Semirings as Building Blocks in Cryptography

Semirings as Building Blocks in Cryptography

By

Mariana Durcheva

Cambridge
Scholars
Publishing

Semirings as Building Blocks in Cryptography

By Mariana Durcheva

This book first published 2020

Cambridge Scholars Publishing

Lady Stephenson Library, Newcastle upon Tyne, NE6 2PA, UK

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

ISBN (10): 1-5275-4170-3
ISBN (13): 978-1-5275-4170-2

CONTENTS

ACKNOWLEDGEMENTS

This book is the revised, updated and expanded version of my monograph "Applications of Some Algebraic structures in cryptography", published in 2015 in Bulgarian under the editing of Prof. Ivan Trendafilov to whom I would like to express my sincere gratitude.

PREFACE

Nowadays when the main communication between people takes place over the Internet, information protection is crucial. With the development of virtual business and e-commerce, data sharing through unprotected public channels is expanded. The role of cryptography is to protect users' data from malicious intrusions. Public-key cryptography underlies the security features of many issues such as signed and encrypted email, single sign-on, and Secure Sockets Layer (SSL) communications. Public-key introduces a concept involving key pairs: one for encrypting, the other for decrypting. Various approaches are used to create the key exchange protocol. In this book we show the role of some algebraic structures as building blocks for these protocols. We first review and analyze some algebraic structures that have already been used to build cryptographic protocols – finite fields, groups and rings. Then we discuss the role in cryptography of the structures that are either not used for such purposes or used in a limited way. Such structures are idempotent semirings and semirings of the endomorphisms of a finite chain. Based on these studies, several key exchange protocols are developed. We hope that "Semirings as building blocks in cryptography" will be useful not only for cryptographers and specialists in Applied Algebra, but also for students of Cryptography or Applied Algebra.

CHAPTER ONE

INTRODUCTION

Let us answer the following questions honestly: Do the measures now used to increase information security offer comfort to ordinary people, or do these measures provide the protection we think we need? Nowadays, when the Internet is the main means of communication between millions of people around the world and constitutes an important trading tool, e-security is becoming extremely important.

There are many aspects of security, including security in e-commerce, e-money, peer-to-peer communications, password protection and digital signatures. One of the main aspects of secure communications is ensured by cryptography.

Cryptography is a centuries-old science. The term *cryptography* comes from Greek words $\kappa\rho\upsilon\pi\tau\sigma\sigma$ – hidden and $\gamma\rho\alpha\phi\omega$ – to write. In the past, it has been used primarily by military and diplomatic organizations to ensure the secrecy of their messages. The first data on the application of cryptography are from 1900 BC in ancient Egypt, where special hieroglyphs were used for this purpose. Nowadays, the role of cryptography has radically changed: it includes protection of information.

With the development of virtual business and e-commerce, data sharing is expanded through unprotected public channels. Since users who share information on unprotected channels are usually remote from each other, the low level of protection can tempt some malicious individuals to perform prohibited acts, such as disclosing the confidentiality of information, secretly modifying data, falsifying facts and so on.

In each cryptographic system, from a security point of view, the following requirements need to be met:

- *Authentication*: This is the process of establishing the authenticity of the subject.
- *Confidentiality*: This is to ensure the confidentiality of information, i.e., ensuring that the message you send will be read only by the person for whom it is intended.

- *Data integrity*: The aim is to ensure correct transmission of messages between users without making changes, additions, deletions or modifications.
- *Indisputability*: The aim is to ensure the origin of each message so that the person who sent it cannot subsequently abandon his authorship.

Encryption and decryption processes are managed by cryptographic keys. There are two types of cryptosystems: symmetric key cryptosystems (private-key cryptosystems), where the encryption and decryption procedures are performed with the same key, and asymmetric key cryptosystems, which are called more often public-key cryptosystems.

These two cryptosystems use two different types of keys – encryption keys and decryption keys. The encryption key is public and the decryption key is secret. Public key cryptosystems are used on unprotected channels, or when users are remote from each other and are unable to communicate directly. Because asymmetric algorithms are much slower than symmetric ones, the two types of algorithms are used in combination to optimize the speed of communication without compromising security.

Public key cryptosystems rely on the hardness of two main problems: the problem of *integer factorization* and the *discrete logarithm problem*. Of course, there is no strong evidence that these two problems are really hard. We recall that Peter Shor in [273] presented an algorithm for these problems that runs in polynomial time on a theoretical quantum computer. While some cryptographers do not agree on the possibility of physically implementing this model, the US National Security Agency (NSA) is concerned about this possibility to warn that government and industry should move away from these cryptosystems in the "not too far future" [232].

The most important issue in public key cryptography (the question $P \neq NP$?) is not yet solved. So, it can be concluded that most of the cryptographic primitives rely on unproven assumptions. Some researchers (see [18]) advocate so-called "reduction-based security" which means "to reduce the security of a great many cryptographic constructions to a relatively small number of simple-to-state and widely studied assumptions".

In public key cryptography there are two main approaches: The first one deals with algebraic (group-theoretical) constructions and is based on integer factoring and the discrete logarithm problem; the second one deals with geometric (coding/lattice) constructions and relies on geometric

computational problems on linear codes or integer lattices for their security.

To the first type can be assigned the RSA cryptosystem (based on the problem of integer factoring) [253], the Diffie-Hellman key exchange (based on the discrete logarithm problem) [83] and its elliptic curve variants ([213], [175]), the ElGamal encryption scheme [93], the Rabin-Miller test [249], and the Goldwasser-Micali scheme [126]. To the second type can be assigned the McEliece cryptosystem [205], the Goldreich-Goldwasser-Halevi Cryptosystem [125] and its variants, the NTRU cryptosystem invented by Hoffstein, Pipher and Silverman [147], and Knapsack cryptosystems [211].

It is known that the complexity of the algorithms for an "algebraic" cryptosystem is of the type $\mathbf{NP} \cap \mathbf{coNP}$. According to [18], both hard problems – integer factoring and discrete logarithm – fall into the class **TFNP**, which are **NP** search problems where every input is guaranteed to have a solution. Problems in this class cannot be **NP**-hard unless **NP = coNP**. The complexity of "geometric" constructions is $\mathbf{NP} \cap \mathbf{coNP}$ or **SZK**.

The aim of the present work is to investigate some algebraic structures that have so far not been used in cryptography and to consider the use of these structures as platforms for cryptographic protocols.

The monograph consists of four main chapters. The first of them has an introductory character, focusing on the applications of one of the basic "one-way" functions in public key cryptography, namely discrete logarithms. The discrete logarithm problem over finite fields, in groups and in semirings is considered. The second of them is devoted to the idempotent semirings, also termed *dioids*, which can be used as building blocks for various cryptosystems. The third of them considers distributed multicast key-management and a key-exchange protocol, based on idempotent semirings, is proposed. The fourth of them deals with endomorphism semirings of a finite chain; the concepts of simplices and simplicial complexes are examined as well. Several cryptographic protocols are built on these semirings.

CHAPTER TWO

THE ROLE OF DIFFERENT ALGEBRAIC STRUCTURES AS BUILDING BLOCKS IN PUBLIC KEY CRYPTOGRAPHY

2.1 Discrete Logarithms in Finite Fields

2.1.1 Definitions

Encryption has become tangibly more and more important in our everyday life. Many of the methods used to keep our communications secret and our important information private involve the *Discrete Logarithm Problem* (DLP) in some way. The difficulty of the DLP underlies security for many algorithms in public key cryptography, for performing tasks such as exchanging secret keys via public channels, authentication in electronic messages, digital signatures, and so on.

Definition. Let g be a generator of the group $\mathbb{F}_q$. For all nonzero elements h of $\mathbb{F}_q$, *the discrete logarithm* of h to the base g (denoted $\log_g(h)$) is the least nonnegative integer t of the set $\{0,\ldots,q-2\}$ such that $g^t = h$. Note that $\log_g(h)$ is unique modulo $q - 1$.

Definition. The *Discrete Logarithm Problem* in a finite field is: for the finite field $\mathbb{F}_q$ with a generator g of the group F_q^* and for the element $h \in \mathbb{F}_q^*$, calculate the discrete logarithm $\log_g(h)$.

The discrete logarithm problem in a finite *prime* field $\mathbb{Z}_p$ is equivalent to the problem of the solvability of the exponential *Diophantine equation*: $g^t = h + px$, or in other words, the discrete logarithm problem in a prime field $\mathbb{Z}_p$ can be viewed as a Diophantine function (see [330] for more details).

Some authors (as in [319]) examined the discrete logarithm problem as a formal problem specification, depending on the order of the cyclic group:

- *DLP* (Discrete Logarithm Problem) – when the order of the cyclic group is unknown;
- *DLKOP* (Discrete Logarithm with Known Order Problem) – when the order of the cyclic group is known;
- *DLKOFP* (Discrete Logarithm with Known Order Factorization Problem) – when the factorization of the order of the cyclic group is known.

2.1.2 Public key Cryptosystems based on the Discrete Logarithm Problem

The Diffie-Hellman key exchange protocol. In 1976 Whitfield Diffie and Martin Hellman published the paper *New Directions in Cryptography* ([83]), in which they proposed an algorithm that allows two users (called Alice and Bob for convenience) to communicate via an insecure (public) channel, creating a common secret key for this purpose.

The Diffie-Hellman Key Exchange Protocol consists of the following steps:

1. Alice and Bob publicly agree on the finite field $\mathbb{F}_q$, as well as on a primitive element $g \in \mathbb{F}_q^*$.
2. Alice and Bob choose respectively integers a and b from the set $\{2,\ldots, q-2\}$. These integers are their *secret* keys.
3. Alice computes g^a and transmits it to Bob (g^a is her *public* key), while Bob computes g^b and transmits it to Alice (g^b is his public key).
4. Alice computes $k_a = (g^b)^a = g^{ba}$ and Bob computes $k_b = (g^a)^b = g^{ab}$.

At the end of the protocol, both users obtain the same secret key $k = k_a = k_b$.

Definition. The problem of finding g^{ab} in the field $\mathbb{F}_q$ for given g, g^a and g^b is called the *Diffie-Hellman Problem* (*DHP*).

It is clear that if we can solve the DLP, then we could solve the DHP; however, the opposite has not been proven so far. For most groups, the

two problems (DLP and DHP) are considered to be of similar complexities (see [26], [43], [64], [176], [178], [328]).

Some authors (see, for example, [296]) discussed two variants of the Diffie-Hellman problem: the *Computational Diffie-Hellman problem (CDH)* and the *Decisional Diffie-Hellman problem (DDH)*.

In [49] and [201], several improvements of the Diffie-Hellman protocol are proposed.

The Diffie-Hellman's key exchange protocol is standardized: ANSI x 9.42 ([13]). Furthermore, it is the basis of a number of protocols, such as TLS.

The ElGamal cryptosystem. In 1985 Tahir ElGamal ([93]) published a public key cryptosystem, which is the following protocol:

Key generation

1. Alice selects the final field $\mathbb{F}_q$ and a primitive element g of this field.
2. Alice selects a random integer $a \in \{2,\ldots,q-1\}$, which is her secret key.
3. Alice computes $k_a = g^a$ and publishes her public key $(\mathbb{F}_q, g, k_a)$.

Encryption

1. Bob encodes the message m that he wants to send to Alice as an element of the field $\mathbb{F}_q$, using a public encryption scheme.
2. Bob selects a random integer $b \in \{2,\ldots,q-1\}$, which is his secret key.
3. Bob computes $c_1 = g^b$ and $c_2 = (k_a)^b m$.
4. Bob sends Alice the encrypted message (c_1, c_2).

Decryption

1. Alice receives the encrypted message (c_1, c_2).
2. Alice computes $c_1^{-a} c_2 = (g^b)^{-a} (g^a)^b m = g^{-ab} g^{ab} m = m.$
3. Alice decodes the message sent by Bob from the acquired m, with the help of the same public encryption scheme used by Bob.

Some authors (see, for example, [233]) stated that the security of the ElGamal cryptosystem is equivalent to that of the Diffie-Hellman protocol

and that the algorithms used to solve the Diffie-Hellman problem can also be used to break the ElGamal cryptosystem, and vice versa.

Various modifications of the ElGamal cryptosystem can be found in [16], [44], [56], and [188].

The Digital Signature Scheme (DSS). In the same paper [93], ElGamal also introduced a digital signature scheme, which owes its security to the difficulty of solving the discrete logarithm problem.

In 1994, the version of this scheme was accepted as a standard in the United States for public use (NIST standard). The ElGamal digital signature scheme (DSS) has undergone a number of modifications (see [314]).

We briefly describe the idea of the ElGamal digital signature scheme. It consists of three algorithms:

- The first phase is *keys generation*, or *distributed keys generation* (DKG), during which an appropriate large prime finite field $\mathbb{F}_p$ and secret key are selected. In addition, at this stage, the public key is computed.

Some improvements to the algorithm for this first phase are proposed in [132].

- The second phase is *message signing*.
- The third phase is *signature verification*.

Peter Schnorr ([269]) suggested a variant of the ElGamal scheme which has some advantages over the classical ElGamal scheme; for example, the Schnorr scheme is not vulnerable to the "adaptive chosen message" (see [242]). In order to make the ElGamal scheme more resistant to attacks of the type mentioned, a number of authors (see [311], [327], [328]) proposed improvements to the ElGamal scheme. Modification of the ElGamal scheme is presented in [201]. In [237] it is demonstrated that breaking the Schnorr scheme is not equivalent to solving the DLP.

The free software GnuPG uses ElGamal DSS as a standard for digital signature (see [207], [229]). This scheme is also the basis of the products Open SSL and Pretty Good Privacy (PGP), as well as of other software (see [236]).

It is predicted that in the near future the discrete logarithm problem will not be difficult to solve, and this will affect the length of the keys of these cryptosystems that owe their security to this problem (see [186]).

There are various trials to find the explicit form of the discrete logarithms over some finite fields. Mullen and White in [222] for the first time gave an explicit form of the discrete logarithm over special finite

fields. Meletiou [209] and Niederreiter [230] showed the explicit form of the discrete logarithm over the field $GF(p,k)$ (see also [208]). Wan improved their results in [321]. These explicit forms have no big value for cryptographic purposes, and we will not focus our attention on them.

In recent years, a number of papers have appeared on polynomial approximation of discrete logarithms over a finite field (see [48], [171], [182], and [206]).

2.1.3 Generic algorithms for solving the Discrete Logarithm Problem

Some known attacks on the discrete logarithm problem are considered in [94] and [170]. An algorithm that computes a discrete logarithm is called *generic* if it can solve the DLP in an arbitrary finite cyclic group.

The Baby-step, giant-step algorithm. This algorithm was first proposed by Shanks [271], and it is a generic method applicable to each cyclic group.

Let $G = \langle g \rangle$ be a cyclic group of prime order p, and $h \in G$. We want to find the value of the integer k $(0 \le k \le p-1)$ modulo p, such that $h = g^k$.

Let $k = k_0 + k_1[\sqrt{p}]$. Then (since $k \le p-1$), the following inequalities hold: $0 \le k_0$, $k_1 \le [\sqrt{p}]-1$. It is clear that to find k, it is enough to find the values of k_0 and k_1.

The *baby-step* consists of computing $g_i = g^i$ for $0 \le i \le [\sqrt{p}]$; the *giant-step* consists of computing $h_j = h.g^{-j[\sqrt{p}]}$ for $0 \le j \le [\sqrt{p}]$. Then the algorithm tries to find a match between the individual values in the tables obtained in the two steps, i.e. the goal is to find such g_i, for which $g_i = h_j$. Each element, that is common to both tables, allows us to find the discrete logarithm of h. For each cyclic group of prime order p, this algorithm has running time $O(\sqrt{p})$; it also requires $O(\sqrt{p})$ memory to store values in the tables.

The Pollard ρ method. The main idea for solving the DLP $g^k \equiv h \pmod p$ in $\mathbb{F}_p^*$ is to find a match between $g^i h^j$ and $g^l h^m$ for some known exponents $i, j, l, m \in \mathbb{N}$ (see [248]). Then $g^{i-l} = h^{m-j}$ and

by taking the roots in $\mathbb{F}_p$ we will solve the problem of representing h as a power of g. The difficulty is to find a function $f:\mathbb{F}_p \to \mathbb{F}_p$ which is both easy to compute and seemingly random. If the function $f:\mathbb{F}_p \to \mathbb{F}_p$ has these properties, then the expected running time of the algorithm is $O(\sqrt{p})$ group multiplications.

In [303] E. Teske suggested an alternative method for the series of generators. Better parameters for Teske's iteration function were achieved in [17], [172], [214]. In [235] a parallel search for matches was considered, which is used in cryptography. A new efficient matching algorithm for the Pollard ρ method was presented in [322]. Some ways to speed up the performance of the Pollard ρ method were discussed in [63].

At the same time, Pollard suggested another method, called the λ method (ses [248]). Like the ρ method, the λ method is based on finding a sequence of collisions and the complexity of the algorithm is estimated by the so-called *birthday paradox.*

In 2000, Pollard published an article ([247]) in which he revised both methods using improvements suggested by different authors.

The Pohlig-Hellman algorithm. This algorithm ([242]) works in any cyclic group. In [51] is shown that the discrete logarithm problem in a cyclic group G can be reduced to the DLP in a cyclic group of prime order if the factorization of the group order is known:

$$n = |G| = \prod_{p/n} p^{\varepsilon(p)}.$$

Here $\varepsilon(p)$ means the maximal power such that $p^{\varepsilon(p)}$ divides n.

Let us assume that we can factorize the integer p–1 into prime factors:

$$p-1 = \prod_{i=1}^{n} q_i^{\alpha_i}.$$

Then the Pohlig-Hellman algorithm works in 3 steps:

- *Step 1* consists of creating a table of numbers;
- In *Step 2*, the table from Step 1 is used for obtaining the values of $\log_g h \pmod{q_i^{\alpha_i}}, i = 1,\ldots,s$;
- In *Step 3*, the *Chinese remainder theorem* is used to compute $\log_g h \pmod{p-1}$ from the obtained $\log_g h \pmod{q_i^{\alpha_i}}$.

The Pohlig-Hellman algorithm has the polynomial complexity $O\left((\log p)^{C_1}\right)$, when all prime divisors q_i of p–1 are smaller than $(\log p)^{C_2}$, for nonzero constants C_1 and C_2. If p–1 has a prime divisor q, such that $q \geq p^C$ for a positive constant C, then the complexity of the Pohlig-Hellman algorithm becomes exponential (see [317]).

For the field $\mathbb{F}_{2^n}$ of characteristics 2, it is possible to choose n so that the number $2^n - 1$ is prime (*Mersenne prime*). In this case, the Pohlig-Hellman reduction presents no benefit for attacking. For fields of odd characteristics, there exist primes p, known as *Sophie Germain primes*, with the property that the numbers of type $2p+1$ are also prime. Utilizing such numbers minimizes the usefulness of the Pohlig-Hellman reduction (see [80] and [251]).

Lower bounds. In 1997 Victor Shoup improved the results from [228] and defined a lower bound for a cyclic group of order p^r for a prime p (see [274]). In his model, the generic algorithm starts with 1 and g^x; during run-time it supports a list with elements of the group g^{s_i}. The discrete logarithm can only be calculated by finding the collision. After m group operations, the probability of collision is $O(m^2 p)$.

A combinatorial point of view on generic attacks on the DLP was first suggested by Schnorr ([270]) and was further developed in [60], where characteristics are given for generic attacks on groups of prime order (see also [296]). In [215], the theory of lower bounds for the generic group model of the discrete logarithm problem was developed, constrained by the subset $S \subseteq \mathbb{Z}_p$, known to the attacker (constrained DLP).

2.1.4 Index Calculus algorithms for finite fields

The term "*index calculus*" describes a family of algorithms for computing discrete logarithms in which the details of calculations depend on the fields used (see [233]).

The first known method of computing the discrete logarithms is due to Adleman ([3]). His algorithm is applicable to prime order fields $\mathbb{F}_p$. Recently, an algorithm has been developed for all finite fields [1].

The first generalization for the fields $\mathbb{F}_{p^n}$ was given by Hellman and Reyneri ([144]). In [42], this method was improved for an arbitrary field

$\mathbb{F}_{p^n}$, and a variant for the field of characteristic 2 ($\mathbb{F}_{2^n}$) was also proposed. Then, the last concept was extended by Coppersmith, Odlyzko, Schroeppel in [73], in order to create a fast algorithm (COS) for the field $\mathbb{F}_{2^n}$.

The *Waterloo* algorithm, which is a variant of the index calculus method for computing discrete logarithms in the field $\mathbb{F}_{2^n}$, was suggested in [87]. The possibility of using the factor base for the field $\mathbb{F}_{p^n}$ is considered in [29].

The best known algorithm for computing the discrete logarithms in the field $\mathbb{Z}_p^*$ is a variant of the index calculus method, called the *number field sieve* (NFS). In [2] is also considered a *function field sieve*. Different improvements of the algorithm NFS were suggested in [161], [263]–[268]. A variant of the index calculus method for computing the discrete logarithms on an *algebraic torus* was shown in [134].

We will briefly discuss the index calculus method.

Let us consider the equation $g^x = h$, where $g, h \in (\mathbb{Z} / p\mathbb{Z})^*$, p is prime, and g is of order $p - 1$ (modulo p).

The first step of the algorithm is to choose a factorial base B, which is a subset consisting of "small" prime numbers.

The second step is to compute the discrete logarithm for the selected element h, using discrete logarithms of the elements of the factor base.

A new methodology for this phase of the algorithm is proposed in [237].

The complexity of the Adleman algorithm is $L_p(\frac{1}{2}; c)$; the COS algorithm achieves $L_p(\frac{1}{2}; 1)$; the complexity of the NFS algorithm is $L_p(\frac{1}{3}; 1{,}923)$ (see [148], [210], [317]).

The index calculus method for solving the DLP in the field $\mathbb{F}_p^*$ is subexponential (see [148], [95]).

Progress in computing the discrete logarithm in the multiplicative group of the field of characteristic 2 was achieved by Joux ([159]), whose algorithm for a finite field of order $Q = p^n$ reached the heuristic complexity $L_Q(\frac{1}{4}; o(1))$. Gary McGuire et al. improved Joux's algorithm

and set a world record for computing the discrete logarithm using a 1971 bit number (see [127]).

Recently, quasipolynomial-time algorithms have been shown for discrete logarithms over finite fields of small characteristic [162].

2.2 Group Based Cryptography

Definition. Let G be a finite cyclic group of order n. Let g be a generator of G and let $h \in G$. Then a *discrete logarithm* of h at base g (denoted by $\log_g(h)$) is the only integer x of the interval $0 \leq x \leq n-1$.

In cryptographic literature, two main approaches to group-based cryptography are considered. In the first approach, it is suggested to use abelian groups, and in the second one, nonabelian (see [309]). To investigate the possibility of using the group as a platform of a cryptographic protocol, attention should be paid to several questions: how should it be set and what properties should a given group, taken as a potential candidate, have in order to be a platform for building cryptographic protocols; what should the algorithmic problem be which is the basis of the protocol; what are the general principles of the proposed constructions and which are the conditions providing the resilience of the cryptosystem?

2.2.1 The Discrete Logarithm Problem in abelian groups

Here we will briefly discuss some of the applications of abelian groups in public key cryptography. Simon Blackburn showed in his paper [33] that by using *Picard's groups* in finite graphs, the DLP can be efficiently solved in *Biggs's groups*. The same author in [34] breaks the Arifin-Abu cryptosystem, indicating that the discrete logarithm problem in this case is easily computable.

In [35], a cryptosystem based on *Drinfeld modules* is considered; it is shown that this cryptosystem is insecure. In several papers, there is a suggestion of using cyclic subgroups of matrix groups, but in these groups the DLP is not harder than the DLP over the multiplicative group of the finite field (see [184] for details).

In [194], an ElGamal like public-key cryptosystem is proposed in which the order of the underlying cyclic group is hidden. In paper [300], a generic algorithm for computing discrete logarithms in a finite abelian p-group is presented. Paper [288] is a survey of twisting commutative algebraic groups and applications to discrete logarithm based

cryptography. Different applications of the DLP in abelian groups can be found in [111]–[115].

To the cryptosystems based on the DLP in abelian groups can be also assigned the cryptosystems operating in groups of points over elliptic curves, as well as those using the Jacobian of hyperelliptic curves.

Excluding these examples, it can be summarized that the cryptosystems, based on the difficulty of the discrete logarithm problem in abelian groups, are not really secure. This is the reason why the cryptographic community focuses on the use of nonabelian groups.

2.2.2 The Discrete Logarithm Problem in nonabelian groups

The idea of using nonabelian groups in cryptography originated with Wagner and Magyarik ([320]) in 1985. This cryptosystem owes its security to the difficulty of solving the *word problem* for finitely represented groups, but the cryptoscheme proposed by the authors is too theoretical, with unresolved issues. A cryptoanalysis of the Wagner-Magyarik scheme can be seen in [27], [130], and [187].

There are finitely represented groups with a recursively unsolved word problem (see, for the details, [231]). Birget et al. in [28] presented their cryptosystem, which is inspired by the Wagner-Magyarik scheme. The cryptoanalysis of it is shown in [40], [131]; in the last article, it is also proved that the cryptosystem [119] is insecure.

Surveys of the application of nonabelian groups in cryptography are presented in [102], [103], [226], and [227]; especially for the discrete logarithm problem in non-abelian groups, see: [153] – [155], and [173].

The conjugator search problem. One of the possible generalizations of the discrete logarithm problem in arbitrary groups is the *conjugator search problem*, which can be formulated as follows: for two elements a, b from a nonabelian group G, find at least one element $x \in G$ such that $a^x = b$. Here by a^x we understand xax^{-1}. The computational complexity of this problem in some special groups (e.g. in the braid group) is used in a number of cryptosystems ([277], [280] and [285]).

The conjugator search problem is the basis of the security of the two most popular cryptosystems based on non-abelian groups, namely:

- Ko, Lee, Cheon, Han, Kang and Park cryptosystem ([174]);
- Anshel, Anshel and Goldfeld cryptosystem ([11], [12]).

Both of them use the braid group.

Of course, the braid group is not the only platform for cryptosystems using the conjugator search problem. As such, *Thompson's groups* F, *matrix groups*, *Artin's groups*, *Grigorchuk's groups* are also used (for details, see [226]).

We will briefly review the above-mentioned two protocols, which are analogous to the Diffie-Hellman key-exchange, and in which the conjugator search problem is the basis of their security.

Ko-Lee-Cheon-Han-Kang-Park key-exchange protocol. Let G be a nonabelian group and let g be a publicly known element of G. Let A and B be two commuting subgroups of the group G, given by their generating (finite) subsets, such that $ab = ba$ for all $a \in A$, $b \in B$.

Alice and Bob, who want to create a common secret key, do the following:

1. Alice selects an arbitrary element $a \in A$, computes $g^a = a^{-1}ga$ and transmits the obtained value to Bob.
2. Bob selects an arbitrary element $b \in B$, computes $g^b = b^{-1}gb$ and transmits the obtained value to Alice.
3. Alice computes $k_A = (g^b)^a$, while Bob computes $k_B = (g^a)^b$.

Since $ab = ba$, it follows that $k_A = k_B$.

Anshel-Anshel-Goldfeld key-exchange protocol. The advantage of this protocol is that, unlike the Ko et al. protocol, there is no need to search for commuting subgroups A and B of group G. In addition, it can be applied to any nonabelian group in which the word problem can be efficiently solved.

Definition. Let S be an arbitrary set. A *word* w in S is a finite sequence (it may be empty as well) of elements

$$w = y_1 \ldots y_n,\ y_i \in S.$$

The integer n is called the *length* of the word w.

Definition. The *word problem* (WP) consists of the following: For a given recursive representation of the group G and an element $g \in G$, decide whether $g = 1$ in G.

Then the protocol consists of the following:

Let G be a nonabelian group and let the elements $a_1,...,a_l,b_1,...,b_m \in G$ be public. In order to create a common secret key, Alice and Bob have to proceed in the following way:

1. Alice chooses a word x from $a_1,...,a_l$ and transmits $b_1^x,...,b_m^x$ to Bob.
2. Bob chooses a word y from $b_1,...,b_m$ and transmits $a_1^y,...,a_l^y$ to Alice.
3. Alice computes $x(a_1^y,\ldots,a_l^y) = x^y = y^{-1}xy.$
4. Bob computes $y(b_1^x,...,b_m^x) = y^x = x^{-1}yx.$
5. The shared secret key k is the commutator $[x,y] := x^{-1}y^{-1}xy.$ To obtain this key, Alice multiplies $y^{-1}xy$ on the left by x^{-1}, while Bob multiplies $x^{-1}yx$ on the left by y^{-1} and then gets the inverse element $(y^{-1}x^{-1}yx)^{-1} = x^{-1}y^{-1}xy.$

The difficulty of the protocol is based on the difficulty of the following

Problem. For given elements $x,a_1,...,a_l$ and group G, find the representation (if it exists) of x as a word from the sequence $a_1,...,a_l$.

It is worth noting that this protocol is not fully determined, as it is necessary to specify how to select the elements a_i,b_j and how Alice and Bob generate the words x and y, respectively. Some attacks on this protocol can be seen in [149], [224], and [225].

Some authors (see [102]) consider also the following

Simultaneous search conjugacy problem. Let G be a finitely represented group and let the following elements of the group be given:

$u_1,\ldots,u_k,v_1,\ldots,v_k \in G,$ so that $x^{-1}u_ix = v_i$ for all $i \in \{1, 2, \ldots, k\}$.

Finally, an algorithm is needed to find an element $z \in G$ that satisfies the condition $z^{-1}u_iz = v_i$ for all $i \in \{1, 2,\ldots, k\}$.

The decomposition problem and protocols based on it.

The decomposition problem is the following: in a non-abelian group G, find two elements a, $b \in G$ (usually asking for elements a, b of some subgroup of G) with the property $h = a \cdot g \cdot b$ for g and h being two known elements of the group (see [226]).

In general, a protocol, whose security is based on the decomposition problem, consists of the following:

Two users publicly agree on group G, the element $g \in G$, and two subgroups $A, B \subseteq G$, whose elements commute, i.e. $a \cdot b = b \cdot a$ for all $a \in A, b \in B$.

1. Alice selects two random elements $a_1, a_2 \in A$. She transmits the value $a_1 \cdot g \cdot a_2$ to Bob.
2. Bob selects two random elements $b_1, b_2 \in B$. He transmits the value $b_1 \cdot g \cdot b_2$ to Alice.
3. Alice computes $k_A = a_1 \cdot b_1 \cdot g \cdot b_2 \cdot a_2$, while Bob computes $k_B = b_2 \cdot a_1 \cdot g \cdot b_1 \cdot a_2$.

Since $a_i \cdot b_i = b_i \cdot a_i$ in group G, by the end of the protocol the two users receive the shared secret key $k_A = k_B = k$ (as an element of G).

Some of modifications of the above protocol are:

- the Shpilrain-Ushakov protocol for "twisted" groups ([281]);
- the Shpilrain-Ushakov protocol for hidden subgroups ([282]);
- the Kurt protocol for triple decomposition ([180]);
- the Stickel protocol ([295]).

We point out that the Shpilrain-Ushakov protocol used Thompson's groups as platform, noting that the word problem in these groups is solvable for almost linear time. In [86] it was shown that these cryptosystems are not secure.

Thompson's groups are also used in [313]. Stickel employed matrix groups as a platform for his protocol, which was broken by Sramka [291], Shpilrain [278], and Mullan [220].

Logarithmic signatures and cryptosystem using them. *Logarithmic signature* for the finite group G is called an ordered n-tuple $\alpha = (A_1, A_2, \ldots, A_n)$ of the subsets A_i of G, such that every element $g \in G$ can be represented in an unique way in the form $g = a_1 \ldots a_n$ for $a_i \in A_i$.

A natural way to construct a logarithmic signature for a group G is to select subgroups from the following chain

$$1 = G_0 < G_1 < \ldots < G_n = G .$$

Let A_i be the set of representatives of cosets G_{i-1} in G_i. Then $\alpha = (A_1, A_2, \ldots, A_n)$ is a logarithmic signature of the group G.

Cryptosystems using logarithmic signatures are suggested in [185], [190]–[192], [289].

We note that Lempken et al. (see [184]) invented the cryptosystem MST_3, based on logarithmic signatures, for which Magliveras et al. in [193] showed that it is not secure using a special method of generating secret keys. Another attack against the cryptosystem MST_3 was described in [38].

Paeng et al. in [238] proposed a new cryptoscheme based on the difficulty of the DLP in the group of *inner automorphisms*.

A cryptoanalysis of the protocol proposed by Grigoriev and Ponomarenko ([135]) was made in [65] using the heuristic method of rediscovering the secret key from the public key.

In [92], a new cryptosystem using *polycyclic groups* is proposed. Frey [107] showed that the Brauer group plays an important role in cryptosystems whose security is based on the discrete logarithm problem.

In [259], the authors (using some ideas from [287]) proposed a key-exchange protocol, the security of which relies on two simultaneous problems in group representation level: the matrix conjugator search problem and matrix discrete logarithm problem.

2.2.3 Groups used as platforms for cryptosystems

The braid group. Due to the fact that many of the cryptosystems using groups are based on the *braid group*, we will first focus on it.

The braid group was introduced by E. Artin in 1947. This group is very interesting in many aspects: there are equivalent representations in completely different mathematical areas; the word problem in this group is solvable relatively easy, but some other problems, such as the problem of the conjugate element and the decomposition problem, in this group seem to be hard to solve.

There are several definitions of braid groups; we consider one algebraic-geometric definition. For $n \geq 2$, the braid group B_n is defined as follows:

$$\left\langle \sigma_0,\ldots,\sigma_{n-1} \left| \begin{array}{ll} \sigma_i\sigma_j = \sigma_j\sigma_i & \text{for } |i-j| \geq 2 \\ \sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1} & \text{for } |i-j| = 1 \end{array} \right. \right\rangle.$$

This way of defining the braid group is called *Artin's representation* and the generating elements are called *Artin's generators*. We refer to σ_i, $0 \leq i < n-1$ as an elementary braid on n strands and interpret that as the braid that interchanges strand i and $i+1$ by passing $i+1$ over i. An element from B_n is called an n-braid. For each n, the identity mapping on

$\{\sigma_1,\ldots,\sigma_{n-1}\}$ induces an embedding from B_n into B_{n+1}, so that we can consider an n-braid as a part of (n+1)-braid. Since B_2 is an infinite cyclic group, it is isomorphic to the group of integers. For $n \geq 3$, the group B_n is not commutative and its center is an infinite cyclic subgroup. When a group is specified using presentation, each element of the group is an equivalence class of words with respect to the congruence, generated by the relations of the presentation. Hence (by definition), every n-braid is an equivalence class of n-braid words under the congruence.

Birman et al. in [29] and [30] presented a new canonical form for the elements of the braid group. Another normal form is given in [91] and [156]. Some efforts to solve the conjugacy problem in polynomial time can be found in [128] and [183].

Campagna in [54] proposed a new canonical form called the *max run form*, using Artin's generators, and also provided some algorithms that can be used for cryptographic purposes.

Wiest in [325] showed an algorithm for finding a unique short representative for any given element of Artin's braid group. Hofheinz and Steinwandt in [149] used a heuristic algorithm for attacking the conjugacy search problem, which is the basis of the cryptosystems presented in [11] and [174]. Myasnikov and Ushakov ([224], [225]) suggested a variant of the "length-based attacks" for these cryptosystems. Similar attacks are proposed in [54]. Other attacks are also considered in [45] and [223]. Chowdhury in [66] showed that the suggested implementation of the "Algebraic Eraser scheme" to the braid group is not secure. For more details about the *algebraic eraser*, see also [124]. The algebraic eraser is used in the recently patented protocol [10].

Some solutions to the word problem in the braid group based on the "handle reduction process" are discussed in [79] and [117]. Three authentication schemes based on the conjugacy search problem and the root extraction problem are presented in [286]. A cryptanalysis of the root extraction problem can be found in [137].

There are also attempts to combine various problems in the braid group to create more resistant cryptosystems. For example, Thomas and Lal in [304] suggested a digital signature scheme that combines the conjugate search problem, the decomposition problem, and the root search problem.

Chowdhury in [67] considered cryptographic protocols using noncommutative semigroups to improve the security of the Cha-Ko-Lee-Han-Cheon cryptosystem based on the braid group, while providing two algorithms to solve the decomposition problem.

The Thompson group. The Thompson group has the following infinite presentation:

$$F = \langle x_0, x_1, x_2, \ldots \mid x_i^{-1} x_k x_i = x_{k+1}\ (k > i) \rangle.$$

The classic normal form for the Thompson's group is a word of the type

$$x_{i_1} \ldots x_{i_s} x_{j_t}^{-1} \ldots x_{j_1}^{-1},$$

so that the following two conditions are met:

- $i_1 \le \ldots \le i_s$ and $j_1 \le \ldots \le j_t$;
- if both x_i and x_i^{-1} are present, then either x_{i+1}, or x_{i+1}^{-1} is also present.

There is a relatively simple procedure for reducing an arbitrary word w to normal form in the Thompson group. (Different properties of the Thompson group are discussed in [55]).

We want to note that in the Thompson group there are effective attacks on the decomposition problem ([200]). As some authors have stated (see, for example, [226]), in the Thompson group, as well as in the braid group, the use of different representations of normal forms for group elements poses cryptographic risk due to the fact that what we are trying to hide in one normal form is quite possibly revealed in another.

Matrix groups. The advantage of using matrix groups over finite commutative rings as platforms for cryptographic protocols is that, on the one hand, the matrix product is not commutative, and on the other hand, matrix entries that are elements of the commutative ring offer a good mechanism to hide the information we want. Another advantage of matrix groups is the periodicity of the matrix when it is defined over a finite ring.

Different finite rings can be considered, over which matrix groups can be defined. The simplest example is the ring $\mathbb{Z}_n$. We point out that matrix groups over the ring $\mathbb{Z}_n$ can also be used as a platform for the "classical" Diffie-Hellman key-exchange protocol, but the disadvantage is that the number n must be very large to provide a relatively efficient key space.

Another ring of interest is $R = \mathbb{F}_p[x]/(f(x))$ (here $\mathbb{F}_p$ is a field with p elements, $\mathbb{F}_p[x]$ is a ring of polynomials over the field $\mathbb{F}_p$, and ($f(x)$) is an ideal of $\mathbb{F}_p[x]$, generated by an irreducible polynomial $f(x)$ of degree n). This ring is isomorphic to the field $\mathbb{F}_{p^n}$, but the presentation of R allows a large key space to be obtained with relatively small basic parameters. This ring is employed by Tillich and Zemor in [306] to

construct hash functions (see also [276]). The matrices used by them are from the group $SL_2(R)$.

It is also possible to use the ring of the reduced polynomials over the ring $\mathbb{Z}_n$. *Reduced polynomials* are expressions of the type $\sum_{k=0}^{N} a_k x^k$, taken with the normal addition, and the multiplication is in accordance with the rule

$$x^i \cdot x^j = x^{(i+j) \mod (N+1)}.$$

The ring of reduced polynomials is a factor-ring; the ideal generated by the polynomial x^{N+1} is prime, and, for this reason, the factor-ring calculations are quite efficient. This ensures that a small amount of money is spent to provide a large space of keys.

There are also mixed protocols in which the authors combined matrix groups with some other groups. For example, in the Climent-Ferrandez-Vicent-Zamora key-exchange protocol (see [68]), matrix groups are combined with the sets of points on an elliptic curve defined over a finite field. Analysis of the protocol was performed in [69], where a number of improvements were proposed to enhance security.

Another example of a mixed protocol is the one proposed by P. Vitkus, E. Sakalauskas, N. Listopadskis, and R. Vitkiene in ([318]), which uses the left and right actions of a matrix on a matrix. As a platform for their protocol, the authors employed the braid group B_n.

The following map is used to transform the braid group into a matrix group

$$\rho : B_n \to GL(n-1, \mathbb{Z}_p), \ p \text{ prime};$$

$$\sigma_i \mapsto I_{i-2} \oplus \begin{pmatrix} l & -t & 0 \\ 0 & -t & 0 \\ 0 & -t & 1 \end{pmatrix} \oplus I_{n-i-2}$$

for $t \in \mathbb{Z}_p$.

Cryptanalysis of the protocols using matrix groups is shown in [218].

Extra special groups. For a given prime p, all groups of order p^2 are abelian. The first nonabelian group G is of order p^3. There exists a complete classification of groups of order p^3.

For odd p, there are two non-isomorphic classes of extra special groups of order p^3: