g) \cdot g$ of the (semi)group G.

3. Alice computes $\langle b, x \rangle \cdot \langle a,\ \phi^m \rangle = \langle \phi^m(b) \cdot a,\ x \cdot \phi^m \rangle$. Her key is now of the form $K_A = \phi^m(b) \cdot a$. Note that she does not actually "compute" $x \cdot \phi^m$ because she does not know the automorphism $x = \phi^n$ (since it is not transmitted to her, and she does not need it to compute K_A.)

4. Bob computes $\langle a, y \rangle \cdot \langle b,\ \phi^n \rangle = \langle \phi^n(a) \cdot b,\ y \cdot \phi^n \rangle$. His key is now $K_B = \phi^n(a) \cdot b$. Similar to Alice, Bob does not actually "compute" $y \cdot \phi^n$ because he does not know the automorphism $y = \phi^m$.

5. Since $\langle b, x \rangle \cdot \langle a,\ \phi^m \rangle = \langle a,\ y \rangle \cdot \langle b,\ \phi^n \rangle = \langle g,\ \phi \rangle^{m+n}$, we have the shared secret key $K := K_A = K_B$.

The protocol we will describe in the following section is similar but more general: the platform group G does not need to be a semidirect product, but it can be any reliable group platform with a subgroup H, which itself does not have to be normal in G. To be more precise, our procedure relies on decomposition of a group G into a set $T \times H$, where T is a transversal from the set of cosets G/H for a subgroup $H < G$. The set $T \times H$ can be made into a group $T \cdot H$ by defining the product as

$$\langle t_1, h_1 \rangle \cdot \langle t_2, h_2 \rangle := \langle \overline{t_1 h_1 t_2}\,,\ \delta(t_1 h_1, t_2)\, h_2 \rangle \tag{4.1.1}$$

for any $\langle t_1, h_1\rangle$, $\langle t_2, h_2\rangle \in T \times H$, where $\overline{k}$ denotes the representative of $k \in G$ in G/H and $\delta(g,t) := (\overline{g\,t})^{-1} g\,t$ (see theorem 3.5.2, Chapter 3, for the derivation of this equation.) In other words, the product of any two elements in G of the form $g_1 = t_1 h_1$ and $g_2 = t_2 h_2$, may be expressed by the normal form

$$g_1\, g_2 = \overline{t_1 h_1 t_2}\, \delta(t_1 h_1, t_2)\, h_2$$

from the set $\{\, th \mid t \in T, h \in H \,\}$, where $\overline{t_1 h_1 t_2} \in T$ and $\delta(t_1 h_1, t_2)\, h_2 \in H$. Now, in order to describe our protocol, we only have to determine the expression of the power of an element $\langle t, h\rangle$ from the group $T \cdot H$.

Theorem 4.1.2. *Let T be a transversal of G/H and $g \in G$ be of the from $g = th$, where $t \in T$ and $h \in H$. Then the expression of g as $\langle t, h\rangle$ in $T \cdot H$ satisfies*

$$\forall_{\,n\,\in\mathbb{N}} \left[\; \langle t, h\rangle^{\,n+1} \;=\; \langle\, \overline{g^n t}\,,\, \delta(g^n t, t)\, h \,\rangle \;\right] \tag{4.1.2}$$

Proof: Let $g \in G$ be of the from $g = th$ such that g is expressible as $\langle t, h\rangle$ in TH, where $t \in T$ and $h \in H$. First, suppose $n = 1$ in (4.1.2). Then

$$\langle t, h\rangle^{\,1+1} \;=\; \langle t, h\rangle \cdot \langle t, h\rangle \;=\; \langle\, \overline{tht}\,,\, \delta(tht, t)\, h \,\rangle \;=\; \langle\, \overline{g^1 t}\,,\, \delta(g^1 t, t)\, h \,\rangle$$

since $g = th$. Next, suppose equation (4.1.2) is satisfiable up to $n \in \mathbb{N}$. Then,

$$\begin{aligned} \langle t, h\rangle^{\,n+1} &= \langle t, h\rangle \cdot \langle t, h\rangle^{\,n} = \langle t, h\rangle \cdot \langle\, \overline{g^{(n-1)} t}\,,\, \delta(g^{(n-1)} t, t)\, h \,\rangle \;-\; \textit{a priori}\ \text{induction;} \\ &= \langle\, \overline{th\overline{g^{(n-1)}\, t}}\,,\, \delta(th, \overline{g^{\,(n-1)} t})\; \delta(g^{\,(n-1)}\, t, t)\, h \,\rangle\,. \end{aligned} \tag{4.1.3}$$

Now $\overline{th\overline{g^{(n-1)}\, t}} \;=\; \overline{th g^{(n-1)}\, t}$ and $\delta(th, \overline{g^{(n-1)}\, t})\, \delta(g^{(n-1)}\, t, t) \;=\; \delta(th g^{(n-1)}\, t, t)$ by lemma 3.2.3 and lemma 3.4.1. Then equation (4.1.3) becomes

$$\langle t, h\rangle^{\,n+1} \;=\; \langle\, \overline{g g^{(n-1)}\, t}\,, \delta(g g^{(n-1)}\, t, t) h \,\rangle \;=\; \langle\, \overline{g^n\, t}\,, \delta(g^n t, t) h \,\rangle\,,$$

again, since $g = th$. □

4.2 A Transversal Key-Exchange Protocol

Let T be a transversal of G/H and $g \in G$ of the form $\langle t, h\rangle \in T \times H$. Both Alice and Bob are going to work with elements of the form

$$\langle t, h\rangle^{r+1} \;=\; \langle\, \overline{g^r t}\,,\, \delta(g^r t, t)\, h\,\rangle \tag{4.2.1}$$

for a power $r \in \mathbb{N}$, where Bob and Alice will choose a private $n \in \mathbb{N}$ and $m \in \mathbb{N}$ for r, respectively (cf. theorem 4.1.2.)

Protocol Assumptions: Let G be a group and H a subgroup of G.

1. A transversal set T of G/H is chosen, and it is made public along with an arbitrary element $g \in G$ of the form $\langle t, h\rangle \in T \times H$.
2. Alice chooses a private $n \in \mathbb{N}$ and Bob chooses a private $m \in \mathbb{N}$.

Key-Exchange Protocol: Alice and Bob generate a key from $g \in G$ as follows.

1. Alice computes $\langle t, h\rangle^{n+1} \;=\; \langle\, \overline{g^n t}, \delta(g^n t, t) h\,\rangle$ and sends **only the first component** to Bob, i.e., she sends **only** the element $a := \overline{g^n t}$ in T.
2. Bob computes $\langle t, h\rangle^{m+1} \;=\; \langle\, \overline{g^m t}, \delta(g^m t, t) h\,\rangle$ and sends **only the first component** to Alice, i.e., he sends **only** the element $b := \overline{g^m t}$ in T.
3. Alice computes $\overline{g^n b} = \overline{g^n \overline{g^m t}} = \overline{g^n g^m t} = \overline{g^{n+m} t}$. Her key is now set to be $K_A = \overline{g^{n+m} t}$.
4. Bob computes $\overline{g^m a} = \overline{g^m \overline{g^n t}} = \overline{g^m g^n t} = \overline{g^{m+n} t}$. His key is now set to be $K_B = \overline{g^{m+n} t}$.

Now since $\overline{g^{n+m}t} = \overline{g^{m+n}t}$, Alice and Bob share $K := K_A = K_B$ as their secret key.

Remark. Notice that both Alice and Bob share the first component of

$$\langle t, h \rangle^{(n+m)+1} \; = \; \langle \, \overline{g^{n+m}t}, \delta(g^{n+m}t, t)h \, \rangle$$

in $T \cdot H$. Moreover, similar to the remark made by by Kahrobaei and Shpilrain, the contrast between our protocol and the generic Diffie-Hellman key exchange protocol is based on the equality $\overline{g^n\overline{g^m t}} = \overline{g^n g^m t} = \overline{g^{n+m}t}$ rather than $(g^n)^m = (g^n)^m = h^{nm}$. This maneuver would not work in the Diffie-Hellman set up, because, if the shared key K was just the product of two openly transmitted elements, then anybody could determine the shared key.

Bibliography

[1] I. Anshel, M. Anshel, and D. Goldfeld, *An algebraic method for public-key cryptography*, Math. Res. Lett. **6** (1999), 287.

[2] G. Arzhantseva and A. Ol'shanskii, *The class of groups all of whose subgroups with lesser number of generators are free is generic*, Mathematical Notes **59** (1996), no. 4, 350–355.

[3] Y. Aumann, Y. Z. Ding, and M. O. Rabin, *Everlasting security in the bounded storage model*, IEEE Trans. Information Theory **48** (2002), no. 6, 1668–1680.

[4] S. Bachmuth and H. Y. Mochizuki, *$Aut(F) \to Aut(F/F'')$ is surjective for free group f of rank ≥ 4*, Trans. Amer. Math. Soc. **292** (1985).

[5] G. Baumslag, A. G. Myasnikov, and V. Shpilrain, *Open problems in combinatorial group theory.*, http://www.grouptheory.info/.

[6] G. Baumslag, R. Strebel, and M. W. Thomson, *On the multiplicator of $F/\gamma_c R$*, J. Pure Appl. Algebra (1980), no. 16, 121–132.

[7] Jean-Camille Birget, Spyros S. Magliveras, and Michal Sramka, *On public-key cryptosystems based on combinatorial group theory*, IACR Cryptology ePrint Archive **2005** (2005), 70.

[8] A. Borovik, A. G. Myasnikov, and V. Shpilrain, *Measuring sets in infinite groups,*, Contemp. Math., Amer. Math. Soc. **298** (2002), 21.

[9] R.M. Bryant, C.K. Gupta, F. Levin, and H.Y. Mochizuki, *Non-tame automorphisms of free nilpotent groups*, Communications in Algebra **18** (1990), no. 11, 3619–3631.

[10] J. C. Cha, K. H. Ko, S. J. Lee, J. W. Han, and J. H. Cheon, *An efficient implementation of braid groups*, pp. 144–156, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[11] T. Coulbois, *Propriétés de ribes-zaleskii, topologie profinie, produit libre et généralisations*, Ph.D. thesis, Université de Paris VII, 2000.

[12] K. Delaram and V. Shpilrain, *Using semidirect product of (semi)groups in public key cryptography*, Springer International Publishing, 2016.

[13] W. Diffie and M. E. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory **IT-22** (1976), 644–654.

[14] D. Garber, S. Kaplan, M. Teicher, B. Tsaban, and U. Vishne, *Probabilistic solutions of equations in the braid group*, preprint.

[15] Max H. Garzon and Yechezkel Zalcstein, *The complexity of grigorchuk groups with application to cryptography*, Theor. Comput. Sci. **88** (1991), no. 1, 83–98.

[16] S. Goldwasser and S. Micali, *Probabilistic encryption*, Journalof Computer and System Sciences **28** (1998), 270–299.

[17] Louis Goubin and Nicolas Courtois, *Cryptanalysis of the TTM cryptosystem*, ASIACRYPT 2000, Proceedings, 2000, pp. 44–57.

[18] Dima Grigoriev and Ilia V. Ponomarenko, *On non-abelian homomorphic public-key cryptosystems*, CoRR **cs.CR/0207079** (2002).

[19] ———, *Homomorphic public-key cryptosystems over groups and rings*, CoRR **cs.CR/0309010** (2003).

[20] ———, *Homomorphic public-key cryptosystems and encrypting boolean circuits*, Appl. Algebra Eng. Commun. Comput. **17** (2006), no. 3-4, 239–255.

[21] N. Gupta, *Free group rings*, Contemp. Math., vol. 66, American Math. Soc., 1987.

[22] M. Habeeb, D. Kahrobaei, and V. Shpilrain, *Public key exchange using semidirect product of (semi)groups*, in: ACNS 2013, Lecture Notes Comp. Sc. **7954** (2013), 475–486.

[23] P. Hall, *On the finiteness of certain soluble groups*, London Math. Soc., vol. 9, 1959, pp. 565–622.

[24] I. Kapovich, A. Myasnikov, P. Schupp, and V. Shpilrain, *Generic-case complexity, decision problems in group theory, and random walks*, Journal of Algebra **264** (2003), no. 2, 665 – 694.

[25] O. Kharlampovich and M. Sapir, *Algorithmic problems in varieties*, Internat. J. Algebra and Comput. **5** (1995), 379–602.

[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*, CRYPTO 2000 (Santa Barbara, California), Lecture Notes in Comput. Sci., vol. 1880, Springer, 2000, pp. 166–183.

[27] Chen L, Yi-Kai L, and et al. Jordan S, *Report on post-quantum cryptography*, Article, (NISTIR 8105). Gaithersburg (MD): National Institute of Standards and Technology, 2016.

[28] S. Mac Lane, *Categories for the working mathematician*, (1994).

[29] Eonkyung Lee, *Right-invariance: A property for probabilistic analysis of cryptography based on infinite groups*, Asiacryp.

[30] R. C. Lyndon and P. E. Schupp, *Combinatorial group theory*, Springer, 2001.

[31] W. Magnus, A. Karrass, and D. Solitar, *Combinatorial group theory: presentations of groups in terms of generators and relations*, Dover, 1966.

[32] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of applied cryptography*, CRC-Press, 1996.

[33] J. Milnor, *Growth of finitely generated solvable groups.*

[34] T. Moh, *A public key system with signature and master key functions*, Communications in Algebra **27** (1999), no. 5, 2207–2222.

[35] A. G. Myasnikov, V. Shpilrain, and A. Ushakov, *Non-commutative cryptography and complexity of group-theoretic problems*, Amer. Math. Soc. Surveys and Monographs (2011).

[36] Alexei G. Myasnikov and Alexander Ushakov, *Random van kampen diagrams and algorithmic problems in groups*, Groups Complexity Cryptology **3** (2011), no. 1, 121–185.

[37] C. H. Papadimitriou, *Computational complexity*, 1^{st}*ed.* ed., Addison-Wesley, Mass., 1994.

[38] A. N. Platonov, *An isoparametric function of the baumslag-gersten group*, (Russian) Vestnik Moskov. Univ. Ser. I Mat. Mekh., no. 3, 2005, pp. 12–17.

[39] N. S. Romanovskii, *The occurrence problem for extensions of abelian groups by nilpotent groups*, Siberian Mathematical Journal **21** (1980), no. 2, 273–276.

[40] J. J. Rotman, *An introduction to the theory of groups*, 4th ed ed., Springer-Verlag, New York, 1995.

[41] V. Shpilrain, *Automorphisms of F/R' groups*, Internat. J. Algebra and Comput. **1** (1991), no. 2, 177–184.

[42] V. Shpilrain and A. Ushakov, *Thompson's group and public key cryptography*, Lecture Notes in Comp. Sci., vol. 3531, 2005, pp. 151–163.

[43] ———, *The conjugacy search problem in public key cryptography: Unnecessary and insufficient*, Appl. Algebra Eng. Commun. Comput. **17** (2006), no. 3-4, 285–289.

[44] V. Shpilrain and G. Zapata, *Combinatorial group theory and public key cryptography*, Appl. Algebra Eng. Commun. Comput. **17** (2006), no. 3-4, 291–302.

[45] ———, *Using the subgroup membership search problem in public key cryptography*, Contemp. Math., Amer. Math. Soc. **418** (2006), 169–179.

[46] ———, *Using decision problems in public key cryptography*, Groups Complexity Cryptology **1** (2009), no. 1, 33–49.

[47] Tom Simonite, *The cia and jeff bezos bet on quantum computing*, Article, MIT Technology Review., 2012.

[48] N. R. Wagner and M. R. Magyarik, *A public key cryptosystem based on the word problem*, CRYPTO 1984 (Santa Barbara, California), Lecture Notes in Comput. Sci., vol. 196, Springer, Berlin, 1984, pp. 19–36.

[49] L. C. Washington, *Elliptic curves: Number theory and cryptography*, Chapman and Hall/CRC, 2008.

[50] W. Woess, *Cogrowth of groups and simple random walks*, Archiv der Mathematik **41** (1983), no. 4, 363–370.