Analysis and Applications of Two Group-Theoretic Problems in Post-Quantum Cryptography

Christopher Battarbee

PhD

University of York

Computer Science

July 2023

Abstract

This thesis makes significant contributions to the analysis of two computational problems arising from a cryptosystem in group-based, post-quantum cryptography, and proposes a novel application of the underlying mathematical structure.

After an introductory Chapter 1 setting the historical context in which our research appears, Chapter 2 begins by introducing Semidirect Product Key Exchange (SDPKE), a generalisation of the famous Diffie-Hellman Key Exchange. Various cryptosystems are discussed in this framework and their respective cryptanalyses are systematised and interpreted as analysis of the complexity of a computational problem called the Semidirect Computational Diffie-Hellman problem. We also augment some of this analysis with our own results, and fill out technical gaps implicit in the literature.

SDPKE also naturally gives rise to an analogue of the Discrete Logarithm Problem, called the Semidirect Discrete Logarithm Problem (SDLP). Almost nothing was known about this problem - partially because of a misunderstanding of its importance in the literature - but in Chapter 3 we classify its quantum complexity by proving that the structure of SDPKE occurs as an example of a so-called cryptographic group action. Doing so requires the development of a bespoke quantum algorithm to get around certain technical difficulties; this is the first example of a quantum algorithm constructed for use in the cryptanalysis of group-based cryptography.

The structure of a cryptographic group action gives us access to a surprisingly rich variety of work, including an idea for an efficient Digital Signature Scheme based on the structure of cryptographic group actions. In Chapter 4 we define this scheme, christened SPDH-Sign; we prove its security, and show that the SDPKE-type group action offers advantages with respect to efficient sampling compared to other group actions. We also propose a particular group for use with SPDH-Sign, taking into account the cryptanalytic work discussed throughout the rest of the thesis.

List of Contents

List of Tables

List of Figures

Acknowledgements

I am chiefly indebted to the tireless efforts of my PhD supervisors, Delaram Kahroabei and Siamak F. Shahandashti. Both have been in their own, complementary ways, utterly invaluable. In particular I should like to thank Delaram for keeping me on as a student even when moving across the Atlantic, and for hosting me at the City University of New York in February to March of 2022; I should like to thank Siamak for his eye for detail, and warm friendship closer to home.

The Department of Computer Science at the University of York has been my professional home for close to three years now, and I would like to thank all of the pastoral staff and friends I have made within these halls. The first year or so of my PhD was marred by the COVID pandemic of the early 2020s, and in this the Department also provided immaculate support to a young man in a strange new city.

Special technical thanks go to Ludovic Perret at the Sorbonne, who spotted the potential in my more mature work and pushed me towards successful publication; and to my former undergraduate peer and old friend Alfred Dabson at City, University of London, for his consistently useful advice in all matters mathematical.

Finally, and by no means least, the utmost gratitude goes to my wonderful partner Sophie, without whom the completion of this PhD would not have been possible. I dedicate this work to her.

Declaration

I, Christopher Battarbee, declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for a degree or other qualification at this University or elsewhere. All sources are acknowledged as references.

The material in this thesis has been published as the following papers: *Cryptanalysis of Semidirect Product Key Exchange Using Matrices Over Non-Commutative Rings*, published in Mathematical Cryptology [3]; *On the efficiency of a general attack against the MOBS cryptosystem*, published in Mathematical Cryptology [5]; *Semidirect Product Key Exchange: the State of Play*, to appear in Journal of Algebra and Applications [2]; *A Subexponential Quantum Algorithm for the Semidirect Discrete Logarithm Problem*, submitted to Asiacrypt 2023 [4]; and *SPDH-Sign: towards Efficient, Post-quantum Group-based Signatures*, to appear at PQCrypto 2023 [6]. Each of these works was authored solely by me, with editorial remarks and advice coming from the various co-authors. A detailed list of where the results from these papers appear in the thesis is contained in Chapter 1.

The thesis makes use of the pronoun *'we'*, rather than *'I'*. This is a stylistic choice common to the field, and should not be taken to imply that work was completed collaboratively.

1 Introduction

> Make not your thoughts your prisons.
>
> *Antony and Cleopatra*, Act V

Suppose that you are an ill-behaved student in a classroom, and that you wish to communicate with an equally unscrupulous neighbouring classmate. Your only means of doing so - for speaking aloud will rouse the unwelcome attention of the notoriously strict teacher - is to pass each other notes. Your classmate sits at the desk adjacent to yours: you may therefore pass notes between each other directly, without fear of any impersonation of your classmate by third parties, nor the possibility of your messages being tampered with in transit. There is, however, a possibility that the ever-vigilant teacher will spot you exchanging notes and read them out for the class. In order to avoid this embarrassing scenario, how can you ensure that the contents of your notes remain known only to you and your classmate?

Historically - at least as far back as the time of Caesar - you and your intended recipient would, before the beginning of class, agree upon a secret *key*, under which your messages (or *plaintexts*) can be *encrypted* to produce a *ciphertext*. Your classmate, sharing the same secret key, can *decrypt* the resulting information, with the added property that even if the method of encryption and decryption is known, it is difficult to recover a message from its ciphertext without access to the secret key. Suppose, however, that the teacher suspects this tactic may be employed, and randomises the class seating plan. Your former neighbour with whom a key has been agreed is now seated too far away for notes to be reliably passed. Fortunately, you have by chance been seated next to another trusted peer, but the teacher's plan of randomisation has seemingly worked: without the prior agreement of a secret key, efforts to ensure the privacy of communication with your new neighbour appear hopeless.

A solution to this problem is given by Whitfield Diffie and Martin

Hellman in their 1976 paper *New Directions In Cryptography* [26]. The idea, in terms of our analogy, is effectively to use the notes themselves to replace the prior agreement of a secret key - henceforth referred to as a *shared key* to avoid a confusion of nomenclature. Indeed, two neighbouring students can be thought of as in possession of distinct pairs $(a, A), (b, B)$, where a can be thought of as the solution to some computational problem implied by public rules and the value A. As such, a and b are kept private to their respective owners but A and B may be shared freely. Suppose some public function f is such that $f(a, B) = f(b, A)$; in order to arrive at the required shared key, the parties need only exchange the values A and B, and use the function f to derive a shared key from their private values a and b. So long as it is difficult to recover the shared key from the values A and B, and a and b remain private, we may now proceed as before with the privacy of our messages protected by encryption under the shared key. Notice that the privacy of this approach now relies additionally on the difficulty of the computational problem implied by a pair (a, A).

In order for this to be useful we need to find a suitable function f; Diffie and Hellman achieve this in [26]. The resulting protocol became known as *Diffie-Hellman Key Exchange* (DHKE), and its invention is now regarded as marking the birth of the modern field of *public-key cryptography*[1]. DHKE remains massively relevant to the modern Internet, most notably as a key component of the Transport Layer Security protocol used extensively today. This protocol, as well as providing further security guarantees not directly related to privacy of the information, works much as we have described above: two parties not in possession of a shared key can establish one over an insecure channel using DHKE.

We have not yet discussed the mechanics of how DHKE is constructed. The protocol relies on so-called *discrete logarithms*, which appear to be difficult for a classical computer to recover. The two problems of interest are the *Discrete Logarithm Problem* (DLP), which, given a value and a base, is the problem of recovering the integer to which the base was raised

[1] More precisely the authors of [26] distinguish between *public key distribution systems*, which they claim DHKE is an example of, and public-key cryptosystems. For them, a public-key cryptosystem is a means of encrypting and decrypting directly - such a protocol would be invented a year later as the famous RSA cryptosystem [64]. For efficiency reasons, however, the modern practice is to use what Diffie and Hellman would call public-key cryptosystems as public key distribution systems; the latter term has now fallen out of use, and the former now refers to a much broader range of ideas.

to obtain the value; and the *Computational Diffie-Hellman Problem* (CDH). When our description of DHKE is translated to its discrete-logarithm based structure, a pair (a, A) is such that a is the discrete logarithm of A; so for a pair (a, A) one can recover a from A provided one can solve DLP. CDH, meanwhile, is the task of recovering[2] the resulting shared key from the values A and B. Since extracting a from A allows one to calculate $f(a, B)$ with the value B sent in the clear, one can solve CDH if one can solve DLP; the converse, however, is not known.

In other words, the assumption that DLP is difficult - where difficult in this thesis will almost always mean prohibitively time-consuming - underpins much of modern internet security protocols. In particular, some new technology or technique suddenly rendering DLP easy to solve would be disastrous: not only would security software such as TLS be totally vulnerable, but any attempt to distribute updates to security software would rely on protocols like TLS to guarantee their integrity. Considering the modern requirements of private citizens - online banking, email, and accessing medical records, to name a small sample - a sudden and severe compromise of modern cryptographic standards is a bleak picture indeed.

In 1994, Peter Shor demonstrated in his paper *Algorithms for quantum computation: discrete logarithms and factoring* [69] that this scenario is more plausible than we would hope. It was known already that DLP, factorising large numbers, and various other computational problems reduce to a period-finding problem: in [69], Shor demonstrates an efficient quantum algorithm for executing this period-finding problem. The technique relies on three inherently 'non-classical' aspects of a quantum computer: the ability to compute numerous values of a function 'at once'; the ability to manipulate such a state into a more desirable form by the *Quantum Fourier Transform*; and the constructive interference that occurs upon measurement of this more desirable form, returning the information of interest with high probability.

Today's quantum computers are nothing like powerful enough to threaten practical methods of modern cryptography: at time of writing, the largest quantum computer has 433 qubits, orders of magnitude fewer than the required number for large factoring or discrete logarithm operations. Nevertheless, there is clear motivation to develop quantum-

[2]The modern treatment is to define the security of DHKE in terms of how difficult is to distinguish the shared key from a random group element, rather than how difficult it is to recover the shared key, resulting in a third problem called the Decisional Diffie-Hellman problem, or DDH. We do not deal with this problem in this thesis.

resistant or *post-quantum* alternatives: in 2016 the National Institute for Standards in Technology (NIST), a branch of the NSA, announced plans to upgrade current standards in cryptography to quantum-resistant alternatives, citing fears that classical cryptography could be vulnerable as early as 2030 [17]. In 2022 the first candidates for standardisation were selected [52], largely based on computational problems in lattices and coding theory.

As we shall see later on in this thesis the announcement of finalists in the NIST standardisation process is by no means the end of post-quantum cryptography as a field. Indeed, spirited research continues in this area. In this thesis we are interesting in cryptographic protocols arising from computational problems in group theory: this area has received significantly less attention than its more mainstream peers, but is perhaps the most diverse of the significant players in the post-quantum landscape. This thesis examines computational problems induced by one of many protocols in group-based, post-quantum cryptography.

This concludes our summary of the context in which this research is set. It is time now to hone in more precisely on the protocol of interest, which is called Semidirect Product Key Exchange (SDPKE), and first appeared in 2013 [35]. We shall say little else about its definition until Chapter 2. Before the main body of the thesis can begin some housekeeping is in order: we now set out the structure of the thesis and highlight the main results, discuss the relationship between the thesis and the various papers it is assembled from, and give an array of preliminary material.

1.1 Organisation and Main Results

1.1.1 Research Questions

The original remit of this project was rather broad and sought to answer the following questions:

1. For which choices of group is SDPKE secure?
2. For which choices of group do schemes related to SDPKE have useful applications?

We answer the first question by providing a comprehensive survey of the difficulty of a problem related to the security of SPDKE, the so-called Semidirect Computational Diffie-Hellman problem, or SCDH. We consolidate and augment the state-of-the-art for solving SCDH in various

groups. The answer to to the second question arises in a rather unexpected fashion; that is, by proving that SDPKE is effectively one example of a larger and significantly better understood class of cryptographic protocols. As well as allowing us to define a signature scheme based on the structure inherent to SDPKE, this also gives us an estimate of the quantum complexity of solving the DLP analogue arising in SDPKE, about which virtually nothing was formerly known. In other words, we make significant progress in the complexity analysis of two computational problems in group-based post-quantum cryptography, and show one application of these problems.

A more detailed of the organisation of the thesis is as follows: in Chapter 2 we provide a comprehensive review of the state of the art with respect to SDPKE and its various cryptanalyses, both analytical and experimental, filling in several implicit gaps in the literature and augmenting with our own results. We treat the results primarily as addressing the complexity of SCDH, thereby correcting a persistent misconception in the literature that SDPKE cryptanalysis somehow does not address the relevant security problem. We also solidify the connection between the CDH problem arising from SDPKE and the rich field of representation theory. This chapter is a restructuring of [2, 3, 5] into a thesis chapter, with some results unique to this thesis.

In Chapter 3 we turn our attention to the DLP-type problem arising from SDPKE, called the Semidirect Discrete Logarithm Problem or SDLP, for which there are no known classical algorithms. We show that a careful rephrasing of the construction of SDPKE establishes a link to the theory of cryptographic group actions; after constructing a bespoke quantum algorithm to navigate some technicality specific to the choice of semigroup, we obtain a reduction to well-studied quantum algorithms of subexponential complexity. This time the chapter draws from a single paper - [5] - and makes significant use of the material within.

Finally, in Chapter 4 we use our new-found connection to cryptographic group actions to build the first signature scheme based on SDLP, and provide a careful security analysis thereof. Crucially, it turns out that a persistent issue with otherwise promising schemes derived from cryptographic group actions is partially solved by our novel group action: in particular, we make a step towards efficient sampling and therefore towards efficient, post-quantum signature schemes. This chapter is closely related to [6], though some material is omitted, re-organised or altered slightly.

1.1.2 Results

Table 1.1 lists the technical contributions made in this thesis and cross-references them with theorems in the various publications made during the PhD programme.

1.2 Preliminaries

1.2.1 Mathematical Background

Aside from the presentation of algorithms in pseudocode and the presentation of some experimental findings at the back of Chapter 2, this thesis proceeds according to the usual Definition-Theorem-Proof style of mathematical exposition. Some familiarity with the basics of group theory, linear algebra and proof technique is therefore required. Nevertheless, we highlight two definitions we will need repeatedly:

Definition 1.1. Let G be a finite group and X a finite set. A *group action* is a tuple $(G, X, \star)$ where $\star : G \times X \to X$, written $g \star x$ for $g \in G$ and $x \in X$, is such that

1. $1 \star x = x$ for all $x \in X$
2. $(gh) \star x = g \star (h \star x)$

Definition 1.2. An *algebra over a field* is a vector space $(V, +)$ over a field $\mathbb{F}$ equipped with a multiplication $\cdot$ satisfying the following: for $\lambda, \mu \in \mathbb{F}$ and $u, v, w \in V$ we have

1. $u \cdot (v + w) = u \cdot v + u \cdot w$
2. $(u + v) \cdot w = u \cdot w + v \cdot w$
3. $(\lambda u) \cdot (\mu v) = \lambda\mu(u \cdot v)$

Almost all of our examples of an algebra over a field will arise from a matrix ring $M_n(\mathbb{F})$; that is, $n \times n$ matrices with entry in a field $\mathbb{F}$. Clearly $M_n(\mathbb{F})$ is an n^2-dimensional vector space over $\mathbb{F}$; one can easily check that the usual notion of matrix multiplication defines a product satisfying the required properties.

Finally, we note that all logarithms are base 2.

Result	Summary	Other Appearances
2.11	Description and proof of validity of method of solving SCDH	Original idea (not the author's) in [65, 53], completion of missing detail unique to this thesis
2.13	Lower bound on complexity of dimension attack	Unique to this thesis; hinted at in the literature but made precise here
2.15	Using linear extension of group representation to group algebra representation to obtain a more favourable dimension for cryptanalysis	Essentially [3, Theorem 1] with more sophisticated presentation and applied differently
2.16	Conditions for injectivity of extension of group representation to group algebra representation	More sophisticated version of [3, Proposition 3]
2.18	For certain p-groups the dimension attack can do no better than $\mathcal{O}(p^2)$ complexity	Mentioned in [43], details filled in here
2.22	Conditions for mapping matrices over a non-commutative ring into matrices over a commutative ring	[3, Theorem 1], essentially the same as 2.15 in this thesis
3.6	Structure of SDPKE defines a group action	[4, Theorem 3]
3.12	Quantum algorithm for recovering the two parameters associated to SDPKE structure	[4, Theorem 4]
3.18	Algorithm to solve SDLP	[4, Theorem 9]
4.3	Enforcing invertibility defines a different group action structure now dependent on only one parameter	[6, Theorem 3]
4.4	The value of this parameter is restricted to a relatively small set of possible values	[6, Theorem 4]
4.16	A digital signature based on the group action is secure with respect to SDLP	[6, Theorem 7]
4.18	One can efficiently compute the appropriate parameter for a choice of group action	[6, Theorem 9]

Table 1.1: Technical contributions made in this thesis

1.2.2 Diffie-Hellman Key Exchange

We have made extensive reference to the Diffie-Hellman Key Exchange (DHKE) protocol thus far; it is sufficiently important to warrant a more thorough examination.

Definition 1.3 (Diffie-Hellman Key Exchange)**.** Suppose two parties - traditionally called Alice and Bob - agree on a finite cyclic group G of order n, and a generator g of G. They can arrive at a shared group element as follows:

- Alice picks a random integer $a \in \{1, ..., n\}$, which she keeps secret, and calculates $A \leftarrow g^a$ (here the exponentiation refers to repeated application of the group operation). She sends this latter value to Bob.
- Bob similarly calculates group element $B \leftarrow g^b$ from his secret random integer b and sends this to Alice.
- Upon receipt of g^b Alice uses her private exponent to calculate $K_A \leftarrow (g^b)^a$; similarly, Bob calculates $K_B \leftarrow (g^a)^b$. Since $g^{ab} = g^{ba}$ we have key agreement.

With respect to the discussion above the motivation for recording the following computational problems is clear.

Definition 1.4 (Discrete Logarithm Problem)**.** Let G be a finite cyclic group generated by $g \in G$. Given g, g^x for x sampled uniformly from $\{1, ..., |G|\}$, the *Discrete Logarithm Problem* (DLP) is to recover x.

Definition 1.5 (Computational Diffie-Hellman Problem)**.** Let G be a finite cyclic group generated by $g \in G$. Given g, g^x, g^y for x, y sampled uniformly from $\{1, ..., |G|\}$, the *Computational Diffie-Hellman Problem* (CDH) is to recover g^{xy}.

1.2.3 Complexity

This thesis makes frequent use of the standard 'big O' notation, which we briefly recall: two functions f and g are such that $f = \mathcal{O}(g)$ if there is a real constant $C > 0$ and some $N \in \mathbb{N}$ for which all $n \geq N$ have $f(n) \leq Cg(n)$. Complexity is given either in terms of the number of elementary operations required or the number of arithmetic operations required, depending on which most naturally highlights the main parameter with

which complexity grows. We will always note when the number of arithmetic operations is being estimated.

In more detail, by an elementary operation we mean a single digit addition or multiplication - since we consider integers in terms of their binary representation we may just as well say that we mean the binary AND and XOR operations. Our basic assumption is that for elements in a finite field of size N, each element by definition has binary representation of length $\log N$. Long Multiplication therefore gives their product in time $\mathcal{O}((\log N)^2)$, and one calculates the residue modulo N for a final answer by long division, also in time $\mathcal{O}((\log N)^2)$. In other words, we assume that multiplication in a finite field of size N requires $\mathcal{O}((\log N)^2)$ elementary operations. If the process under consideration requires a certain number of operations in a field, but the size of this field is independent of the parameter in terms of which we are giving complexity estimates, the complexity of field operations can be thought of as a constant. In this case we can give complexity in terms of the number of arithmetic operations required without losing much insight into the complexity in terms of elementary operations.

1.2.4 Quantum Computation

The following summary of key concepts in the theory of quantum computation is assembled from [56, 37, 63].

A *qubit* (for our purposes) is a 2-dimensional vector space H_2 over $\mathbb{C}$. We will always use the so-called *computational basis* $\begin{bmatrix}1\\0\end{bmatrix}$, $\begin{bmatrix}0\\1\end{bmatrix}$ - these *basis states* are written by convention as $|0\rangle, |1\rangle$ respectively. A *state* of the qubit is unit-length vector $\alpha|0\rangle + \beta|1\rangle$; that is, one has $|\alpha|^2 + |\beta|^2 = 1$. If α, β are both non-zero the basis states are in *superposition*. A key idea is that one can *observe* this qubit, collapsing the superposition and seeing 0 with probability α and 1 with probability β.

We can represent a system of n qubits by the complex vector space H_{2^n}, where the basis states are exactly the n-fold tensor products of basis states of H_2. An ordered system of n qubits is called a *quantum register of length n*, and the basis states are sometimes written $\{|i\rangle : 0 \leq i < 2^n\}$ by identifying i with its binary representation.

We say a state $\mathbf{z} \in H_{2^n}$ is *entangled* if it cannot be written as the tensor product of H_2 states. In particular, an observation of one qubit in an entangled state affects the state of the other: suppose integers $\{0, ..., M-1\}$ can be represented by a quantum register of length l for

some $l, M \in \mathbb{N}$, and moreover that a function f on $\{0, ..., M-1\}$ is such that $\{f(0), ..., f(M-1)\}$ can be represented similarly. A state of the form

$$\frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} |j\rangle |f(j)\rangle$$

is such that when observation of the second register gives some $Y \in \{f(0), ..., f(M-1)\}$, the first register is left in superposition

$$\frac{1}{\sqrt{L}} \sum_{j:f(j)=Y} |j\rangle$$

where L is the number of $j \in \{0, ..., M-1\}$ such that $f(j) = Y$. The factor $1/\sqrt{L}$ is to normalise the probabilities, ensuring the state is a unit vector.

Finally, recall that we can create the uniform superposition efficiently with a Hadarmard gate. For an l-qubit register, the computational basis vector $|0\rangle$ is such that the Hadamard gate (written H_{2^l}) has

$$H_{2^l} |0\rangle = \frac{1}{\sqrt{2^l}} \sum_{i=0}^{2^l-1} |i\rangle$$

Moreover, this transformation can be carried out efficiently, in time $\mathcal{O}(l)$.

Circuits

Implementing functions on a quantum computer is usually described in terms of *circuits*; that is, for positive integers n, m a function $F : \{0,1\}^n \to \{0,1\}^m$ is thought of as executed by a series of elementary operation gates. It turns out that one can describe all functions of this type in terms of AND, OR or NOT gates. As a relatively harmless heuristic we will assume that the number of elementary operations required to compute the function F is the same as the number of gates required to construct a circuit computing F.

For reasons outside the scope of this thesis all the computations executed by a quantum computer must be *reversible*; that is, a quantum computer cannot destroy information. We therefore cannot use some standard gates when describing a quantum circuit - for example, the AND gate is not reversible. It turns out, however, that one can implement any circuit with reversible gates without much extra cost: by [63, Section 6.2.2] for any $\epsilon > 0$ one can construct a circuit implementing a

function F on N gates using $\mathcal{O}(N^{1+\varepsilon})$ elementary reversible gates. Together with our heuristic on the relationship between the number of gates in a circuit and the time complexity of computing the associated function, we will assume that a quantum computer can compute a function in at least the same time as a classical computer.

2 Platforms for the Semidirect Product Key Exchange

God has given you one face,
and you make yourself another.

Hamlet, Act III

This chapter deals with the various instances of the so-called Semidirect Product Key Exchange, or SDPKE. More specifically, we analyse the difficulty of the underlying security problem for a variety of different choices of mathematical object, giving a comprehensive survey of the state of the art; we also offer a critique of the works in this area, and fill in some of the technical gaps that have been implicitly left by these papers.

2.1 Motivation

We have already seen that the Diffie-Hellman key exchange of [26] is vulnerable to the quantum methods developed in [69], and that an attempt to develop new quantum-resistant, or post-quantum, alternatives is well underway. There are two more nuanced motivating factors setting out the niche that SDPKE occupies.

Firstly, we note that that for security purposes it is important to develop a landscape of post-quantum schemes based on a rich variety of computational problems, for obvious reasons - a resolution to one class of computational problems should not imply the breakage of all post-quantum cryptography. Unfortunately, this is not currently the case: the only key encapsulation mechanism recommended for standardisation by NIST is based on lattices, and the fourth round candidates are either code-based or isogeny based - and the isogeny-based key encapsulation mechanism, SIKE[1] [42], was shown to admit efficient secret key recovery

[1]Supersingular Isogeny Key Exchange.

in [14, 48] (all of this information can be found at [59, 55]). The situation with digital signature schemes is even worse - the multivariate-based digital signature scheme, Rainbow [27], was shown to admit forgeries with the recommended parameter sets in [8]. All the remaining candidate digital signature schemes are based on lattice problems.

Secondly, the NIST standardisation process does not feature any non-interactive key exchanges, which have roughly the following advantage. Recall in the description of Diffie-Hellman Key Exchange we described two parties sending each other an exchange value in order to facilitate shared key agreement. The idea of a non-interactive key exchange is that we can think of Alice's pair (g, g^a) as a private key - public key pair: in particular, if g^a is published, Bob can compute g^{ab} without Alice having sent any messages. If Bob then encrypts under this shared key and attaches to his ciphertext some identification material, Alice can also derive g^{ab} by checking Bob's published public key g^b. Again, Bob has not had to send Alice any additional messages in order to accomplish this. A closer look at the relevant formalisation of such schemes can be found in [30] - for our purposes, we mean a method of establishing a shared key that has the syntax of Diffie-Hellman key exchange. The important distinction is that under this syntax key establishment can occur without both parties needing to be online at the same time, which has applications, for example, in wireless networks where one needs to conserve battery power.

For these reasons it is desirable to seek a diverse roster of non-interactive key exchanges based on a variety of computational problems. SDPKE is an example of a non-interactive key exchange defined so as to be based on a problem arising from the lesser-known field of group-based cryptography. Before getting to its definition, however, we must first develop some of the appropriate algebraic machinery.

2.2 The Semidirect Product

Definition 2.1. Let G be a finite semigroup and $End(G)$ be its endomorphism semigroup. The *semidirect product* of G by $End(G)$, written $G \ltimes End(G)$ consists of the set of ordered pairs $G \times End(G)$ equipped with multiplication defined by

$$(g, \phi)(h, \psi) = (\psi(g)h, \psi\phi)$$

where $\psi\phi$ refers to the function obtained by first applying ϕ, then ψ.

If G is indeed a semigroup - that is, at least one of its elements does not have an inverse - the resulting structure $G \ltimes End(G)$ is itself a semigroup. On the other hand, if we allow for invertibility, we do indeed get a group - the following is standard and lightly adapted for our specific notation.

Theorem 2.2. *Let G be a finite semigroup. $H = G \ltimes End(G)$ is also a semigroup, and if G is a full group, $G \ltimes Aut(G)$ is a full group.*

Proof. $(1, \mathrm{id.})$ is the identity. For associativity, let $(p, \phi), (q, \psi), (r, \omega)$ be elements of H; then doing the calculations one has

$$\begin{aligned}((p,\phi)(q,\psi))\,(r,\omega) &= (\psi(p)q, \psi\phi)\,(r,\omega)\\ &= (\omega\psi(p)\omega(q)r, \omega\psi\phi)\\ &= (p,\phi)\,(\omega(q)r, \omega\psi)\\ &= (p,\phi)\,((q,\psi)(r,\omega))\end{aligned}$$

Finally, if G is a full group, for any $(g, \phi) \in G \ltimes Aut(G)$ we have

$$(g,\phi)(\phi^{-1}(g^{-1}), \phi^{-1}) = (\phi^{-1}(g^{-1}), \phi^{-1})(g,\phi) = (1, \mathrm{id.}),$$

and so we are done. □

In the exposition for this chapter the setup of a finite group and its automorphism group, and a finite semigroup and its endomorphism semigroup, are used more or less interchangeably in the general case. Later on we will see specific examples of groups and semigroups where invertibility is explicitly required or omitted; but these should be clear from context. Indeed, in the general case we prefer the definitions in terms of a finite semigroup and its endomorphism semigroup.

2.3 Semidirect Product Key Exchange

In Chapter 4 we will see in closer detail the effect of allowing for invertibility. For now, we are not particularly concerned with the semidirect product itself, but instead the following quantity which it gives rise to:

Definition 2.3. Let G be a finite semigroup and $End(G)$ be its endomorphism semigroup. Each pair $(g, \phi) \in G \ltimes End(G)$ induces the function $s_{g,\phi} : \mathbb{N} \to G$, where for each $x \in \mathbb{N}$, $s_{g,\phi}(x)$ is defined as the element of G such that

$$(g,\phi)^x = (s_{g,\phi}(x), \phi^x)$$

It is not difficult to check that $s_{g,\phi}(x) = \phi^{x-1}(g)...\phi(g)g$. The key insight that makes this quantity of cryptographic interest, however, is the following:

$$\begin{aligned}(s_{g,\phi}(x+y), \phi^{x+y}) &= (g,\phi)^{x+y} \\ &= (g,\phi)^x(g,\phi)^y \\ &= (s_{g,\phi}(x), \phi^x)(s_{g,\phi}(y), \phi^y) \\ &= (\phi^y\left(s_{g,\phi}(x)\right) s_{g,\phi}(y), \phi^{x+y})\end{aligned}$$

It follows that $s_{g,\phi}(x+y) = \phi^y(s_{g,\phi}(x))s_{g,\phi}(y)$. An entirely symmetrical argument shows that $s_{g,\phi}(x+y) = \phi^x(s_{g,\phi}(y))s_{g,\phi}(x)$; in other words, given $s_{g,\phi}(x)$, we can calculate $s_{g,\phi}(x+y)$ with knowledge only of y, and vice versa.

This insight is sufficiently fundamental that we codify it in a theorem.

Theorem 2.4. *Let G be a semigroup and $End(G)$ be its endomorphism semigroup. For each $(g,\phi) \in G \ltimes End(G)$ and $x, y \in \mathbb{N}$, we have*

$$\phi^x(s_{g,\phi}(y))s_{g,\phi}(x) = s_{g,\phi}(x+y) = \phi^y(s_{g,\phi}(x))s_{g,\phi}(y)$$

It is precisely these equalities that allows the definition of SDPKE in [35][2]:

Definition 2.5 (Semidirect Product Key Exchange)**.** Suppose two parties Alice and Bob agree on a finite semigroup G, its endomorphism semigroup $End(G)$, and a pair $(g,\phi) \in G \ltimes End(G)$. Let $N = |\{s_{g,\phi}(i) : i \in \mathbb{N}\}|$ - we know N is finite since each value of $s_{g,\phi}(i)$ is contained in G, which is itself finite. The two parties can arrive at a common group element as follows:

- Alice chooses an integer x at random from $\{1,...,N\}$ and calculates $A = s_{g,\phi}(x)$. She sends this value to Bob.
- Bob chooses an integer y at random from $\{1,...,N\}$, calculates $B = s_{g,\phi}(y)$, and sends it to Alice.
- Upon receipt of Bob's value B, Alice uses her integer x to calculate $K_A := \phi^x(B)s_{g,\phi}(x)$.
- Bob similarly uses his integer y to calculate $K_B = \phi^y(A)s_{g,\phi}(y)$.

[2]The notation used in this thesis does not reflect that of [35], which deals with the explicit form of the group element defined by $s_{g,\phi}(i)$.

Remark. The notation employed for our presentation of SDPKE is non-standard, and certainly is not used by any of the proposed instances of the scheme. Its closest cousin is the notation in the cryptanalytic work [65], whereby a_i is defined as our $s_{g,\phi}(i)$. The similarity is perhaps because [65] is one of few works in the literature to consider the set of all possible exchange values, although this work does not discuss the finite quantity N denoting the size of this set. Indeed, all discussions of an instance of SDPKE in the literature make reference only to selecting some positive integer at random without specifying the bounds that this integer should be specialised within, largely because the goal of these works is to address some vulnerability in a previously proposed platform.

We also note that our framing of the quantity N invites the question of the value of $s_{g,\phi}(i)$ when $i > N$. For now it suffices to note that such a value has one of the N distinct values already outlined, but the resulting structure is discussed in much more detail in Chapter 3.

We have seen that $K = K_A = K_B$ in SDPKE, so indeed we have correctness. Before we go on to discuss the efficiency and security of these schemes, for convenience let us standardise some terminology.

Definition 2.6. Consider the key exchange scheme presented in Definition 2.5. We call the semigroup $G \ltimes End(G)$ the *platform* for the key exchange, the pair (g, ϕ) the *base pair*, and each value $s_{g,\phi}(i)$ an *exchange value*[3]. For a pair of integers x, y selected by Alice and Bob to output a value $s_{g,\phi}(x + y)$, we call x, y the *private keys* and $s_{g,\phi}(x + y)$ the *shared key*.

2.3.1 Efficiency

For our purposes, and certainly in this chapter, the relevance of the semidirect product structure is chiefly the ability to calculate $s_{g,\phi}(i)$. In particular, we can use the standard technique of square-and-multiply as follows:

Theorem 2.7. *Let G be a finite semigroup and $(g, \phi) \in G \ltimes End(G)$. Suppose that j is the largest integer such that $i > 2^j$. Algorithm 1 takes as input (g, ϕ) and an integer $i \in \mathbb{N}$, and returns the G-element $s_{g,\phi}(i)$.*

Correctness of the algorithm follows from the observation that for any $x \in \mathbb{N}$, $(g, \phi)^x = ((g, \phi)^{x/2})^2$ if x is even, and $((g, \phi)^{(x-1)/2})^2(g, \phi)$ if x is

[3] If we think of each party as publishing their exchange value such that key agreement can be established non-interactively, we might also call these values *public keys*.

Algorithm 1 Calculating $s_{g,\phi}(i)$ given g, ϕ, i

Input: (g, ϕ), integer i
Output: $s_{g,\phi}(i)$
1: $\{b_1, ..., b_j\} \leftarrow$ binary expansion of i
2: $r \leftarrow (g, \phi)$
3: **for** $1 \leq k \leq j$ **do**
4: $\quad r \leftarrow r^2 r^{b_k}$
5: **end for**
6: **return** r

odd. In order to calculate $s_{g,\phi}(i)$ one therefore has to carry out $\mathcal{O}(\log i)$ operations in the semigroup, which is significantly better than the naive approach demanding i such operations. Moreover, if we suppose that applying ϕ has about the same complexity - up to constants, say - as the semigroup operation in G (which is the case in [35, 34, 43]), calculating the shared key from the other party's exchange value $s_{g,\phi}(y)$ and the integer x has similar complexity to that of calculating $s_{g,\phi}(x)$. This is because we get the endomorphism ϕ^x for free from the calculation of our own exchange value $s_{g,\phi}(x)$, whence calculation of $\phi^x(s_{g,\phi}(y))s_{g,\phi}(x)$ requires some fixed constant number of additional applications of the group operation.

In other words, the number of operations required to execute SDPKE with respect to a semigroup G is a logarithmic factor worse than the number of operations required to carry out the semigroup operation in $G \ltimes End(G)$. Eschewing the details for now we conclude that the key exchange is 'efficient' provided multiplication in the underlying group is efficient.

2.3.2 Security

As discussed in the introduction, we analyse the *security* of SDPKE as the computational difficulty of computing the shared key value $s_{g,\phi}(x + y)$. Before we go on let us define more precisely what we mean by this.

Definition 2.8. Let G be a finite semigroup and let $(g, \phi) \in G \ltimes End(G)$. Suppose Alice and Bob have public keys $A \leftarrow s_{g,\phi}(x), B \leftarrow s_{g,\phi}(y)$. An adversary passively eavesdropping $(g, \phi), A, B$ recovers the shared key if they can compute $s_{g,\phi}(x + y)$.

Note that our definition allows for an adversary eavesdropping an

interactive round of SDPKE, or an adversary recovering published public keys as in the non-interactive key exchange scenario.

By now it should be clear that SDPKE and the classical Diffie-Hellman Key Exchange are rather closely related: notice that if $G = \mathbb{Z}_p$ for some prime p, and ϕ is chosen as the identity map, we have DHKE as a special case of SDPKE. The analogy is continued in the computational problems underpinning their security - that is, the discrete logarithm problem and the related computational Diffie-Hellman problem have extremely natural analogues[4].

In each of the following let G be a finite semigroup and $End(G)$ its endomorphism semigroup. For a pair $(g, \phi) \in G \ltimes End(G)$, consider the integer $N = |\{s_{g,\phi}(i) : i \in \mathbb{N}\}|$. Each of the following problems should be thought of as corresponding to the specific choice of (g, ϕ).

Definition 2.9 (Semidirect Discrete Logarithm Problem (SDLP))**.** Given (g, ϕ), and a value $s_{g,\phi}(x)$ for some x selected at random from $1 \leq x \leq N$, one *solves the Semidirect Discrete Logarithm Problem* if one can recover x.

Certainly one can efficiently achieve key recovery in SDPKE if one can efficiently solve SDLP, but the converse is not clear *a priori* - that is, there is no obvious way to convert an SDPKE key-recovery algorithm into an SDLP solver. Similarly to the classical case we define the following problem to address this gap.

Definition 2.10 (Semidirect Computational Diffie-Hellman Problem (SCDH))**.** Given (g, ϕ) and the values $s_{g,\phi}(x), s_{g,\phi}(y)$ for integers x, y selected at random from $1 \leq x, y \leq N$, one *solves the Semidirect Computational Diffie-Hellman problem* if one can recover the value $s_{g,\phi}(x + y)$.

It should be noted that solving SCDH is exactly the problem of recovering the shared key in SDPKE. With respect to a classical adversary, we immediately have that SDLP is at least as hard as SCDH, since one can calculate $s_{g,\phi}(x + y)$ by using an SDLP oracle to recover the integers x and y. However, as we will see in the remainder of this chapter, there are several examples of a semigroup G for which SCDH can be solved in polynomial time, but no efficient algorithm for SDLP is known. Note that this is in contrast to the classical case: the original assertion in [26] that there does not appear to be a method of key recovery in DHKE that does

[4]A notable omission is that of the natural decision variant of the problem, about which there has been little study - although it is worth noting that statistical evidence is provided in [35] suggesting that this problem may indeed be hard.

not involve solving the classical DLP has held true, in that indeed no examples of groups for which DHKE is vulnerable but the DLP is hard have been found. In our setting, there are examples of semigroups for which one can efficiently achieve key recovery, but no efficient solution to SDLP is known - that is, SDLP seems to be a strictly harder problem than SCDH.

2.4 Key Recovery for SDPKE

For the remainder of this chapter we will see some of the strategies mentioned above for solving SCDH in various groups. As alluded to in the introduction our goal is to systematise the various approaches; before we do so, let us review the current list of platforms proposed.

2.4.1 SDPKE Platforms

In line with [2] we list the proposed platforms in chronological order. Indeed, since in the literature a new proposal of platform is directly a response to some cryptanalytic idea on a previous platform, this will also serve as the motivation for selecting rather arbitrary-looking semigroups as a platform.

The first semigroup proposed for use with SDPKE appears in the original proposal of the key exchange [35]. The authors platform semigroup $M_3(\mathbb{Z}_7[A_5])$ under matrix multiplication, with a base pair automorphism defined as conjugation by some invertible matrix in the semigroup. Here, $\mathbb{Z}_7[A_5]$ denotes[5] the group ring consisting of formal sums of the form

$$\sum_{g \in A_5} a_g . g \quad a_g \in \mathbb{Z}_7.$$

One can define a notion of addition and multiplication on this ring; equipped with these operations we get a ring that is at the same time an $|A_5|$-dimensional algebra over $\mathbb{Z}_7$.

Notice that we can also consider this same set of matrices under addition, rather than multiplication. As a consequence of the more general method put forward in [53], we can use this fact as the following observation of [65]: if the set of all exchange values $\{s_{g,\phi}(i) : i \in \mathbb{N}\}$ occurs as a subset of an algebra over a field, one can use Gaussian elimination to find a maximal linearly independent subset of the exchange

[5] We will see a more formal definition in Section 2.7.

values. A combination of equation solving and application of Theorem 2.4 allows for recovery of the shared key.

Such an attack runs in time polynomial in the dimension of the algebra which the exchange values occur in. The strategy of [65] is to take a group representation; that is, an injective homomorphism into a matrix group with entries in a finite field, which can also be considered under addition. Provided that the accompanying choice of automorphism in the base pair preserves this notion of addition, we can carry out our Gaussian elimination in this larger space and invert the result. Since $M_3(\mathbb{Z}_7[A_5])$ embeds in a low-dimensional algebra over a field[6] one can achieve key recovery efficiently.

If, on the other hand, there exists groups for which the only injective homomorphisms are into prohibitively high-dimensional vector spaces the attack would be ineffective. Relying on a result of Janusz [41], this is precisely the approach of [43], in which the following platform is proposed: consider the free group on r elements F_r. The normal subgroup F_r^p is generated by all elements of the form g^p for $g \in F_r$ and some prime p. With notation $[a, b]$ denoting the product $a^{-1}b^{-1}ab$ and $[[a_1, ..., a_n], a_{n+1}]$ defined inductively (that is, $[a_1, a_2, a_3] = [a_1, [a_2, a_3]]$), the normal subgroup generated by all elements of the form $[[a_1, ..., a_{c-1}], a_c]$ is denoted $\gamma_{c+1}(F_r)$. Assembling these components the finite group $F_r / F_r^{p^2}.\gamma_c(F_r)$ is given as the recommended platform - crucially, this group has an element of order p^2, which we will see later on guarantees that only high-dimensional representations are admissible.

Another approach taken to avoid these types of attacks is to insist the platform group occurs as the additive group formed by an algebra under addition. Since (as we will see) the computation of the shared key relies on the distributive property of multiplication in the algebra, the vulnerability to the type of attack mentioned above is eliminated. This approach is taken in [61][7] by proposing $M_3(\mathbb{Z}_p)$ under addition as a platform group. The resulting protocol is dubbed MAKE, which stands for 'a Matrix Action Key Exchange'.

[6] The situation is in fact slightly more subtle than the claim of [65], in which it is claimed that the native additive structure of this group suffices. Since Gaussian elimination is only guaranteed to work in a vector space, rather than a module, this will not quite suffice, and in fact to complete the cryptanalysis we will later introduce a technique from significantly later on in the literature, chronologically speaking.

[7] It is worth noting that this work appears after a several year gap in the literature, and that the authors do not claim to have taken the new approach for the reasons we describe.

It turns out, however, that a method of linear decomposition similar in spirit to that of [65] is available for additive groups as well, as demonstrated in [13]. One of the key pieces of data required for this approach is the value $\phi^x(g)$, which can be recovered uniquely in the case of [61]. Motivated by a desire to make the relevant quantity difficult to recover, and to prevent the kind of embedding in an algebra we have already seen is problematic[8], the most recently proposed platform for use with SDPKE is that found in the MOBS ('Matrices Over Bit Strings') cryptosystem [60]. Roughly speaking, matrices over a semiring are employed; since we only need to be able to add and multiply in the underlying ring to define matrix multiplication, we get a multiplicative semigroup.

We note also that the platform variant defined in [34] is not covered by the chronological exposition given above, largely because the proposal and resulting cryptanalysis [39, 67] are largely self-contained. With this in mind, this branch of the literature shall not be further discussed.

In summary, we will discuss the security of the following four platforms:

- The semigroup of matrices $M_3(\mathbb{Z}_7(A_5))$ under matrix multiplication defined by arithmetic in $\mathbb{Z}_7(A_5)$
- The finite p-group defined by the quotient of a free group r generators
- The abelian group $M_3(\mathbb{Z}_p)$ of matrices considered under addition
- The semigroup formed by taking matrices over a semiring, with the semigroup operation defined by matrix multiplication with respect to the arithmetic in the semiring.

Recalling our discussion Section 2.3.2, *security* for us means the difficulty of SCDH in these semigroups. Let us now discuss the key strategies for solving SCDH.

2.5 The Dimension Attack

We now turn our attention to the so-called *linear decomposition attack*, or the *dimension attack*, put forward in [53] and given with specific attention to our case in [65]. Indeed, for the most part we will be making reference

[8] Though as we will see this is not an explicitly stated goal of the work.

to [65] since its first half is devoted to a method of solving SCDH - though this is not how the paper is introduced, and indeed a small critique of the work is necessary in order to best frame its contents.

Essentially, the contribution of [65] is to show that if a platform semigroup occurs as the semigroup of an algebra over a field considered under multiplication, one can use the underlying linearity of the algebra to recover the shared key, effectively by Gaussian elimination. The claim in [65] that "...one does not need to solve the underlying algorithmic problem to break the scheme" is a misrepresentation of this important result's place in the literature: as we shall see (although this is not specifically claimed) a very general method of solving SCDH is given - but in the first appearance of SDPKE in [35], the security is explicitly given as exactly SCDH. It is true, however, that one need not necessarily solve SDLP to achieve shared key recovery - note that this is in contrast to classical DHKE.

The generality of the method of [65] comes from the ability to embed any finite group in an algebra. This is rather vaguely defined in the literature - the original method assumes that the semigroup is defined as an algebra under multiplication, but in [43] the idea of any choice of group being in theory susceptible to this attack is implicitly mentioned. The following result is principally indebted to the arguments set out in [65, Section 2] - with respect to the discussion above, we fill in most of the missing details.

Theorem 2.11 (Dimension Attack)**.** *Let G be a finite semigroup and fix $(g, \phi) \in G \ltimes End(G)$. Suppose there is an injective homomorphism $r : G \to \mathbb{A}$, where $\mathbb{A}$ is a finite dimensional algebra over a field, and that there exists some linear map $T : \mathbb{A} \to \mathbb{A}$ such that $T \circ r = r \circ \phi$. There is a deterministic algorithm to recover $s_{g,\phi}(x + y)$ running in time polynomial in the dimension of $\mathbb{A}$.*

Proof. Suppose $|\{s_{g,\phi}(i) : i \in \mathbb{N}\}| = N$ for some $N \in \mathbb{N}$, and for randomly selected integers $1 \leq x, y \leq N$ we are given the values $s_{g,\phi}(x), s_{g,\phi}(y)$. The following method computes $s_{g,\phi}(x + y)$.

The first step is to find a basis for the set $\{r(s_{g,\phi}(1)), ..., r(s_{g,\phi}(N))\}$. The ability to do so is a special case of the powerful result [53, Lemma 3.1]; we give a case-specific version of their constructive method. For each $i \geq 2$ use Gaussian elimination to check if $L_i = \{r(s_{g,\phi}(j)) : 1 \leq j \leq i\}$ is linearly independent. If it is, increment the value of i and continue; if not, the procedure outputs L_{i-1}. The result is a set $\{r(s_{g,\phi}(1)), ..., r(s_{g,\phi}(k))\}$ for some $k \in \mathbb{N}$ that is linear independent, but is such that $\{r(s_{g,\phi}(1)), ..., r(s_{g,\phi}(k+$

$1))\}$ is not linear independent. Without loss of generality we can write $r(s_{g,\phi}(k+1))$ as a linear combination in the underlying field of the values in $\{r(s_{g,\phi}(1)), ..., r(s_{g,\phi}(k))\}$; we claim that every $r(s_{g,\phi}(i))$ has this property.

To see this, first note that the result holds trivially for $1 \leq i \leq k$. For $i > k$, suppose $i = k + t$ for some $t \in \mathbb{N}$. Suppose moreover for induction that the claim holds for each $k + j$, where $1 \leq j \leq t - 1$. A lengthy series of calculations shows that

$$\begin{aligned} r(s_{g,\phi}(k+t)) = r(\phi(s_{g,\phi}(k+t-1))g) &= r(\phi(s_{g,\phi}(k+t-1)))r(g) \\ &= T(r(s_{g,\phi}(k+t-1)))r(g) \\ &= T\left(\sum_{i=1}^{k} \lambda_i r(s_{g,\phi}(i))\right) r(g) \\ &= \left(\sum_{i=1}^{k} \lambda_i T(r(s_{g,\phi}(i)))\right) r(g) \\ &= \sum_{i=1}^{k} \lambda_i r((\phi(s_{g,\phi}(i)))r(g) \\ &= \sum_{i=1}^{k} \lambda_i r(\phi(s_{g,\phi}(i))g) \\ &= \sum_{i=1}^{k} \lambda_i r(s_{g,\phi}(i+1)) \end{aligned}$$

In other words, renaming constants we have

$$r(s_{g,\phi}(k+t)) = \lambda_k r(s_{g,\phi}(k+1)) + \sum_{i=2}^{k} \mu_i r(s_{g,\phi}(i))$$

The claim follows by noticing that since $r(s_{g,\phi}(k+1))$ is a linear combination of the appropriate form, so is $\lambda_k r(s_{g,\phi}(k+1))$. It remains to compute the value $s_{g,\phi}(x+y)$ - since r is injective it suffices to compute $r(s_{g,\phi}(x+y))$, which we can do with our linear independent subset by similar arguments to the above.

Starting with $r(s_{g,\phi}(x))$ (although either value will do by symmetry), we know that $r(\phi^y(s_{g,\phi}(x))s_{g,\phi}(y)) = r(s_{g,\phi}(x+y))$. The strategy is to use our linear decomposition of $r(s_{g,\phi}(x))$ to 'swap' all the unknown y exponents for the small exponents in our linear independent set, so that we need only use the known value of $r(s_{g,\phi}(y))$. With this in mind, using

Gaussian elimination we can find field coefficients such that

$$r(s_{g,\phi}(x)) = \sum_{i=1}^{k} \nu_i r(s_{g,\phi}(i))$$

Notice that

$$r \circ \phi^2 = T \circ (r \circ \phi) = T^2 \circ r$$

so by induction that $r \circ \phi^y = T^y \circ r$. Since the composition of a linear map with itself is also a linear map, we get the following:

$$\begin{aligned}
r(s_{g,\phi}(x+y)) &= r(\phi^y(s_{g,\phi}(x))s_{g,\phi}(y)) \\
&= T^y(r(s_{g,\phi}(x)))r(s_{g,\phi}(y)) \\
&= T^y\left(\sum_{i=1}^{k} \nu_i r(s_{g,\phi}(i))\right) r(s_{g,\phi}(y)) \\
&= \left(\sum_{i=1}^{k} \nu_i T^y(r(s_{g,\phi}(i)))\right) r(s_{g,\phi}(y)) \\
&= \left(\sum_{i=1}^{k} \nu_i r(\phi^y(s_{g,\phi}(i)))\right) r(s_{g,\phi}(y)) \\
&= \sum_{i=1}^{k} \nu_i r(\phi^y(s_{g,\phi}(i))s_{g,\phi}(y)) \\
&= \sum_{i=1}^{k} \nu_i r(\phi^i(s_{g,\phi}(y))s_{g,\phi}(i))
\end{aligned}$$

Since we know the value of each ν_i, and of $r(s_{g,\phi}(y))$, we can use this equality to calculate $r(s_{g,\phi}(x+y))$ and therefore $s_{g,\phi}(x+y)$. In summary, Algorithm 2 computes $s_{g,\phi}(x+y)$. □

Before moving on to analysing the complexity of the attack a few remarks are in order. Recall that in the original presentation of these ideas the platform group is assumed to occur directly as the multiplicative semigroup formed by an algebra under multiplication. In that setting, the application of the techniques described above relies on the assumption that the base homomorphism ϕ also preserves the native additive structure of the algebra. Our requirement that some linear map T is such that $T \circ r = r \circ \phi$ effectively generalises this assumption.

Interestingly, as highlighted in Algorithm 2, at no point is one required to evaluate the function T. In other words one does not require an explicit

Algorithm 2 Computing $s_{g,\phi}(x+y)$ with Gaussian elimination.

Computing the basis set.
Input: (g, ϕ), map r
Output: basis of $\{r(s_{g,\phi}(i)) : i \in \mathbb{N}\}$

1: $L \leftarrow \{r(s_{g,\phi}(1)), r(s_{g,\phi}(2))\}$
2: $i \leftarrow 2$
3: **while** L is linear independent **do**
4: $\quad i \leftarrow i + 1$
5: $\quad L \leftarrow L \cup r(s_{g,\phi}(i))$
6: **end while**
7: **return** $L \setminus \{r(s_{g,\phi}(i))\}$

Recovering $s_{g,\phi}(x+y)$
Input: $(g, \phi), s_{g,\phi}(x), s_{g,\phi}(y)$, map r, basis of $\{r(s_{g,\phi}(i)) : i \in \mathbb{N}\}$
Output: $s_{g,\phi}(x+y)$

1: $k \leftarrow |L|$
2: $(\nu_1, ..., \nu_k) \leftarrow$ coefficients of $r(s_{g,\phi}(x))$ as linear combination of L-elements
3: $K \leftarrow \sum_{i=1}^{k} \nu_i r(\phi^i(s_{g,\phi}(y)) s_{g,\phi}(i))$
4: **return** $r^{-1}(K)$

description of T - an existence proof that such a linear map exists for each choice of base homomorphism ϕ would suffice. We do not currently have access to a result of this type, and leave it for further work.

On the other hand one does need an explicit description of the function r, since we compute its evaluation on exchange values frequently. It is not immediately obvious that an appropriate r exists for each semigroup G, but in fact this is the case provided G is finite. Let us demonstrate this now, keeping in mind that we are effectively presenting a corollary to Cayley's theorem stating that each finite group is isomorphic to a subgroup of the symmetric group. The extended semigroup version of which the original Cayley's theorem is a special case appears, for example, as [38, Theorem 1.1.2].

Theorem 2.12. *Let G be a finite semigroup. There is an injective homomorphism $r : G \to M_{|G|}(\mathbb{Z}_2)$.*

Proof. Fix some arbitrary enumeration of G, say $\{g_1, ..., g_k\}$, and consider the function $G \to M_{|G|}(\mathbb{Z}_2)$ defined by

$$(\psi(g_k))_{i,j} = \begin{cases} 1 & \text{if } g_k g_j = g_i \\ 0 & \text{otherwise} \end{cases}$$

Clearly this is an injective map. To see that it preserves multiplication, notice that

$$\begin{aligned} 1 = (\psi(g_l)\psi(g_{l'}))_{i,j} &\iff \sum_{k=1}^{|G|} \psi(g_l)_{i,k}\psi(g_{l'})_{k,j} = 1 \\ &\iff g_{l'}g_j = g_k \text{ and } g_l g_k = g_i \text{ for some } k \\ &\iff (g_l g_{l'})g_j = g_i \\ &\iff \psi(g_l g_{l'})_{i,j} = 1 \end{aligned}$$

In other words, for each pair $g_l, g_{l'}$, the matrices $\psi(g_l)\psi(g_{l'})$ and $\psi(g_l g_{l'})$ agree on all their entries, so multiplication is preserved. □

The point here is that we can think of $M_{|G|}(\mathbb{Z}_2)$ as an algebra over a field, in the sense that one can add matrices and scale them by field elements[9] in such a way as to satisfy the appropriate distributivity properties. Notice that there the proof of the above result does not depend on

[9]We might also think of $M_{|G|}(\mathbb{Z}_2)$ as the vector space of $|G|^2$-dimensional vectors over $\mathbb{Z}_2$, where the description of these vectors as a matrix serves only to define the multiplication in the algebra.

the choice of $\mathbb{Z}_2$ as the underlying field, and indeed that we could just as well have embedded our group into an algebra over any finite field. Any choice of field using our method, however, would have yielded a $|G|^2$-dimensional algebra. As we shall now see, we are primarily interested in this latter quantity.

2.5.1 Complexity of the Dimension Attack

In the following let us disregard the complexity of computing the function $s_{g,\phi}()$ and evaluating r in order to obtain a clearer picture of the steps common to any choice of semigroup G. For a fixed semigroup G and base pair (g, ϕ), suppose there is an injective homomorphism $r : G \to \mathbb{A}$ for some algebra $\mathbb{A}$ over a field. In order to carry out the dimension attack method outlined in Theorem 2.11, we require a number of field operations bounded in terms of the size of the linear independent set of exchange values - say k - and the dimension of the algebra - say m. Notice that the space generated by the k exchange values is a subspace of the algebra, so immediately we have $k \leq m$.

Ignoring the complexity of computing the appropriate number of values of $r(s_{g,\phi}(i))$, at some stage of the procedure we need to determine whether $k+1$ vectors of length m are linearly dependent. We can do this by Gaussian column elimination on an $m \times (k+1)$ matrix whose columns are $\{r(s_{g,\phi}(1), ..., r(s_{g,\phi}(k+1)))\}$ - by the "big-times-small-squared" mnemonic set out in [12, Appendix B], we expect this to require $\mathcal{O}(m(k+1)^2)$ field operations.

We also need to find the decomposition of $r(s_{g,\phi}(x))$ into a linear combination of the values $\{r(s_{g,\phi}(1), ..., r(s_{g,\phi}(k)))\}$. In particular, we need to find the coefficients v_i such that

$$r(s_{g,\phi}(x)) = \sum_{i=1}^{k} v_i r(s_{g,\phi}(i))$$

Notice that by setting the matrix W as the matrix whose columns are the vectors $r(s_{g,\phi}(i))$, we have that

$$W \begin{bmatrix} v_1 \\ ... \\ v_k \end{bmatrix} = r(s_{g,\phi}(x))$$

It suffices to find a left inverse of W; that is, a $k \times m$ matrix W' such that $W'W = I$. By [12, Section 11.5], since the columns of W are linearly

independent (by definition) and W has more rows than columns, the matrix W^TW is invertible, and indeed one has that $(W^TW)^{-1}W^T$ is a left inverse for W. The most expensive step of this procedure is the calculation of the inverse of the $k \times k$ matrix W^TW, requiring at worst $\mathcal{O}(k^3)$ operations.

Little is known about the value[10] of k. Instead, the general strategy for limiting the effectiveness of the dimension attack will be to find semigroups G for which the only injective homomorphisms are into high-dimensional algebras; since the number of operations required to find the necessary linear independent subset is linear in this quantity, a lower bound on the dimension of an admissible algebra would also lower bound the complexity of this approach.

On the Representation Theory of Finite Groups

The study of maps $\rho : G \to GL(V)$ for some group G and vector space V is known as the *representation theory of groups*. If the vector space is over a field $\mathbb{F}$, a pair (ρ, V) is called a *representation* over $\mathbb{F}$. For finite-dimensional vector spaces V - indeed, Theorem 2.12 shows that every finite group admits a finite-dimensional representation - then after specifying a basis one can always think of $GL(V)$ as a matrix group $GL(k, \mathbb{F})$, where k is the dimension of V and $\mathbb{F}$ is the underlying field. When ρ is injective the representation is called *faithful*, and $r = \rho$ is a map meeting the conditions of Theorem 2.11: $GL(k, \mathbb{F})$ elements can be thought of k^2-dimensional vectors equipped with the standard notion of matrix multiplication. In other words, every faithful, finite-dimensional representation (ρ, V) of a group G is such that ρ is an injective homomorphism from G into an m^2-dimensional algebra, where m is the dimension of V over a field $\mathbb{F}$.

In other words, the study of the efficiency of the dimension attack with respect to a platform G is exactly[11] the study of the dimension of faithful representations of G.

Let us summarise the discussion above by recording the following result:

[10] Later on we will see that we have some crude results determining the size of the complete list of exchange values, but not the dimension.

[11] Technically speaking a function r of the type specified by Theorem 2.11 need not necessarily arise from some embedding into a group of transformations, but in practice these are the known examples.

Theorem 2.13. *Let G be a finite group and suppose (ρ, V) is a faithful representation of G for some m-dimensional vector space V over a field $\mathbb{F}$, and suppose that m is the smallest dimension of a vector space admitting such a representation. For a pair $(g, \phi) \in G \ltimes End(G)$, the complexity of the dimension attack is at best $\mathcal{O}(m^2)$.*

Proof. We know $r = \rho$ satisifies the conditions of Theorem 2.11, where the algebra is of dimension m^2. Since the complexity of the dimension attack has a linear factor in the dimension of the relevant algebra the result follows. □

In other words, assuming that the only appropriate homomorphism into an algebra arises from a faithful representation, we have a lower bound on the complexity of the dimension attack in terms of the dimension of such an attack. Notice also that this is the absolute best case; we have not factored in the computation of the exchange values, nor the extra complexity incurred by the final calculation of the shared key.

We also introduce modules here, which are vector spaces whereby the underlying field can be a ring. A fundamental aspect of representation theory is the correspondence between $K(G)$-modules V and representations (ρ, V) of G over K - see, for example, [40, Theorem 4.4]. Moreover, one can check[12] that this correspondence preserves the dimension of V; that is, each representation (ρ, V) of G over K induces a $K(G)$-module V of the same dimension.

2.6 The Telescoping Attack

The other main approaches to solving SDLP are roughly summarised by the general term *telescoping attacks*. There is less of a feel of a well-defined strategy to attack any semigroup here; before we tie up some branches of the literature let us motivate the reason for changing tack.

Recall that a key tool in the application of the dimension attack is the ability to find a basis for the set of exchange values after mapping into an algebra over a field, where we could use the native addition of the algebra. Another key tool is the distributivity present in the algebra: recall in the proof of Theorem 2.11 that we had a sum of the form

$$\left(\sum_{i=1}^{k} \nu_i r(\phi^y(s_{g,\phi}(i))) \right) r(s_{g,\phi}(y))$$

[12] One does so by simply checking the explicit definition of the correspondence.

by distributivity, the term $r(s_{g,\phi}(y))$ is multiplied with *every* term of the sum, so we can apply Theorem 2.4 to each term and remove the need to know the integer y explicitly. Suppose instead that the group G occurs as the native additive group of an algebra - that is, the abelian group formed by considering an algebra under addition. In this case exchange values have the form $s_{g,\phi}(x) = \sum_{i=1}^{x-1} \phi^i(x)$, the $*$ function is now such that $y * s_{g,\phi}(x) = \phi^y(s_{g,\phi}(x)) + s_{g,\phi}(y)$. Certainly one can still find a basis for the set of exchange values relative to some pair (g, ϕ) - suppose $\{s_{g,\phi}(1), ..., s_{g,\phi}(k)\}$ is such a set, and that we can write

$$s_{g,\phi}(x) = \sum_{i=1}^{k} \nu_i s_{g,\phi}(i)$$

Following the proof of Theorem 2.11, and assuming that ϕ preserves addition, we get

$$\begin{aligned} s_{g,\phi}(x+y) &= \phi^y(s_{g,\phi}(x)) + s_{g,\phi}(y) \\ &= \phi^y\left(\sum_{i=1}^{k} \nu_i s_{g,\phi}(i)\right) + s_{g,\phi}(y) \\ &= \sum_{i=1}^{k} \nu_i \phi^y(s_{g,\phi}(i)) + s_{g,\phi}(y) \end{aligned}$$

At this point in the multiplicative case we would be *multiplying* the first term by the second term, distributing the value $s_{g,\phi}(y)$ across and allowing us to proceed without the need to evaluate ϕ^y, where y is secret. In the additive case we would like to exploit the fact that for each i one has $\phi^y(s_{g,\phi}(i)) + s_{g,\phi}(y) = \phi^i(s_{g,\phi}(y)) + s_{g,\phi}(i)$; so, picking such an i, we have

$$\sum_{i=1}^{k} \nu_i \phi^y(s_{g,\phi}(i)) + s_{g,\phi}(y) = \nu_i \phi^y(s_{g,\phi}(i)) + s_{g,\phi}(y) + \sum_{1 \le j \ne i \le k} \nu_j \phi^y(s_{g,\phi}(j))$$

In other words, we are only able to achieve this substitution removing the need to evaluate ϕ^y for at most one term in the sum, rather than all of them, as in the multiplicative case. Leaving out possible ways around this issue let us try a different approach.

It is noticed (albeit in a case-specific context) by the authors of [13] that

in the additive context,

$$\begin{aligned} g + \phi(s_{g,\phi}(x)) - s_{g,\phi}(x) &= g + \phi\left(\sum_{i=1}^{x-1} \phi^i(g)\right) - \sum_{i=1}^{x-1} \phi^i(g) \\ &= \sum_{i=1}^{x} \phi^i(g) - \sum_{i=1}^{x-1} \phi^i(g) \\ &= \phi^x(g) \end{aligned}$$

Since the terms of the sum cancel each other out, or the sum 'telescopes', the resulting equality is dubbed the *telescoping equality*. In fact it is not too hard to see that a similar equality holds for all semigroups G by applying Theorem 2.4 with $y = 1$; we have

$$\phi(s_{g,\phi}(x))g = \phi^x(g)s_{g,\phi}(x)$$

The point is that presented with an instance of SCDH, one is given the data (g, ϕ) and $s_{g,\phi}(x)$, but $\phi^x(g)$ is not known *a priori*. Moreover, the value $\phi^x(g)$ appears to encode information about the private exponent x differently to the value $s_{g,\phi}(x)$. The general idea of the telescoping-type attacks is to exploit this 'extra' value to solve SCDH.

In contrast with the dimension attack there is not a one-size-fits-all algorithm for using this information to solve SCDH - the three known successful approaches are found in [13], [3] and [50]. In the remainder of this chapter let us study how each of the platforms described in Section 2.4.1 interact with the dimension attack and the telescoping attack.

2.7 Matrices over Group Rings

Recall that the original platform suggested for use with SDPKE is the semigroup of matrices $M_3(\mathbb{Z}_7(A_5))$ with a base homomorphism defined by conjugation by an invertible matrix in $M_3(\mathbb{Z}_7(A_5))$. We have so far only loosely alluded to the formal definition of the object $\mathbb{Z}_7(A_5)$ - this will be our first task.

Definition 2.14. Let G be a finite group and $\mathbb{F}$ be a field. The *group ring* or *group algebra* $\mathbb{F}(G)$ is the set of formal sums[13]

$$\sum_{g \in G} a_g.g$$

[13]A more involved definition justifies the use of scalar multiplication and addition for the symbolic representation given here.

where each a_g is in $\mathbb{F}$. We make this object into a vector space over $\mathbb{F}$ with addition and scalar multiplication

$$\sum_{g\in G} a_g.g + \sum_{g\in G} b_g.g = \sum_{g\in G} (a_g + b_g).g \quad \text{and} \quad \lambda \sum_{g\in G} a_g.g = \sum_{g\in G} (\lambda a_g).g$$

and into an algebra over $\mathbb{F}$ with multiplication

$$\left(\sum_{g\in G} a_g.g\right)\left(\sum_{h\in G} b_h.h\right) = \sum_{g,h\in G} (a_g b_h).gh$$

One checks that the necessary properties of an algebra are satisfied. For our purposes, a notion of addition and multiplication in $\mathbb{Z}_7(A_5)$ gives the canonical definition of matrix addition and multiplication in the set of matrices $M_3(\mathbb{Z}_7(A_5))$, and indeed this object is the choice of platform in [35]. Let us now examine how efficiently the dimension attack can be applied in this context.

2.7.1 The Dimension Attack

We note that the claim of the cryptanalysis specifically addressing this choice of platform applies the technique of Theorem 2.11 directly - but since $M_3(\mathbb{Z}_7(A_5))$ is not an algebra over a field we are not necessarily able to carry out the crucial Gaussian elimination steps. Since $M_3(\mathbb{Z}_7(A_5))$ consists of formal sums of the form $\sum_{g\in A_5, r\in\mathbb{Z}_7} rg$ and $|A_5| = 60$, $|\mathbb{Z}_7(A_5)| = 7^{60}$ and $|M_3(\mathbb{Z}_7(A_5))| = 7^{60\times 9}$. Our Theorem 2.12 therefore gives a prohibitively high-dimensional representation, since the number of operations required in the field is linear in the dimension of the algebra. Using our own technique, developed in [3], let us see how we can do better.

Theorem 2.15. *Let G be a finite group and suppose (ρ, V) is a representation of G for some finite-dimensional vector space V over a field $\mathbb{F}$. There is a homomorphism r from $M_n(\mathbb{F}(G))$ into a $(nk)^2$-dimensional algebra, where k is the dimension of the vector space V.*

Proof. Fixing a basis we may suppose without loss of generality that the homomorphism ρ maps into the group $GL(k, \mathbb{F})$. First let us see that the function $\psi : \mathbb{F}(G) \to M_k(\mathbb{F})$ defined by

$$\psi\left(\sum_{g\in G} a_g g\right) = \sum_{g\in G} a_g \rho(g)$$

is a homomorphism. Following the definitions[14] through we get

$$\begin{aligned}\psi\left(\sum_{g\in G} a_g.g\right)\psi\left(\sum_{h\in G} b_h.h\right) &= \left(\sum_{g\in G} a_g\rho(g)\right)\left(\sum_{h\in G} b_h\rho(h)\right)\\ &= \sum_{g,h\in G} a_g b_h\rho(g)\rho(h)\\ &= \sum_{g,h\in G} a_g b_h\rho(gh)\\ &= \psi\left(\sum_{g,h\in G} a_g b_h.gh\right)\\ &= \psi\left(\sum_{g\in G} a_g.g\sum_{h\in G} b_h.h\right)\end{aligned}$$

In other words, ψ inherits the fact that ρ preserves multiplication.

It remains to map $M_n(\mathbb{F}(G))$ into an algebra over a field. The intuition here is that one should be able to replace each element of $\mathbb{F}(G)$ with its corresponding matrix representation under ψ and inherit the homomorphicity of ψ - as we now show, this can indeed be done. Define $r : M_n(\mathbb{F}(G)) \to M_{nk}(\mathbb{F})$ by

$$(r(M))_{ik+g,jk+h} = (\psi(M_{i,j}))_{g,h}$$

for $0 \leq i,j \leq n-1$ and $0 \leq g,h \leq k-1$. For two matrices $M, M' \in$

[14]Note in the group ring $\mathbb{F}(G)$ we preserve usage of the . notation to denote an abstract notion of scalar multiplication, whereas in $M_k(\mathbb{F})$ the scalar multiplication is defined explicitly as multiplication in the field.

$M_n(\mathbb{F}(G))$, checking an arbitrary entry we have

$$\begin{aligned}
r(MM')_{ik+g,jk+h} &= (\psi((MM')_{i,j}))_{g,h} \\
&= \left(\psi\left(\sum_{l=0}^{n-1} M_{i,l}M'_{l,j}\right)\right)_{g,h} \\
&= \left(\sum_{l=0}^{n-1} \psi(M_{i,l})\psi(M'_{l,j})\right)_{g,h} \text{ since } \psi \text{ preserves addition} \\
&= \sum_{l=0}^{n-1} \left(\psi(M_{i,l})\psi(M'_{l,j})\right)_{g,h} \text{ since addition is component-wise} \\
&= \sum_{l=0}^{n-1} \left(\sum_{l'=0}^{k-1} \psi(M_{i,l})_{g,l'}\psi(M'_{l,j})_{l',h}\right) \\
&= \sum_{l=0}^{n-1}\sum_{l'=0}^{k-1} (r(M))_{ik+g,lk+l'}(r(M))_{lk+l',jk+h} \\
&= \left(r(M)r(M')\right)_{ik+g,jk+h}
\end{aligned}$$

as required. We therefore have a homomorphism into $M_{nk}(\mathbb{F})$, which similarly to previous examples we can think of as a $(nk)^2$-dimensional algebra over $\mathbb{F}$. □

Conditions for Injectivity

Note that we have not shown in the above that r is injective. Following through the argument of Theorem 2.11, the only issue we encounter when injectivity is discarded is that the set

$$r^{-1}(s_{g,\phi}(x+y)) = \{g \in G : r(g) = r(s_{g,\phi}(x+y))\}$$

may not be a single-point set; that is, we could be left with some ambiguity about the true value of $s_{g,\phi}(x+y)$. Let us examine some conditions under which we can expect the map r to be injective.

First, note that in order for r to be injective, it suffices for ψ to be injective, for if $r(M)_{ik+g,jk+h} = r(M')_{ik+g,jk+h}$ for each i, j, g, h then by definition $\psi(M_{i,j}) = \psi(M'_{i,j})$ for each i, j. It follows that $M = M'$ if ψ is injective.

Theorem 2.16. *Let G be a finite group and (ρ, V) a representation of G over some finite field $\mathbb{F}$, where V is a k-dimensional vector space over $\mathbb{F}$ for some*

$k \in \mathbb{N}$. *Relative to an arbitrary choice of basis define* $\psi : \mathbb{F}(G) \to M_k(\mathbb{F})$ *by*

$$\psi\left(\sum_{g\in G} a_g.g\right) = \sum_{g\in G} a_g\rho(g)$$

The following are equivalent:

1. ψ *is injective.*
2. *The set* $\{\rho(g) : g \in G\}$ *is linearly independent in* $M_k(\mathbb{F})$.
3. *The set* V *viewed as an* $\mathbb{F}(G)$*-module, with scalar multiplication defined by*

$$\left(\sum_{g\in G} a_g.g\right) \cdot v = \sum_{g\in G} a_g\rho(g)v$$

is a faithful module, where $\rho(g)v$ *denotes the standard action of a matrix on a vector.*

Proof. It suffices to show that condition $(1) \Rightarrow (2), (2) \Rightarrow (3), (3) \Rightarrow (1)$.

- $(1) \Rightarrow (2)$: It is standard that $\ker\psi = \{0\}$ if ψ is injective: certainly $\psi(0) = 0$, so if $\sum_G a_g.g$ is such that $\psi(\sum_G a_g.g) = 0$, injectivity gives that $\sum_G a_g.g = 0$. In other words, the only set of coefficients $\{a_g : g \in G\}$ such that $\psi(\sum_G a_g.g) = 0$ is the set of zero coefficients, and indeed the equality $\sum_G a_g\rho(g) = 0$ is satisfied only by these zero coefficients. This is precisely linear independence.

- $(2) \Rightarrow (3)$: The module in question is faithful if only the zero $\mathbb{F}(G)$ element annihilates every $v \in V$. Suppose some non-zero $\sum_G a_g.g$ annihilates each $v \in V$, then we have $\sum_G a_g\rho(g)v = 0$ for each $v \in V$ - in other words $\sum_G a_g\rho(g)$ is the zero matrix. But $\{\rho(g) : g \in G\}$ is linearly independent, so this is a contradiction.

- $(3) \Rightarrow (1)$: Suppose for two sets of coefficients $\{a_g : g \in G\} \neq \{b_g : g \in G\}$ one has $\sum_G a_g\rho(g) = \sum_G b_g\rho(g)$. It follows that there is a non-zero set of coefficients $\{c_g := a_g - b_g : g \in G\}$ such that $\psi(\sum_G c_g.g)$ is the zero matrix - in particular, there is a non-zero $\mathbb{F}(G)$ element $\sum_G c_g.g$ that annihilates every $v \in V$, which contradicts the module being faithful. Any set of coefficients $\{a_g : g \in G\}, \{b_g : g \in G\}$ must therefore be such that if $\sum_G a_g\rho(g) = \sum_G b_g\rho(g)$, one has $\{a_g : g \in G\} = \{b_g : g \in G\}$, and therefore ψ is injective.

□

As alluded to above, the efficiency of the dimension attack is a function of the efficiency of a representation of a group. In this case, by reducing the representation to that of a relatively small group the situation is not too bad and we can almost conclude that the platform $M_3(\mathbb{Z}_7(A_5))$ can be considered relatively vulnerable to the dimension attack. The 'almost' here refers to the fact that we have not met one of the conditions of Theorem 2.11; that is, we require proof of the existence of a linear map T on $M_{3k}(\mathbb{Z}_7)$ such that for the choice of ϕ in SCDH, one has $T \circ r = r \circ \phi$. We leave this for future work.

2.7.2 The Telescoping Attack

As we have already discussed, unlike the dimension attack there is not per se a unified strategy with which to analyse the security of SPDH with respect to the telescoping attack. Nevertheless, since $M_3(\mathbb{Z}_7(A_5))$ is a semigroup we immediately encounter a problem - namely, that the equation

$$\phi(s_{g,\phi}(x))g = Ys_{g,\phi}(x)$$

is not solved uniquely by $Y = \phi^x(g)$. An assessment of how much of a problem this is - that is, how many admissible solutions we can expect in this equation - is carried out in a comparable context in Section 2.10.

2.8 p-groups

2.8.1 The Dimension Attack

In [43] the authors seek to address the vulnerability of matrices over group rings we have just addressed. Their ability to do so comes from the following result of Janusz, which appears as [41, Corollary 2.12]:

Theorem 2.17. *Let G be a finite p-group and K be a field that is either infinite or such that $|K| > |\Omega Z(G)|$, where $Z(G)$ is the centre of G, and $\Omega Z(G)$ is the subgroup of $Z(G)$ generated by all the elements of $Z(G)$ of order p. If G has an element of order p^a for some $a \in \mathbb{N}$, a faithful $K(G)$-module V has dimension at least $1 + p^{a-1}$.*

We claim that this suffices to render the dimension attack prohibitively expensive for groups with the appropriate exponent. First, a fundamental

aspect of representation theory is the correspondence between $K(G)$-modules V and representations (ρ, V) of G over K - see, for example, [40, Theorem 4.4]. Moreover, one can check[15] that this correspondence preserves the dimension of V; that is, each representation (ρ, V) of G over K induces a $K(G)$-module V of the same dimension.

Suppose, then, that G is a p-group with exponent p^2 and suppose also that $\rho : G \to GL(V)$ is a faithful representation for a K-vector space V of dimension strictly less than $1 + p$. By the correspondence there would then exist a faithful $K(G)$-module V of dimension strictly less than $1 + p$, which is a contradiction. In other words, if the group G has an element of order p^2 the smallest dimensional K-vector space V such that $\rho : G \to GL(V)$ is a faithful representation is of dimension $1 + p$. By the discussion in Section 2.5.1 the dimension attack requires at least $\mathcal{O}((p+1)^2)$ operations in the field, which is infeasible for large primes. More precisely, if the p-group G has exponent p^2 it must have order at least p^2, and so elements can be represented by bitstrings of length $\mathcal{O}(\log p)$, so the complexity of the dimension attack is exponential in the length of elements $s_{g,\phi}(x)$ for any $x \in \mathbb{N}$.

Corollary 2.18. *Let G be a finite p-group with an element of order p^2. For any base pair $(g, \phi) \in G \ltimes End(G)$, the dimension attack requires $\mathcal{O}((p+1)^2)$ field operations to solve SCDH.*

2.8.2 Telescoping Attack

This time, since we have chosen a full group as the platform, unlike with the previous example the equation

$$\phi(s_{g,\phi}(x))g = Ys_{g,\phi}(x)$$

has a single solution $Y = \phi^x(g)$. The question is then what to do with this value - we will address this after demonstrating an easier case in the next section.

2.9 MAKE

2.9.1 The Dimension Attack

Recall in Section 2.6 the discussion of a group occurring natively as the additive group of an algebra. We saw that such a choice of platform

[15] One does so by simply checking the explicit definition of the correspondence.

would pose a challenge to the approach of the dimension attack. With this in mind[16], for some prime p the platform group $M_3(\mathbb{Z}_p)$ under addition is suggested in [61], and the resulting SDPKE scheme is dubbed 'MAKE'. We have already seen the manner in which the dimension attack fails; let us at last see an example of a successful application of the telescoping attack.

2.9.2 Telescoping Attack

First, note that the authors of [61] suggest the base pair (M, ϕ), where M is any $M_3(\mathbb{Z}_p)$ matrix and $\phi(g) = \phi_{H_1,H_2} = H_1MH_2$ depends on a choice of auxiliary matrices $H_1, H_2 \in M_3(\mathbb{Z}_p)$. As such we have

$$s_{M,\phi}(x) = \sum_{i=0}^{x-1} H_1^i M H_2^i$$

For an instance of SCDH with respect to this choice of platform the relevant data is therefore a base pair (M, ϕ_{H_1,H_2}), and two $M_3(\mathbb{Z}_p)$ elements $A := \sum_{i=0}^{x-1}(H_1^i M H_2^i)$ and $B := \sum_{i=0}^{y-1}(H_1^i M H_2^i)$. Our task is to recover the value $\sum_{i=0}^{x+y-1}(H_1^i M H_2^i)$

Recall the observation of [13]: using this public data we can calculate

$$M + \phi_{H_1,H_2}(A) - A = \phi^x_{H_1,H_2}(M)$$

It remains to describe a method of calculating $\sum_{i=0}^{x+y-1}(H_1^i M H_2^i)$ with access to $\phi^x_{H_1,H_2}(g)$. A solution to this problem is presented in [13], and relies on the following corollary of the Cayley-Hamilton theorem:

Theorem 2.19 (Cayley-Hamilton)**.** *For some $n \in \mathbb{N}$ let A be an $n \times n$ square matrix with entries in a commutative ring R. For any $x \in \mathbb{N}$ there exist R-coefficients $\{r_0, ..., r_{n-1}\}$ such that*

$$A^x = \sum_{i=0}^{n-1} r_i A^i$$

This result gives rise to the following two-part lemma, the proofs of which appear as Lemma 1 and Lemma 2 of [13]:

[16]Note that the authors do not explicitly state this motivation.

Lemma 2.20. *Let $n \in \mathbb{N}$. Define $L : M_n(\mathbb{Z}_p) \to M_{n^2}(\mathbb{Z}_p)$ component-wise by*

$$(L(Y))_{jn+i,hn+g} = (H_1^g Y H_2^h)_{i,j}$$

for $0 \leq i, j, g, h \leq n-1$, and $vec : M_n(\mathbb{Z}_p) \to \mathbb{Z}_p^{n^2}$ by

$$vec(A)_{jn+i} = A_{i,j}$$

for $0 \leq i, j \leq n-1$. Then there is a vector s in $\mathbb{Z}_p^{n^2}$ such that $L(Y)s = vec(H_1^x Y H_2^x)$ for any $Y \in M_n(\mathbb{Z}_p)$. Moreover, for some $Y \in M_n(\mathbb{Z}_p)$, a vector $u \in \mathbb{Z}_p^{n^2}$ satisfying $L(Y)u = 0$ also satisfies $L(H_1^l Y H_2^l)u = 0$ for any $l \in \mathbb{N}$.

Putting the pieces together allows us to conclude as follows.

Theorem 2.21. *The following algorithm recovers $\sum_{i=0}^{x+y-1}(H_1^i M H_2^i)$.*

Algorithm 3 Computing $\sum_{i=0}^{x+y-1}(H_1^i M H_2^i)$.

Input: $(M, \phi_{H_1,H_2}), H_1^x M H_2^x$
Output: $\sum_{i=0}^{x+y-1} H_1^i M H_2^i$

1: $\phi^x_{H_1,H_2}(g) \leftarrow$ recovered from public information
2: $t \leftarrow$ the 9-dimensional vector obtained by solving the system

$$L(g)y = vec(H_1^x g H_2^x)$$

for y
3: $X \leftarrow vec^{-1}(L(B)t)$
4: **return** $K \leftarrow X + A$

(Sketch of proof). By Lemma 2.20, there is at least one solution to the defined system of equations; application of the second part of the lemma and the fact that L preserves addition shows that any solution satisfies $L(B)t = vec(H_1^x B H_2^x)$. Since *vec* is a bijection, applying its inverse to $L(B)t$ allows one to recover $H_1^x B H_2^x$, and therefore $\sum_{i=0}^{x+y-1}(H_1^i M H_2^i)$ by simply adding A to this quantity. □

In [3] it is noticed that a key part of the above attack is the construction of the vector s promised by Lemma 2.20, which is done by the Cayley-Hamilton theorem. It is here that the technique used to proved Theorem 2.15 arises: notice in Theorem 2.15 that the fact that ψ was an injective homomorphism followed from the injectivity and homomorphicity of the underlying map ϕ. Using precisely the same argument, we get the following (which appears as [3, Theorem 1]):

Theorem 2.22. *Let R be a ring. Suppose there is an injective ring homomorphism $\phi : R \to M_m(S)$ for some $m \in \mathbb{N}$ and a commutative ring S. For any $n \in \mathbb{N}$ define*

$$\begin{aligned}\psi :& M_n(R) \to M_{mn}(S) \\ & (\psi(A))_{im+g,jm+h} = (\phi(A_{i,j}))_{g,h}\end{aligned}$$

where $0 \leq i, j \leq n-1$, $0 \leq g, h \leq m-1$. Then ψ is an injective ring homomorphism.

The point here is that an attempt to mitigate the method of [13] by selecting a platform for which the Cayley-Hamilton theorem does not apply, say $M_n(R)$ under addition for a *non*-commutative ring R (as opposed to the commutative ring $\mathbb{Z}_p$) may not be successful. The remainder of [3] goes on to show that extending the attack with respect to the extension works as one might expect - and indeed notes that an example of a non-commutative ring is the group ring $\mathbb{Z}_p(G)$ for some non-abelian group G. By Theorem 2.12 one can construct a function $\phi : \mathbb{Z}_p(G) \to M_{|G|}(\mathbb{Z}_p)$, and by Theorem 2.15 this extends to a homomorphism $\psi : M_n(\mathbb{Z}_p(G)) \to M_{n|G|}(\mathbb{Z}_p)$ - this is effectively the content of [3, Proposition 3].

No Multiplicative Analogue

As well as the Cayley-Hamilton theorem the method of [13] relies on the clever construction of the morphisms L and *vec*. Supposing instead that the group occurs as (or is embedded into) the multiplicative subgroup of an algebra as with other example platforms, one can check the method of [13] no longer works. It is for this reason that we have been referring to the lack of a unified strategy for the telescoping attack - the only successful method thus far demonstrated appears to be specific to an additive context.

2.10 MOBS

Our final case study is that of the machinery underpinning the so-called MOBS cryptosystem, proposed in [60]. The conceit here is basically to use matrices defined over a semiring - that is, a ring in which we do not

have a notion of subtraction - which form a semigroup under the usual notion of matrix multiplication[17].

More specifically we start with the Boolean semiring - that is, the set $\{0,1\}$ equipped with 'addition' $\vee$ defined by $a \vee b = \max\{a,b\}$, and multiplication $\wedge$ defined by $a \wedge b = ab$. Using these notions of arithmetic we define B_k to be the set of k-bitstrings under the above operations applied component-wise. Formally, we have $B_k = \{\{0,1\}^k\}$ where

$$(a_1...a_k) \vee (b_1...b_k) = \max\{a_1,b_1\}...\max\{a_k,b_k\}$$
$$(a_1...b_k) \wedge (b_1...b_k) = (a_1b_1)...(a_kb_k)$$

The choice of platform in [60] is $M_3(B_k)$, where various values of k are suggested - owing to the choice of base pair, any choice of k must be the sum of primes.

Within the modern framework the motivation for this choice is not entirely clear. Certainly the dimension attack would not be feasible in a semialgebra $M_n(S)$ for a semiring S - but the platform is the semigroup formed by $M_n(S)$ under multiplication, and as such embeds into a suitable algebra. It turns out[18], however, that we have an advantageous bound on the dimension of such an algebra.

2.10.1 The Dimension Attack

As usual our task is to find a homomorphism $r : M_3(B_k) \to \mathbb{A}$ for some finite-dimensional algebra $\mathbb{A}$. Since $|M_3(B_k)| = 2^{kn^2}$ the naive representation of Theorem 2.12 gives an embedding into an algebra of dimension 2^{2kn^2}, which is clearly infeasible for even moderate values of k (the authors of [60], for example, suggest $k = 357$).

There has at time of writing been no further effort in the literature to analyse the complexity of the dimension attack with respect to MOBS. Nevertheless, we point out a hitherto unnoticed connection to a promising result - though as far as we can tell, this result was not known to the authors of MOBS, and is not listed as one of their primary motivations for selecting the group. It is trivial to check that $M_3(B_k)$ is the k-fold direct product of the semigroup of single-bit Boolean matrices - that is,

$$M_3(B_k) \cong \underbrace{M_3(B_1) \times ... \times M_3(B_1)}_{k \text{ times}}$$

[17] Notice that addition and multiplication in the underlying ring suffice to define matrix multiplication.

[18] Though this is not mentioned in the original paper.

where the operation is defined component-wise in the obvious way. It turns out by [44, Theorem 5] that the semigroup $M_n(B_1)$ has no faithful representation of dimension smaller than $2^n - 2$. However, extending a result of this type to the general case $M_n(B_k)$ would require the navigation of some representation theoretic technicalities and is an interesting direction of further research.

2.10.2 Telescoping Attack

Since the platform is a semigroup there is no unique solution to the telescoping equality - but this is not unique to this choice of semigroup, and indeed other work exists demonstrating platform-specific vulnerabilities to this choice of platform. Before we summarise this latter result let us quantify precisely how many solutions to the telescoping equality we can expect; to do so we detail the contents of our paper [5]. Before embarking on this exposition it is important to understand the historical context of [5] - at the time, no method of telescoping attack was known other than to attempt to recover $\phi^x(g)$ and deduce ϕ^x by inspection. Moreover, the conclusion of [60] notes that the only known method of computing $\phi^x(g)$ is by 'brute force' - a term roughly meaning to try all possibilities that we will make more precise later on - but the solution may not be unique. In part, then, [5] was an attempt to quantify exactly how many solutions to the telescoping equality existed, and therefore how viable this approach was.

Note that any permutation of a k-bit string can be extended to a a function on $M_3(B_k)$, simply by applying the permutation to each entry of an element of $M_3(B_k)$. In fact, doing so yields an automorphism of $M_3(B_k)$. Let $g \in M_3(B_k)$ and ϕ such an automorphism. We investigate the number of solutions to the equation

$$\phi(s_{g,\phi}(x))g = Ys_{g,\phi}(x)$$

Experiment Design

As we have already discussed the semigroup $M_3(B_k)$ is just the direct product of k copies of $M_3(B_1)$. This means that we can decompose the telescoping equality into k $M_3(B_1)$ equations, and 'reassembling' any combination of the k solutions in $M_3(B_1)$ will give a solution to the telescoping equality. To find solutions to the single-bit equations we simply try all possible $M_3(B_1)$ matrices, of which there are 2^9. In other

words, instead of checking 2^{9k} matrices, we can check $k2^9$. Even at the highest suggested parameter level of $k = 381$, we therefore have to check $381 \times 2^9 < 200000$ matrix values, which is feasible - and indeed this is what is meant by the 'brute force' method alluded to earlier. If there were a unique solution to the telescoping equality we could recover this unique solution by checking sufficiently many $M_3(B_1)$ matrix multiplications, but if the solution is non-unique, even after recovering all the solutions there is no better strategy known to recover the true value than to simply guess. Recovering $\phi^x(g)$ would then be 'infeasible' in the sense that even though a list of possible values could be feasibly arrived at, the probability of selecting the correct one is small.

We carry out three experiments, each time counting the logarithm of the number of admissible values in the telescoping equality:

- Fix a matrix M and vary the exponent x
- Fix the exponent x and vary the public matrix g
- Fix the private exponent x and vary the public matrix g, each time counting the size of the left principal ideal generated by $s_{g,\phi(x)}$ defined by $|\{Bs_{g,\phi}(x) : B \in M_3(B_k)\}|$

Method

The experiments were carried out in Python. First, since we will need a list of all single-bit matrices to iterate through, we generate this outside the loop for less expensive computation. The function `all_matrices` obtains all length n^2 bitstrings from the binary representation of the integers from 0 to $2^{n^2} - 1$, then 'folds' them into single-bit matrices.

```
global mats
mats = all_matrices(3)
```

In accordance with our general strategy we also need a function to count the number of solutions to single-bit matrix equations. We do this in the obvious way with a function called `c_s_s(a,b)`; where a, b are elements of $M_3(B_1)$, and `c_s_s(a,b)` loops over all elements of `mats`, outputting the number of matrices y in `mats` such that $a = yb$. Because any reassembly of single-bit solutions gives a solution to the full equation, the number of solutions to the full equation is the product of the number of solutions to each single-bit equation:

```
def count_solutions(a, b):
    ct = 1
    for i in range(k):
        ct = ct * c_s_s(pull(i, a), pull(i, b))
    return ct
```

In other words, `count_solutions(a,b)` takes as input two $M_3(B_k)$ matrices A, B and returns the number of $Y \in M_3(B_k)$ such that $A = YB$ (where the function `pull` simply returns the $M_3(B_1)$ matrix formed by the i-th component of each bitstring entry of an $M_3(B_k)$ matrix). We are now ready to define the first of our experiments:

```
def count_telescope_solutions_1(g,\phi):
    x = randint(2**n, 2**m)
    b = generate_A(g,\phi,x)
    a = prod_bool_mat(\phi(b), g)
    return log(count_solutions(a, b))
```

Here the function `generate_A` calculates $s_{g,\phi}(x)$ for some random integer x, and `prod_bool_mat` defines the matrix multiplication particular to $M_3(B_k)$. Since the matrix and matrix permutation are to be fixed and the exponent varied, the matrix and permutation are defined outside of the function. The range of values the exponent can be randomly selected from is defined within the function by the parameters n, m.

For the second experiment:

```
def count_telescope_solutions_2(\phi, x, k):
    g = rand_bool_mat(3, k)
    b = generate_A(g,\phi,x)
    a = prod_bool_mat(\phi(b), g)
    return log(count_solutions(a, b))
```

This time we need to input the fixed exponent, and the parameter k from which random $M_3(B_k)$ matrices are generated by the function `rand_bool_mat(3,k)`.

Finally, we need a way to count the size of the left principal ideal generated by A; that is, the size of the set $\{Ys_{g,\phi}(x) : Y \in M_3(B_k)\}$. We can again exploit the fact that the matrix semigroup is a direct sum of single-bit matrix semigroups: we count the size of the ideal generated by each $M_3(B_1)$ matrix, then multiply these numbers. To count the single-bit solutions:

```
def c_s_o(Y):
    orbit = []
    for x in mats:
        if not prod_bool_mat(x, Y) in orbit:
           orbit.append(prod_bool_mat(x, Y))
    return len(orbit)
```

We can then calculate the size of the full ideal:

```
def count_orbit(Y):
    orbit_count = 1
    for i in range(k):
        orbit_count = orbit_count * c_s_o(pull(i, Y))
    return orbit_count
```

Keeping the exponent fixed[19] and varying the public matrix, the final experiment is assembled as follows.

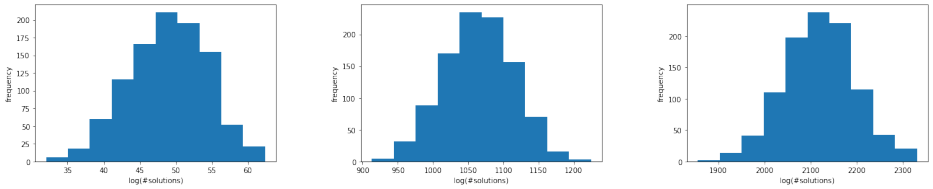
```
def count_telescoping_solutions_orbit(\phi, k, x):
    g = rand_bool_mat(3, k)
    b = generate_A(g, \phi, x)
    a = prod_bool_mat(\phi(b), g)
    return (count_orbit(b), count_solutions(a, b))
```

Results

The results of the trials suggest that:

- Each matrix g corresponds to a fixed number of solutions to the telescoping equality, regardless of exponent
- With the parameters suggested, there are sufficiently many solutions to the telescoping equation for any matrix to make recovery of $\phi^x(g)$ infeasible
- There is negative correlation between the size of the ideal generated by a particular exchange value $s_{g,\phi}(x)$ and the number of solutions to the corresponding telescoping equality

[19]We did not detail the analogous experiment for when the matrix is fixed and the exponent is varied, for reasons that will become clear in the results section.

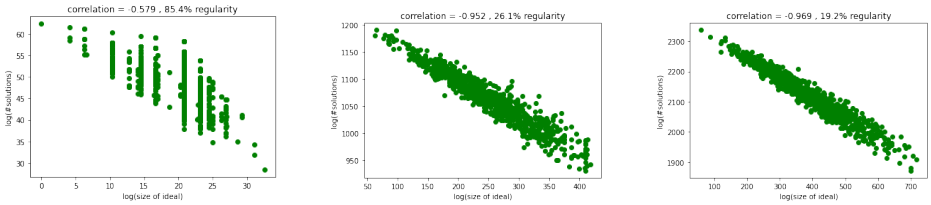


(a) $k = 10$ (b) $k = 197$ (c) $k = 381$

Figure 2.1: Histograms of the logarithm of the number of solutions to the telescoping equality when exponent and permutation are fixed but matrix is chosen at random, for three choices of bitstring length. At the suggested bitstring length the smallest such number of solutions recorded is unphysically large.

Number of solutions is independent of exponent. When the matrix and permutation are fixed but the exponent in the calculation of $s_{g,\phi}(x)$ is varied, over several thousand trials we did not encounter a case where the number of solutions changed. This suggests that the number of solutions to the telescoping equality is independent of the exponent, although we do not have an explanation for this behaviour. For our purposes, assuming that independence from exponent does indeed hold, we conclude that we can run the remaining experiments on small exponents for less expensive computation; that is, we do not need to use the large parameters suggested by the authors of MOBS. In fact, we find that the number of solutions is dependent on exponent for very small values of exponent, but stabilise after a while to independence of exponent; we therefore choose a fixed exponent to balance low computational cost with surpassing this boundary. Arbitrarily, x was fixed at $x = 100$ for the remainder of the trials.

Number of solutions when g is varied. Figure 2.1 shows a histogram of the logarithm of the number of solutions to the telescoping equality when the exponent and permutation are fixed and the public matrix is varied, conducted over a thousand trials. The key takeaway is that in all trials there are far too many solutions to make recovering the correct one a viable strategy; for the suggested parameter $k = 381$ even the smallest number of solutions is in the range of 2^{1900}, and this number of solutions did not occur frequently. The histograms also seem to suggest that the number of solutions are roughly normally distributed within their range.



(a) $k = 10$ (b) $k = 197$ (c) $k = 381$

Figure 2.2: Graphs obtained for different bitstring lengths by fixing the permutation and exponent, choosing the matrix at random, and plotting the logarithm of the size of the left principal ideal generated by that matrix against the logarithm of the number of solutions to telescoping equality corresponding to that matrix. At each bitstring length negative correlation between these two quantities is observed.

Number of solutions decreases with ideal size. Figure 2.2 shows the logarithm of the number of solutions against the logarithm of the size of the principal ideal generated by the corresponding value of $s_{g,\phi}(x)$. The tests were conducted over a thousand trials, each time selecting a $M_3(B_k)$ matrix at random and fixing the matrix permutation and exponent[20]. The graphs exhibit reasonably strong negative correlation, which one would intuitively expect - a larger ideal means that YM lands on the quantity in the telescoping equality less frequently. The vertical lines in graph (a) show that for two matrices whose corresponding exchange value $s_{g,\phi}(x)$ has the same ideal size, their corresponding telescoping equality does not necessarily have the same number of solutions. Indeed, these vertical lines would be present on the other two graphs at higher resolution.

At the top of each graph two data points are noted: first, Spearman's correlation coefficient, second, the percentage of data points achieving regularity. We say a bitstring matrix M is *regular* if the number of Y satisfying $h(A)M = YA$ is the same as the number of Y satisfying $Yh(A)M = A$. We note that as the bitstring length increases we get better correlation and worse regularity rates.

In conclusion, we know of no better way of identifying which of the quantities satisfying the telescoping equality is the correct value to conduct an attack than simply guessing. The large number of solutions

[20]Experiments into ideal size when exponent was varied and other parameters fixed yield similar results to those when number of solutions is counted.

in our results suggest that the probability of choosing the correct value is vanishingly small - in other words, it does not seem feasible to recover the value $\phi^x(g)$ directly. On the other hand, the later work of [50] shows that it suffices to find a function ψ such that $\phi \circ \psi = \psi \circ \phi$ and $\phi(s_{g,\phi}(x))g = \psi(g)s_{g,\phi}(x)$; moreover, a constructive method of computing such a ψ is given, relying on the decomposition into prime order cycles of the permutation ϕ.

It is difficult to make a general statement detailing the impact of this work on the difficulty of SCDH with respect to $M_3(B_k)$. On the one hand, the method of [50] is highly specific to the choice of ϕ in [60] - on the other, the experimental work of [5] is also specific to this choice of ϕ. In other words, choosing a different ϕ to mitigate the impact of [50] could well remove the infeasibility of recovering $\phi^x(g)$ suggested by [5]. The only phenomenon observed that plausibly generalises - or at least, that we have any reason to expect generalises - is the negative correlation between ideal size and number of solutions, for the intuitive reasons mentioned in the relevant section of results. Clearly, much more work is required on this subject.

2.11 Conclusion and Further Work

Semidirect Product Key Exchange, a generalisation of Diffie-Hellman Key Exchange motivated by the post-quantum landscape, gives rise to two problems similar to the Diffie-Hellman problems: SDLP generalising DLP, and SCDH generalising CDH. We have seen that, unlike in the analogous classical case, that there is a gap between SDLP and SCDH; that is, there are groups for which SCDH is easy but no SDLP algorithm is known. These examples arise from groups in which one of the two main attack strategies is applicable. Nevertheless, with suitable care there are examples of groups for which neither of these attacks appear feasible; and most of the open questions on limiting the efficiency of these attacks give rise to a number of interesting problems in representation theory and semigroup theory. The reduction of the dimension attack to the study of efficiency of faithful representations is a particularly rich connection which this work has only really scratched the surface of.

Despite all this, the difficulty of SDLP remains unaddressed. Certainly this is crucially relevant to the study of the difficulty of SCDH - any group in which one can efficiently solve SDLP is trivially a group in which one can solve SCDH - but none of the literature covered in the

preceding chapter has much to say about its difficulty, whether classical or otherwise. In the next chapter we will take important steps to resolving this state of affairs.

3 Solving the Semidirect Discrete Logarithm Problem

> Is this a dagger which I see before me, The handle toward my hand?
>
> *Macbeth*, Act II

At this point in the thesis we will start to see a marked shift in tone, away from the cryptanalysis of the previous chapter that was classical and more broadly linear algebraic in nature, and towards quantum, more group-theoretic methods. In this chapter a change in perspective allows us to prove, via some standard ideas in semigroup theory, that SDLP is remarkably well described by the notion of a group action. In particular, this connection will allow us to obtain an upper bound on the quantum hardness of SDLP by repurposing a seemingly unrelated area of the literature - that of the theory of cryptographic group actions.

In other words, in order to understand the quantum complexity of solving SDLP we must first understand some ideas in semigroup theory and in the theory of cryptographic group actions. This is our first task.

3.1 Background

A natural candidate for post-quantum group-based cryptography is DLP in semigroups - after all, the reduction of DLP to Shor's algorithm relies on the invertibility of the relevant group elements. It can be shown, however, that the cyclic structure generated by powers of a non-invertible semigroup - called the *monogenic semigroup* - must contain a cyclic group (this is a rather standard fact which can be found, for example, in [38]). With a little extra work one can use this fact to show that the difficulty of the DLP in a semigroup is asymptotically no worse than that of the usual DLP, although this result comes in two varieties. One strategy is to reduce the DLP in a semigroup to the DLP in some full group, which

has the advantage of potentially allowing a fully classical algorithm if the DLP is classically easy in the resulting group. The other, and the one we use, is the work of [19]: here, an algorithm similar to Shor's elementary period-finding method (originally given in [20]) is used to detect the cyclic group contained in the monogenic semigroup. We prefer this method, as an adaptation of the period-finding algorithm will allow us to detect an analogous quantity.

Of course, one should not resort to quantum methods automatically, and indeed our choice of a quantum method to clear the semigroup-related obstacle is justified by the best-known algorithm for the remaining problem being quantum. The problem in question here originates in [23] and is known as the *vectorisation problem*, essentially generalising DLP to the setting of certain 'well-behaved' group actions (in the sense that classic DLP occurs as a special case). The examples of suitable group actions here arise from what would become known as *isogeny*-based cryptography, and are arrived at independently by Stolbunov and Rostovstev in [70, 71, 66]. For our purposes, the key result we are interested in is that of [18] - the appropriate isogeny problem is solved effectively by solving the vectorisation problem, which is itself reducible to the abelian hidden shift problem solved by Kuperberg in his famous paper on the quantum difficulty of the dihedral hidden subgroup problem [45]. It is shown that Kuperberg's subexponential complexity translates to the setting of the vectorisation problem, and therefore that we have subexponential complexity for solving the vectorisation problem.

Whether subexponential complexity is appropriate for cryptography remains rather up for debate. Concurrent with the discovery of a subexponential algorithm in [18] the isogeny-based cryptography literature becomes, for a while, more focused on SIKE - first appearing in [42] - which does not have the vulnerability of reduction to the abelian hidden shift problem. On the other hand, in [15], a development in isogeny theory tackles the other, more serious flaw in the constructions of Couveignes, Stolbunov and Rostostev - an unacceptably slow execution time. These new fast isogenies allow for the definition of an efficient non-interactive key exchange called CSIDH. Following the demise of SIKE through cryptanalysis in [14], CSIDH is at time of writing arguably the best-known example of isogeny-based cryptography, and indeed the paper [15] contains a section arguing that the subexpontential complexity of quantum attacks, while asymptotically suboptimal, may be hindered by large constants at the suggested parameter levels. We regard this investigation as ongoing, and of sufficient complexity to warrant its own

body of work.

It is worth pointing out that some connection between group actions and group based cryptography had already been established in Monico's PhD thesis [51], in which *semigroup actions* are proposed - that is, the action of some finite semigroup on a set. Later, a more detailed example of a semigroup action arising for semirings would be proposed in [36]. Aside from a concurrent mention in the introduction of the recent work [36], we are not aware of the connection between the structure of SPDKE and group actions in the literature.

Finally, before going on to argue technically about the appropriate group actions, we insist in this chapter that a semigroup G means a strict semigroup - that is, we assume that any semigroup element we pick does not have an inverse contained in the semigroup. As we shall see in the next chapter, allowing invertibility changes the structure rather significantly, and a catch-all argument is difficult. In the original work [4] that the algorithm draws from, the decision to analyse semigroups rather than full groups was taken since semigroup examples are marginally more prevalent in the SDPKE literature.

3.2 A Novel Group Action

All of the algorithms in this chapter rely on the construction of a certain group action - recall that such an object consists of a group, a set, and a function. As a general outline to our strategy, we first define and deduce properties of a particular set, from which the appropriate group and function will follow.

For the remainder of this section by G we mean an arbitrary finite semigroup, and by $End(G)$ we mean its associated endomorphism semigroup.

Definition 3.1. For a pair $(g, \phi) \in G \times End(G)$, define

$$\mathcal{X}_{g,\phi} := \{s_{g,\phi}(i) : i \in \mathbb{N}\}$$

We will often write $\mathcal{X}_{(g,\phi)}$ as $\mathcal{X}$ when clear from context. Certainly this object is neither a group nor a semigroup - numerous counterexamples can be found whereby multiplication of elements in this set are not contained in the set - but we can make some progress by borrowing from the standard theory of monogenic semigroups; presented, for example, in [38]. Since $\mathcal{X} \subset G$, $\mathcal{X}$ is finite — the set $\{x \in \mathbb{N} : \exists y \neq x \quad s_{g,\phi}(x) =$

$s_{g,\phi}(y)\}$ must therefore be non-empty, or the set would be in bijection with the natural numbers, contradicting the fact that G is finite. We may therefore choose the smallest element of this set, say n. By definition of n the set $\{x \in \mathbb{N} : s_{g,\phi}(n) = s_{g,\phi}(n+x)\}$ must also be non-empty, so we may again pick its smallest element and call it r.

The structure of $\mathcal{X}$ is further restricted by the ability to add in the argument of $s_{g,\phi}()$. Recall that Theorem 2.4 tells us that for integers x and y, one has

$$\phi^x\left(s_{g,\phi}(y)\right) s_{g,\phi}(x) = s_{g,\phi}(x+y)$$

This method of inducing addition in the integer argument of s is sufficiently important that we will invoke a definition for it.

Definition 3.2. Let $(g,\phi) \in G \times End(G)$ and define a function $f : \mathbb{N} \times \mathcal{X} \to \mathcal{X}$ by

$$f(i, s_{g,\phi}(j)) = \phi^i(s_{g,\phi}(j)) \cdot s_{g,\phi}(i)$$

where $f(i, s_{g,\phi}(j))$ may also be written as $i * s_{g,\phi}(j)$. By Theorem 2.4, $i * s_{g,\phi}(j) = s_{g,\phi}(i+j)$

Thus far we have established that corresponding to any fixed pair $(g,\phi) \in G \times End(G)$ is a set $\mathcal{X}_{g,\phi} = \mathcal{X}$ and a pair of integers n, r. Armed with our new definition of $*$ we know that $i * s_{g,\phi}(j) = s_{g,\phi}(i+j)$ for any $i, j \in \mathbb{N}$, so by definition of n, r we have

$$\begin{aligned} s_{g,\phi}(n+2r) &= r * s_{g,\phi}(n+r) \\ &= r * s_{g,\phi}(n) \\ &= s_{g,\phi}(n+r) \end{aligned}$$

We conclude, by extending this argument in the obvious way, that $s_{g,\phi}(n+qr) = s_{g,\phi}(n)$ for each $q \in \mathbb{N}$. In fact, we have the following:

Lemma 3.3. *Fix $(g,\phi) \in G \times End(G)$ and let n, r be the corresponding integer pair as above. One has that*

$$s_{g,\phi}(n+x+qr) = s_{g,\phi}(n+x)$$

for all $x, q \in \mathbb{N}$.

We will frequently invoke Lemma 3.3. Indeed, we immediately get that the set $\mathcal{X}$ cannot contain values other than $\{g, ..., s_{g,\phi}(n), ..., s_{g,\phi}(n+r-1)\}$. If any of the values in $\{g, ..., s_{g,\phi}(n-1)\}$ are equal we contradict the minimality of n, and if any of the values in $\{s_{g,\phi}(n), ..., s_{g,\phi}(n+r-1)\}$ are equal we contradict the minimality of r. We have shown the following:

Theorem 3.4. *Fix $(g, \phi) \in G \times End(G)$. The set $\mathcal{X} = \{s_{g,\phi}(i) : i \in \mathbb{N}\}$ has size $n + r - 1$ for integers n, r dependent on g, ϕ. In particular*

$$\mathcal{X} = \{g, ..., s_{g,\phi}(n), ..., s_{g,\phi}(n + r - 1)\}.$$

We refer to the set $\{g, ..., s_{g,\phi}(n-1)\}$ as the *tail*, written $\mathcal{T}_{g,\phi}$, of $\mathcal{X}_{g,\phi}$; and the set $\{s_{g,\phi}(n), ..., s_{g,\phi}(n + r - 1)\}$ as the *cycle*, written $\mathcal{C}_{g,\phi}$, of $\mathcal{X}_{g,\phi}$. The values $n_{g,\phi}$ and $r_{g,\phi}$ are called the *index* and *period* of the pair (g, ϕ). We shall feel free to omit the subscript at will when clear from context.

One can see that unique natural numbers correspond to each element in the tail, but infinitely many correspond to each element in the cycle. In fact, each element of the cycle corresponds to a unique residue class modulo r, shifted by the index n. This is a rather intuitive fact, but owing to its usefulness we will record it formally. In the following we assume the function $\mod$ returns the canonical positive residue.

Theorem 3.5. *Fix $(g, \phi) \in G \times End(G)$ and let $x, y \in \mathbb{N}$. We have*

$$s_{g,\phi}(n + x) = s_{g,\phi}(n + y)$$

if and only if $x \mod r = y \mod r$.

Proof. In the reverse direction, setting $x' = x \mod r$ and $y' = y \mod r$, we have by Lemma 3.3 that $s_{g,\phi}(n + x) = s_{g,\phi}(n + x')$ and $s_{g,\phi}(n + y) = s_{g,\phi}(n + y')$. By assumption $x' = y'$, and $0 \leq x', y' < r$. The claim follows since we know values in the range $\{s_{g,\phi}(n), ..., s_{g,\phi}(n + r - 1)\}$ are distinct by Theorem 3.4.

On the other hand, suppose $s_{g,\phi}(n + y) = s_{g,\phi}(n + x)$ but $x \not\equiv y \mod r$. Without loss of generality we can write $y = x' + u + qr$ for some $q \in \mathbb{N}, 0 < u < r$ and $x' = x \mod r$. By Lemma 3.3, since $s_{g,\phi}(n + y) = s_{g,\phi}(n + x)$ we must have

$$s_{g,\phi}(n + x') = s_{g,\phi}(n + x' + u)$$

where $s_{g,\phi}(n + x) = s_{g,\phi}(n + x')$ also by Lemma 3.3. There are now three cases to consider; we claim each of them gives a contradiction.

First, suppose $x' + u = r$, then $s_{g,\phi}(n + x') = s_{g,\phi}(n)$. Since $x' < r$ we contradict minimality of r. The case $x' + u < r$ gives a similar contradiction.

Finally, if $x' + u > r$, without loss of generality we can write $x' + u = r + v$ for some positive integer v, so we have $s_{g,\phi}(n + x') = s_{g,\phi}(n + v)$. Since $x' \neq v$ (else we contradict $u < r$), and both values are strictly less

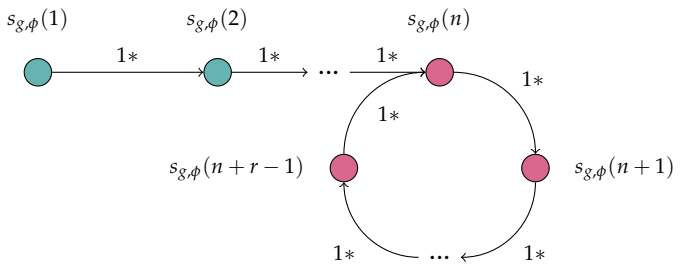


Figure 3.1: Structure of the exponents when at least one of g or ϕ is not invertible, showing the two distinct 'regions' of the set.

than r, we have a contradiction, since distinct integers of this form give distinct evaluations of s.

□

Figure 3.1 summarises this state of affairs, where one can think of moving between each adjacent node on the graph by applying the operation $1*$ to the current node.

3.2.1 A Group Action

It should be clear by now that we are interested in the argument of s in terms of residue classes modulo r. Recall that the group of residue classes modulo r is denoted $\mathbb{Z}_r$, and its elements are written as $[i]_r$. We conclude the section by constructing the action of $\mathbb{Z}_r$ on the cycle $\{s_{g,\phi}(n), ..., s_{g,\phi}(n+r-1)\}$, where we assume that the operator $\mod r$ returns the unique integer in $\{0, ..., r-1\}$ associated to its argument.

Theorem 3.6. *Fix $(g, \phi) \in G \times End(G)$ and let n, r be the index and period corresponding to g, ϕ. Moreover, let $\mathcal{C}$ be the corresponding cycle of size r. The abelian group $\mathbb{Z}_r$ acts freely and transitively on $\mathcal{C}$.*

Proof. First note that Theorem 3.5 immediately gives that $j * s_{g,\phi}(i+n) = s_{g,\phi}((i+j) \mod r+n)$ for any $j \in \mathbb{N}$. Our current definition of s is not defined for negative integer arguments; nevertheless, we can extend the range of the operator $*$ as follows. Let $* : \mathbb{Z} \times \mathcal{C} \to \mathcal{C}$ be defined by

$$j * s_{g,\phi}(i) = \phi^{j \mod r}(s_{g,\phi}((i+n)) \cdot s_{g,\phi}(j \mod r)$$

Since $j \mod r \geq 0$, as usual we have $j * s_{g,\phi}(i+n) = s_{g,\phi}(i+j \mod r+n)$; but since $s_{g,\phi}(i+n) \in \mathcal{C}$, we know $0 \leq i < r$, so $i \mod r = i$. It follows that $j * s_{g,\phi}(i+n) = s_{g,\phi}((i+j) \mod r+n)$.

In fact, fix some $i \in \mathbb{N}$, and let $[j]_r$ be a fixed element of $\mathbb{Z}_r$. By definition, every $k \in [j]_r$ is such that $k \mod r = j'$ for some $j' \in \{0, ..., r-1\}$; without loss of generality, $j' = j$. We may therefore define $\circledast : \mathbb{Z} \times \mathcal{C} \to \mathcal{C}$ by

$$[j]_r \circledast s_{g,\phi}(i+n) = s_{g,\phi}((i+j) \mod r+n))$$

where j is the unique element of $[j]_r$ such that $k \mod r = j$ for each $k \in [j]_r$. We claim that $(\mathbb{Z}_r, \mathcal{C}, \circledast)$ is a free, transitive group action.

First, let us verify that a group action is indeed defined. Certainly $[0]_r$ fixes every element in $\mathcal{C}$, since $s_{g,\phi}((i+0) \mod r+n) = s_{g,\phi}(i+n)$ for each $i \in \{0, ..., r-1\}$. Moreover, one has

$$\begin{aligned}
[k]_r \circledast ([j]_r \circledast s_{g,\phi}(i+n)) &= [k]_r \circledast s_{g,\phi}((i+j) \mod r+n) \\
&= s_{g,\phi}(((i+j) \mod r) + k \mod r+n) \\
&= s_{g,\phi}((i+(j+k)) \mod r+n) \\
&= [j+k]_r \circledast s_{g,\phi}(i+n) \\
&= ([k]_r + [j]_r) \circledast s_{g,\phi}(i+n)
\end{aligned}$$

It remains to check that the action is free and transitive. If $[j]_r \in \mathbb{Z}_r$ is such that $[j]_r$ fixes an arbitrary element of $\mathcal{C}$, say $s_{g,\phi}(i+n)$, then we have $s_{g,\phi}((i+j) \mod r+n) = s_{g,\phi}(i+n)$. By Theorem 3.5, we must have $i+j \equiv i \mod r$, so $[j]_r = [0]_r$ and the action is free. Moreover, for arbitrary $s_{g,\phi}(i+n), s_{g,\phi}(j+n) \in \mathcal{C}$, $[k]_r = [j-i]_r \in \mathbb{Z}_r$ is such that $[k]_r \circledast s_{g,\phi}(i+n) = s_{g,\phi}(j+n)$, so the action is also transitive and we are done.

□

We summarise the above by noting that for each $(g, \phi) \in G \times End(G)$ we have shown the existence of a free, transitive, commutative group action $(\mathbb{Z}_r, \mathcal{C}, \circledast)$, where r and $\mathcal{C}$ depend on the choice of pair (g, ϕ).

3.3 Group Action Discrete Logarithms

Now that we have established that the structure of SDLP defines a group action we aim to use techniques from the theory of cryptographic group actions to study the difficulty of SDLP. In this section we show that, along

with some quantum techniques, that we therefore have a reduction to the vectorisation problem mentioned in the background section. We will refer to this problem henceforth as the *group action discrete logarithm problem*, or GADLP:

Definition 3.7 (Group Action Discrete Logarithm)**.** Given a public commutative group action $(G, X, \star)$, sample $g \in G$ and $x \in X$ uniformly at random, compute $y = g \star x$ and create the pair (x, y). The Group Action Discrete Logarithm Problem (GADLP) with respect to x is to recover g given the pair (x, y).

3.3.1 Modelling Parameterisation

The purpose of this chapter is to estimate the complexity of solving SDLP - more accurately, to estimate how the complexity of solving SDLP grows. Here we have a decision to make: in terms of what parameter should we give our complexity estimates? The obvious choice is to give the estimate in terms of the size of the semigroup under consideration; however, we have seen in Chapter 1 that every extant proposal of a platform for SDPKE suggests for use some variety of matrix algebra. In particular, insofar as parameters are recommended, the convention is to fix a matrix size - usually 3 - and adjust the size of an underlying ring in order to increase security.

In other words having defined SDLP relative to some semigroup G and its endomorphism semigroup $End(G)$, we can think of each such semigroup as one of a family of semigroups $\{G_p\}_p$, where the family $\{G_p\}_p$ is indexed by some set parameterising the underlying algebra (usually the primes). Note that this immediately induces a family of endomorphism semigroups $\{End(G_p)\}_p$, so we can talk about pairs (g, ϕ) from the set $G_p \times End(G_p)$ for each p.

Table 3.1 gives examples of platforms over 3×3 matrices, the size of the platform, and the variable that can be considered as the indexing variable[1].

In each of these examples we have a family of semigroups indexed by some set P such that each semigroup G_p has size polynomial in p. We will give complexity estimates as a function of p.

[1]Note here that $|R|$ is chosen as the parameter for reasons of efficiency of representation.

Table 3.1: Growth of proposed platforms as a function of the variable parameterising the size of an underlying algebraic structure.

Proposed Platform	Size of Platform	Indexing Variable
$M_3(G(R))$	$\lvert R\rvert^{9\lvert G\rvert}$	$\lvert R\rvert$
Certain classes of p-group	Polynomial in prime p	Prime p
$M_3(\mathbb{Z}_p)$	p^9	p

Computing $s_{g,\phi}()$

For a semigroup G, note that in $G \ltimes End(G)$ we have $(g,\phi)^x = (s_{g,\phi}(x), \phi^x)$ by definition. By standard square-and-multiply techniques it therefore requires $\mathcal{O}(\log x)$ applications of the operation in $G \ltimes End(G)$ to compute $s_{g,\phi}(x)$.

In order to estimate the complexity of the operation in $G \ltimes End(G)$ we need to know the complexity of multiplication in G and that of applying the endomorphism ϕ. In this direction we note that another characteristic of the currently proposed platforms is that the recommended endomorphisms typically involve multiplication by one or more auxiliary matrices; that is, for a particular semigroup G_p, if $(g,\phi) \in G_p \times End(G_p)$ the group element $\phi(g)$ has the form $A \cdot g \cdot B$, where $A, B \in G$ are fixed. If the matrix size is fixed each application of ϕ therefore requires some constant number of operations in the underlying ring of the matrix semigroup, which we may assume has size polynomial in p. The complexity of this matrix multiplication will be dominated by the multiplication in the underlying ring. Since the size of the underlying ring is also polynomial in p, each multiplication has complexity $\mathcal{O}((\log p)^2)$ (since $\mathcal{O}(\log poly(p)) = \mathcal{O}(\log p)$). We conclude that both multiplication of elements in G_p, and evaluation of $\phi(g)$, can be done in time $\mathcal{O}((\log p)^2)$.

With these observations in mind, we define the following:

Definition 3.8. Let P some countable indexing set. A family of semigroups $\{G_p\}_{p\in P}$ is said to be *easy* if

1. $\lvert G_p\rvert$ grows monotonically and polynomially in p
2. For any p, any tuple $(g,h,\phi) \in G_p \times G_p \times End(G_p)$ is such that $g \cdot h$ and $\phi(g)$ can be evaluated in time $\mathcal{O}((\log p)^2)$.

Many of the complexity results within the chapter assume that we are dealing with an easy family of semigroups, basically in an attempt to

model the behaviour of suggested examples of semigroup family. Note that the discussion above shows the following:

Lemma 3.9. *Let $G_p \in \{G_p\}_{p\in P}$ be one of an easy family of semigroups. For any $p \in P$, one can compute $s_{g,\phi}(x)$ in time $\mathcal{O}(\log x(\log p)^2)$ for any pair $(g,\phi) \in G_p \times End(G_p)$.*

3.4 The Main Reduction

Let $\{G_p\}_p$ be an easy family of semigroups. In the Section 3.2 we have shown that for a fixed p, to each pair $(g,\phi) \in G_p \times End(G_p)$ is associated a pair (n,r) and a set $\mathcal{C}$. In this section we seek to show there is an efficient quantum algorithm to solve SDLP with respect to an arbitrary choice of (g,ϕ), provided one has access to a GADLP oracle for the group action $(\mathbb{Z}_r, \mathcal{C}, \circledast)$.

Before giving this reduction there remains a significant obstacle to overcome: for an arbitrary pair (g,ϕ) we have only proved the existence of the corresponding values n,r, but we do not have a means of calculating them. In order to provide a reduction to a GADLP oracle, however, we need to specify the appropriate group action. We therefore require access to the values n,r - in the next section, we will provide a quantum method of recovering these integers. As pointed out in Section 3.1, assuming access to a quantum computer is, for our purposes, justified since the best-known algorithms for GADLP are quantum anyway.

3.4.1 Calculating the Index and Period

In order to reason on the complexity of our algorithm we will use the following worst-case indicator, defined as follows:

Definition 3.10. Let $\{G_p\}_{p\in P}$ be an easy family of finite semigroups parameterised by some set P. Define the following function on P:

$$N(p) = \max_{(g,\phi)\in G_p\times End(G_p)} |\mathcal{T}_{g,\phi} + \mathcal{C}_{g,\phi}|$$

The function $N(p)$ gives a bound on the size of $\mathcal{X}_{g,\phi}$ for any $(g,\phi) \in G_p \times End(G_p)$. Since a crude such bound is the size of an easy semigroup G_p, which is assumed polynomial in p, we have that $N(p)$ is at worst polynomial in p.

Our method of calculating the index and period borrows heavily from ideas in [19, Theorem 1], which is itself a slightly repurposed version of [20, Algorithm 5]. Indeed, after a certain point we will be able to quote methods of these algorithms verbatim - nevertheless, to cater to our specific context it remains incumbent upon us to justify the following.

Lemma 3.11. *Let $\{G_p\}_p$ be an easy family of semigroups, and for an arbitrary p fix a pair $(g, \phi) \in G_p \times End(G_p)$. For any $l \in \mathbb{N}$, one can construct the superposition*

$$\frac{1}{\sqrt{M}} \sum_{k=0}^{M-1} |k\rangle \, |s_{g,\phi}(k)\rangle$$

in time $\mathcal{O}((\log M)(\log p)^2)$, where $M = 2^l$.

Proof. Recall that we think of $sdpi : G \times End(G) \times \mathbb{N} \to G$ as a function $s_{g,\phi}(i) : \mathbb{N} \to G$. Since $N(p)$ is a bound on the size of $\mathcal{X}_{g,\phi}$, taking m to be smallest integer such that $2^m \geq N(p)$ (note that $m = \mathcal{O}(\log(N(p)))$), the set $\mathcal{X}_{g,\phi}$ has binary representation in the set $\{0,1\}^m$. By definition the integers $\{0, ..., M-1\}$ have binary representation in $\{0,1\}^l$, so we can think of the restriction of $s_{g,\phi}$ on $\{0, ..., M-1\}$ as a function from $\{0,1\}^l$ into $\{0,1\}^m$. Following our discussion in Section 1.2.4, we therefore assume there is a quantum circuit, say $Q_{s_{g,\phi}}$ implementing $s_{g,\phi}()$ that runs in time no worse than that of computing $s_{g,\phi}(M)$. By Lemma 3.9 this time is $\mathcal{O}((\log M)(\log p)^2)$.

If we can show a single application of $Q_{s_{g,\phi}}$ gives the desired superposition we are done. It is standard, however, that the uniform superposition of an M-bit quantum register, together with an ancillary m-bit register in the state $|0\rangle$, can be inputted into $Q_{s_{g,\phi}}$ to produce the desired superposition. This effect is described in [63, Section 7.1.2] as *quantum paralellism*. Since preparing the appropriate uniform superposition can be done by applying a Hadamard gate in time $\mathcal{O}(\log M)$, we are done.

□

Armed with the ability to efficiently calculate the appropriate superposition, we will quickly find ourselves with exactly the kind of state arrived at in [20, Algorithm 5], thereby allowing us to recover the period r in Algorithm 4. A small adaptation of standard binary search techniques completes the task by using knowledge of r to recover the index n.

Theorem 3.12. *Let $\{G_p\}_p$ be an easy family of semigroups, and fix p. For any pair $(g, \phi) \in G_p \times End(G_p)$:*

Algorithm 4 PeriodRecovery($((g, \phi), M)$)

Input: Pair $(g, \phi) \in G_p \times End(G_p)$, upper bound on size of superposition to create M
Output: Period r of (g, ϕ) or 'Fail'

$R_0 \leftarrow |0\rangle |0\rangle$
$R_1 \leftarrow$ Hadamard transform applied to first register
$R_2 \leftarrow$ appropriate quantum circuit applied to R_1
Measure second register leaving collapsed first register R_3
$R_4 \leftarrow$ QFT over $\mathbb{Z}_{\mathbb{M}}$ applied to R_3
$R_5 \leftarrow$ measure R_4
$r \leftarrow$ continued fraction expansion of R_5 / M
if $r * s_{g,\phi}(M) \neq s_{g,\phi}(M)$ **then**
    return 'Fail'
else
    return r
end if

Algorithm 5 BinarySearch($(g, \phi), start, end, r$)

Input: Pair (g, ϕ), integers $start, end$ where $start \leq end$, period r of g, ϕ
Output: Index n of (g, ϕ)

if $start = end$ **then**:
    return $start$
end if
$left \leftarrow start$
$right \leftarrow end$
$mid \leftarrow \lfloor (left + right)/2 \rfloor$
if $r * s_{g,\phi}(mid) \neq s_{g,\phi}(mid)$ **then**
    return BinarySearch($(g, \phi), mid + 1, right, r$)
else
    return BinarySearch($(g, \phi), left, mid, r$)
end if

1. *For sufficiently large $M \in \mathbb{N}$, PeriodRecovery($(g, \phi), M$) recovers the period r of (g, ϕ) in time $\mathcal{O}((\log p)^3)$, and with constant probability.*

2. *BinarySearch($(g, \phi), 1, M, r$) returns the index n of g, ϕ in time $\mathcal{O}((\log p)^4)$.*

Proof. 1. Fix a pair $(g, \phi) \in G_p \times End(G_p)$ and let r be its period. Let

$\ell \in \mathbb{N}$ be the smallest positive integer such that $2^\ell \geq (N(p)^2 + N(p))$, and $M = 2^\ell$. In steps 1-3 of Algorithm 4, we prepare the required superposition as described in Lemma 3.11.

In Step 4, we measure the second register[2]. With probability n/M doing so will cause us to observe an element of the tail; that is, some $s_{g,\phi}(i)$ such that $i < n$. In this case, by the laws of partial observation, the first register is left in a superposition of integers corresponding to this value - but by definition there is only one of these, so the first register consists of a single computational basis state and the algorithm has failed. On the other hand, with probability $(M-n)/M$ measuring the second register gives an element of $\mathcal{C}$. Now, since $M \geq N(p)^2 + N(p)$, we observe an element of $\mathcal{C}$ with probability

$$\frac{M-n}{M} = 1 - \frac{n}{M} \geq 1 - \frac{n}{N(p)^2 + N(p)}$$

Since by definition one has $n \leq N(p)$, it follows that the relevant probability is better than $N(p)/(N(p)+1) \geq 1/2$. In other words, we observe an element of the desired form with constant, positive probability. Provided such an element was observed, after measuring the second register, the superposition of corresponding integers in the first register is the following:

$$\frac{1}{\sqrt{s_r}} \sum_{j=0}^{s_r - 1} |x_0 + jr\rangle$$

To see this, note that the function s is periodic of period r, and by Theorem 3.4 each $s_{g,\phi}(i)$ such that $i \geq n$ can only assume one of the distinct values $s_{g,\phi}(n), ..., s_{g,\phi}(n+r-1)$. In particular, the integers in $\{1, ..., M\}$ that give a specific value of the cycle under s are of the form $x_0 + jr$ for some $x_0 \in \{n, ..., n+r-1\}$. The largest such integer, by definition, is $x_0 + s_r r$, where s_r is just the largest integer such that $x_0 + s_r r < M$. Note that the superposition is normalised by this factor so that the sum of the squares of the amplitudes is 1.

We now have exactly the same kind of state found in [20, Algorithm 5][3], so we may proceed exactly according to the remaining

[2]The principle of implicit measurement [54, p.187] actually gives that we do not need to perform this measurement at all - we can simply discard the second register. Nevertheless, understanding what happens when we do this is best understood by imagining that we have indeed measured the appropriate register.

[3]This type of state also occurs in Shor's factoring algorithm.

steps in this algorithm. In Step 5 we apply a Quantum Fourier Transform (QFT) over $\mathbb{Z}_M$ to the state, which can be done in time $\mathcal{O}((\log M)^2)$. In step 6 we measure the state R_4; it is shown in [20, Algorithm 5, Step 5] that with probability at least $4/\pi^2$, measuring the resulting state leaves one with the closest integer to one of the at most r multiples of M/r (note that M/r is not necessarily an integer) with probability better than $4/\pi^2$. Writing this closest integer as $\lfloor jM/r \rceil$ for some $j \in \mathbb{N}$, one checks that the fraction j/r is a distance of at most $1/2M$ from $(\lfloor jM/r \rceil)/M$; by [37, Theorem 8.4.3], j/r will appear as one of the convergents in the continued fraction expansion of $(\lfloor jM/r \rceil)/M$ provided $1/2M \leq 1/2r^2$. Certainly this holds, since $r < N(p) < M$. Provided we have observed an integer of the appropriate form, then, it remains to carry out a continued fraction expansion on $(\lfloor jM/r \rceil)/M$, which we can do with repeated application of the Euclidean algorithm.

Let us summarise the complexity of the algorithm. The dominating factors are the creating of the relevant superposition in time

$$\mathcal{O}((\log M)(\log p)^2) = \mathcal{O}((\log N(p)(\log p)^2 = \mathcal{O}((\log p)^3))$$

where the last equality follows from the easy property of the relevant semigroup family; that is, one has that $N(p)$ is at worst polynomial in p. Similarly, the application of QFT can be done in time $\mathcal{O}((\log p)^2)$, so we have the complexity estimate claimed at the outset. Note also that the algorithm succeeds provided an element of the cycle is observed after the first measurement, and that the second measurement gives an appropriate integer. Since both of these events occur with probability bounded below by a constant, the algorithm succeeds with probability $\Omega(1)$.

2. We prove correctness of the algorithm by proving that any values $start, end$ such that $start \leq n \leq end$ will return n, which we accomplish by strong induction on $k = start - end + 1$. To save on cumbersome notation we assume (g, ϕ) and r are fixed, and write

$$BinarySearch((g, \phi,), start, end, r) = BS(start, end)$$

First, suppose $k = 1$ and $start \leq n \leq end$. Either $n = start$ or $n = start + 1$, and we know that $mid = start$ after the floor function is applied. In the first case, $r * s_{g,\phi}(mid) = s_{g,\phi}(mid)$, so BS($start, mid$) is returned; but since $start = mid$, $start = n$ is returned. Otherwise,

one has $r * s_{g,\phi}(mid) \neq s_{g,\phi}(mid)$ and BS($mid + 1, end$) is returned, and we are done since $mid + 1 = end = n$.

Now for some $k > 1$ suppose all positive integers $start', end'$ such that $start' \leq n \leq end'$ and $end' - start' + 1 < k$ have BS($start', end'$)=n. We should like to show that an arbitrary choice of $start, end$ with $start \leq n \leq end$ and $end - start + 1 = k$ enjoys this same property. To see that it does we can again consider the two cases.

The algorithm first calculates $mid = \lfloor (end - start)/2 \rfloor$. Suppose $r * s_{g,\phi}(mid) = s_{g,\phi}(mid)$, then BS($start, mid$) is run. Since n is the smallest integer such that $r * s_{g,\phi}(n) = s_{g,\phi}())$ and $n \geq start$ by assumption, we know $start \leq n \leq mid$. Moreover, $mid - start + 1 < end - start + 1 < k$. By inductive hypothesis BS($start, mid$) returns n.

The other case is similar; this time, if $r * s_{g,\phi}(mid) \neq s_{g,\phi}(mid)$ we know $n \geq mid + 1$ by definition of n. We also know that $end - (mid + 1) + 1 = end - mid < end - mid + 1 = k$, so the algorithm returns BS($mid + 1, end$)=n by inductive hypothesis.

Notice that each time *BinarySearch* is called the calculation of $r * s_{g,\phi}(mid)$ is required. We know already that $s_{g,\phi}(mid)$ can be calculated in time $\mathcal{O}(\log mid (\log p)^2) = \mathcal{O}((\log p)^3)$. Given ϕ^r, $s_{g,\phi}(r_{g,\phi})$ and $s_{g,\phi}(mid)$, the calculation of $r * s_{g,\phi}(mid)$ requires evaluating an endomorphism and a semigroup multiplication - we have argued already that this can be done in time $\mathcal{O}((\log p)^2)$. By Lemma 3.9, we can compute $s_{g,\phi}(r)$ in time $\mathcal{O}(\log r (\log p)^2)$, so the total calculation is done in time $\mathcal{O}((\log p)^3)$ since $r < M$. Clearly, BinarySearch will be called $\mathcal{O}(\log M) = \mathcal{O}(\log p)$ times, since the size of the interval to search halves at each iteration, and we conclude that *BinarySearch* recovers the index in time $\mathcal{O}((\log p)^4)$.

□

3.4.2 From SDLP to GADLP

Let us assemble the components developed so far in this section into a reduction of SDLP to GADLP

Theorem 3.13. *Let $\{G_p\}_p$ be an easy family of semigroups, and fix p. Algorithm 6 solves SDLP with respect to a pair $(g, \phi) \in G_p \times End(G_p)$ given access to a GADLP oracle for the group action $(\mathbb{Z}_r, \mathcal{C}, \circledast)$. The algorithm runs*

in time $\mathcal{O}((\log p)^4)$, makes at most a single query to the GADLP oracle, and succeeds with probability $\Omega(1)$.

Algorithm 6 Solving SDLP with GADLP oracle

Input: $(g, \phi), s_{g,\phi}(x)$
Output: x

1: $r \leftarrow$PeriodRecovery$((g, \phi), M)$ for sufficiently large M
2: **if** r='Fail' **then**
3: **return** 'Fail'
4: **end if**
5: $n \leftarrow$BinarySearch$((g, \phi), 1, M, r)$
6: **if** $r * s_{g,\phi}(x) = s_{g,\phi}(x)$ **then**
7: $d \leftarrow s_{g,\phi}(n)$
8: $x' \leftarrow$ GADLP oracle applied to $d, s_{g,\phi}(x)$
9: $x \leftarrow n + x'$
10: **else**
11: $t \leftarrow$BinarySearch2$(s_{g,\phi}(x), 1, n, r)$
12: $x \leftarrow n - t$
13: **end if**
14: **return** x

Proof. Consider an instance of SDLP whereby we are given the pair (g, ϕ) and the value $s_{g,\phi}(x)$, for some x sampled uniformly at random from the set $\{1, .., n + r - 1\}$. We show that Algorithm 2 recovers x.

We start by applying Algorithms 1 and 2 to the pair (g, ϕ), recovering the pair n, r with constant probability. By Theorem 3.12, we can do so in time $\mathcal{O}((\log p)^4)$. Now, $s_{g,\phi}(x)$ might be in tail or in the cycle - but with our knowledge of r we can check in Step 6 which is true by verifying whether $r * s_{g,\phi}(x) = s_{g,\phi}(x)$. As discussed in the proof of Theorem 3.12, we can perform this check in time $\mathcal{O}((\log p)^3)$.

There are now two cases to consider. First, suppose that the check in Step 6 is passed, then $s_{g,\phi}(x)$ is in the cycle, and we may proceed as follows. Compute $s_{g,\phi}(n)$ in time $\mathcal{O}((\log p)^3)$, and query the GADLP oracle on input $s_{g,\phi}(n), s_{g,\phi}(x)$ (Step 8) to recover the $\mathbb{Z}_r$ element $[y]_r$. Without loss of generality the smallest positive representative of this class, say x', is such that $n + x' = x$, so we recover x in Step 9.

Now suppose that $s_{g,\phi}(x)$ is in the tail. We run the algorithm BinarySearch2 to recover t, the smallest integer such that $t * s_{g,\phi}(x)$ is invari-

ant under r. BinarySearch2 is precisely the same as Algorithm 5, except that in the verification step, we check if $r * (mid * s_{g,\phi}(x)) = mid * s_{g,\phi}(x)$, and if this check is not passed we search in the lower, rather than upper, interval. It is not hard to adapt the proof of correctness to show that BinarySearch2 does indeed return t in time $\mathcal{O}((\log p)^4)$. Moreover, by minimality of n and the additivity of $*$, we must have $x + t = n$, from which we recover $x = n - t$.

Finally, we note that the only probablistic step of this algorithm is the application of Algorithm 4, so we successfully recover x with the same success probability as Algorithm 4, and we are done.

□

In summary, we have an efficient quantum reduction from SDLP to GADLP: an efficient quantum procedure extracts the period r, and from there a classical procedure gives the index n. In order to recover x, it remains to either carry out an efficient classical procedure, or recover x with a single query to a GADLP oracle. Moreover, assuming the GADLP oracle always succeeds, the success probability is precisely that of Algorithm 1 - that is, bounded below by a positive constant independently of p.

Remark. The factor $\log p$ in the complexity estimate is really coming from the 'length' of a binary representation of G_p; that is, the number of bits required to represent G_p. In our case the size of G_p happens to be polynomial in p, and therefore the relevant 'length' is of order $\mathcal{O}(\log p)$. One might be used to seeing the complexity of similar period-finding routines, such as Shor's algorithm, presented as cubic in the length of a binary representation of the relevant parameters - see for example [37, Section 3.3.3]. In our case, the total complexity is quartic in the length of a binary representation, essentially because after the quantum part of the algorithm we still need to compute $\mathcal{O}(\log p)$ evaluations of the function s in order to compute the index. In a sense, then, we can think of this extra $\log p$ factor as the extra cost incurred from the slightly more complicated scenario inherent to the problem.

3.5 Quantum Algorithms for GADLP

Now that we have shown SDLP can be efficiently solved with access to an appropriate GADLP oracle it remains to examine the state of the art for GADLP. It is here that the Abelian Hidden Shift Problem (Definition 3.14)

comes in to play. Roughly speaking, we are given two injective functions f, g from a group A to a set S that differ by a constant 'shift' value, and our task is to recover the shift value.

It was first noticed by Stolbunov in [70] that GADLP gives rise to a so-called Abelian Hidden Shift Problem, or AHSP, though he notes that polynomial-time algorithms are only known for a few specific cases of these hidden shift problems. Later, Childs, Jao and Soukharev observed that the hidden shift problem defined by the group actions of [70] defines a special case of the Abelian Hidden Shift Problem, allowing for the application of general-purpose quantum algorithms of subexponential complexity. The proof of these facts does not hinge on any specifics of the context in [70]; nevertheless, in this section, we provide a context-specific proof of the reduction to AHSP, before discussing the best known algorithms. Our first task is to define AHSP.

Definition 3.14 (Abelian Hidden Shift Problem)**.** Given a public abelian group A and a set S, suppose two injective functions f, g hide some $s \in A$. The Abelian Hidden Shift Problem (AHSP) is to recover the group element s.

3.5.1 Group Actions to Hidden Shift

The following result is found more or less verbatim in, for example, [18]. We here give a context-specific reduction, for completeness.

Theorem 3.15. *Let $\{G_p\}_p$ be an easy family of semigroups and fix p. For some pair $(g, \phi) \in G_p \times End(G_p)$ let $(\mathbb{Z}_r, \mathcal{C}, \circledast)$ be the associated group action defined in Theorem 3.6. One can efficiently solve GADLP in $(\mathbb{Z}_r, \mathcal{C}, \circledast)$ given access to an AHSP oracle with respect to $\mathbb{Z}_r, \mathcal{C}$.*

Proof. Suppose we are given an instance of GADLP in $(\mathbb{Z}_r, \mathcal{C}, \circledast)$; that is, we are given a pair $(s_{g,\phi}(n+i), s_{g,\phi}(n+j)) \in \mathcal{C}$ for some $i, j \in \{1, ..., r\}$ and tasked with finding the unique $[k]_r \in \mathbb{Z}_r$ such that $[k]_r \circledast s_{g,\phi}(n+i) = s_{g,\phi}(n+j)$. Our strategy is to construct injective functions $f_A, f_B : \mathbb{Z}_r \to \mathcal{C}$ that hide $[k]_r$, and use the AHSP oracle to recover this value.

Set $f_A, f_B : \mathbb{Z}_r \to \mathcal{C}$ as $f_A([x]_r) = [x]_r * s_{g,\phi}(n+i)$ and $f_B([x]_r) = [x]_r * s_{g,\phi}(n+j)$. Then

$$\begin{aligned} f_B([x]_r) &= [x]_r * s_{g,\phi}(n+j) \\ &= [x]_r * ([k]_r * s_{g,\phi}(n+i)) \\ &= ([x]_r + [k]_r) * s_{g,\phi}(n+i) \\ &= f_A([x]_r + [k]_r) \end{aligned}$$

In other words, f_A, f_B hide $[k]_r$. To complete the setup of an instance of AHSP we require the functions to be injective, which follows from the action being free and transitive.

□

Note that we have in this case left out complexity estimates. This is because in order to give a full description of the functions f_A, f_B we need to compute the group $\mathbb{Z}_r$, which can be done efficiently with knowledge of r. However, since we have already described a method of recovering r, we will discuss the complexity in the full SDLP algorithm at the end of this section.

3.5.2 Hidden Shift Algorithms

We have finally arrived at the problem for which there are known quantum algorithms. The fastest known is of subexponential complexity, and is presented in [45, Proposition 6.1] as a special case of the Dihedral Hidden Subgroup Problem.

Theorem 3.16 (Kuperberg's Algorithm)**.** *There is a quantum algorithm that solves AHSP with respect to $\mathbb{Z}_r, \mathcal{C}$ with time and query complexity $2^{\mathcal{O}(\sqrt{\log r})}$.*

Kuperberg's algorithm also requires quantum space $2^{\mathcal{O}(\log r)}$. For a slower but less space-expensive algorithm, we can also use a generalised version of an algorithm due to Regev [62]. The generalised version appears in [18, Theorem 5.2].

Theorem 3.17 (Regev's Algorithm)**.** *There is a quantum algorithm that solves AHSP with respect to $\mathbb{Z}_r, \mathcal{C}$ with time and query complexity*

$$e^{\sqrt{2}+o(1)\sqrt{\ln r \ln\ln r}}$$

and space complexity $\mathcal{O}(poly(\log r))$.

We note that both Kuperberg's and Regev's algorithms succeed with constant probability.

3.5.3 Solving SDLP

We finish the section by stitching all the components together into an algorithm that solves SDLP. For brevity of exposition we include only complexity estimates for using Kuperberg's algorithm - but finding the bounds in the case of Regev's algorithm is very similar.

Theorem 3.18. *Let $\{G_p\}_p$ be an easy family of semigroups, and fix p. For any pair $(g, \phi) \in G_p \times End(G_p)$, there is a quantum algorithm solving SDLP with respect to (g, ϕ) with time and query complexity $2^{\mathcal{O}(\sqrt{\log p})}$.*

Proof. Let $(g, \phi) \in G_p \times End(G_p)$ and suppose we are given the value $s_{g,\phi}(x)$ for some x sampled uniformly from the set $\{1, ..., N\}$, where N is the size of $\mathcal{X}_{g,\phi}$. The following steps recover x:

1. Run Algorithms 4 and 5 on the pair (g, ϕ). By Theorem 3.12, with positive probability we recover the index and period of (g, ϕ), the pair (n, r), in time $\mathcal{O}((\log p)^4)$.
2. By Theorem 3.13, either we are done efficiently, or it remains to solve an instance of GADLP with respect to the group action $(\mathbb{Z}_r, \mathcal{C}, \circledast)$.
3. By Theorem 3.15, once we have computed the group action $(\mathbb{Z}_r, \mathcal{C}, \circledast)$ it remains to solve an instance of AHSP with respect to $\mathbb{Z}_r, \mathcal{C}$. This can be done with access to the index and period n, r.
4. Solve AHSP using Kuperberg's algorithm or Regev's algorithm.

In summary, the total quantum complexity of solving an SDLP instance for any pair in $G_p \times End(G_p)$ is either $\mathcal{O}((\log p)^4)$, or if a call to the GADLP oracle is required, $2^{\mathcal{O}(\sqrt{\log r})} = 2^{\mathcal{O}(\sqrt{\log p})}$ since G_p is from an easy family of semigroups. Depending on constants, we expect this latter term to dominate the complexity. Moreover, we note that since both our algorithm to extract the period and Kuperberg's algorithm succeed with constant probability, we expect our algorithm to succeed with constant probability also. □

3.6 Conclusion

We have provided the first dedicated analysis of SDLP, showing a reduction to a well-studied problem. Perhaps the most surprising aspect of the work is the progress made by a simple rephrasing; we made quite significant progress through rather elementary methods, and we suspect much more can be made within this framework.

The reader may notice that we have shown that SDPKE shares a very similar structure to that of a commutative action-based key exchange; it is known that breaking all such protocols can be reduced to the Abelian Hidden Shift Problem. Indeed, this work shows the algebraic machinery

of SDPKE is a step towards a candidate for what Couveignes calls a *hard homogenous space*, which was not known until now and provides a welcome alternative candidate not drawing from isogenies. Nevertheless, in line with the naming conventions in this area we propose a renaming of SDPKE to SPDH, which stands for 'Semidirect Product Diffie–Hellman', and should be pronounced *spud*.

Now that we have established that the structure associated with SDLP defines the type of group action upon which one might seek to build cryptography, a natural next step is to try to do so. Two immediate questions are as follows: can we compute the parameters n, r associated with a pair (g, ϕ) efficiently and classically to allow for efficient sampling; and what features of our group action provide a good reason for using it over other, more established group actions? Both of these questions, as we will see in the next chapter, turn out to have a surprisingly similar answer.

4 A Digital Signature Scheme

No legacy is so rich as honesty.

All's Well That Ends Well, Act III

In the final chapter we turn our attention to the application of the security problems we have thus far analysed. In particular, we should like to develop a signature scheme based on SDLP.

As far as we know, the only other existing example of a semidirect product-based signature scheme comes from Moldenhauer's thesis [49], in which the obvious analogue of ElGamal signatures is presented. Much like SDPKE, the security reduces to SCDH; in order to obtain more efficient signatures based directly on SDLP, we take a different, but still rather standard, approach. We outline some of the background for this method below.

4.1 Background

The general strategy we take to construct our signature is to construct an *identification protocol* consisting of an interactive conversation in which one party (the *prover*) wishes to prove their knowledge of some secret to a second party (the *verifier*); and then to apply the famous *Fiat-Shamir transform* to yield a non-interactive signature scheme. The approach is first discussed in [29], in which an interactive identification protocol based on problems in number theory is proposed. To each identification protocol a *transcript* recording the interaction is associated: the authors transform their interactive protocol to a non-interactive signature essentially by using cryptographic hash functions to 'simulate' a transcript produced by two parties, in such a way as to preclude the possibility of producing such a transcript fraudulently. The security of the resulting Fiat-Shamir signature scheme was not demonstrated until some 10 years later in [58]: here, the famous assertion the hash functions need to be modelled as *random oracles* is made. Indeed, it is shown in [32] that there exists an

identification protocol for which the Fiat-Shamir transform with respect to any deterministic hash function yields an insecure signature scheme, so we cannot dispense with this modelling assumption in our security proofs.

In [1] a more general result outlying sufficient conditions for which an identification protocol yields a secure signature scheme under the Fiat-Shamir transform is given. In the random oracle model, it turns out that it suffices for the identification scheme to be secure against passive impersonation attacks - that is, it should not be possible for a cheating prover to convince an honest verifier to accept their proof of knowledge, even if the cheating prover has access to honestly generated transcripts. Indeed, this is part of the appeal of the Fiat-Shamir transform - one inherits the security of the underlying identification protocol 'for free' provided one is willing to accept the necessity of the random oracle model.

Perhaps the most celebrated example of a signature of this type is the famous Schnorr signature [68], based on the difficulty of the discrete logarithm problem. The properties of the underlying identification protocol giving security against passive impersonation attacks turn out to be the properties needed in general - we use the exposition of [10], but the importance of the result is reflected by the wide variety of sources available (see for example [10]). Of particular importance is the notion that the identification scheme should be *zero-knowledge* - that is, the prover should reveal only that he knows a certain secret, but no further information about that secret. The notion of a zero-knowledge proof was introduced in [33][1], and this notion would famously be shown to be sufficiently powerful to capture *any* proof in [31][2] and completed in [7]. Nevertheless, in order to utilise these proof systems for cryptography there are various efficiency-related additional properties required of a zero-knowledge proof. As such, various other identification protocols (and therefore signature schemes) based on zero-knowledge proofs of discrete logarithms have also been proposed: a non exhaustive list includes the Okamoto protocol [57], improving security of Schnorr's ID protocol at a slight cost to efficiency; the Chaum-Evertse-van de Graaf protocol [16], allowing identification based on multiple simultaneous instances of the discrete logarithm problem; and the protocol of [21] based on the gap in difficulty

[1]The version cited is incorporated into a textbook of 2019, but it is important to note that the original paper is from 1985.

[2]This is from the same textbook already mentioned, but is originally from 1986.

between the computational and decisional variants of the Diffie-Hellman problems.

Of course, the same quantum algorithms threatening DHKE threaten all signature schemes described above and several others, and so today there is motivation to search for quantum-resistant signature schemes. Our design choices here are informed by the recent NIST call for additional signature candidates [52], which specifies 'short' (low memory-requiring) signatures not based on lattice problems (recall from Section 2.1 that all the current NIST candidates for signatures are based on lattice problems). Another key advantage of the Fiat-Shamir approach is that the resulting signatures tend to be comparatively short: as such, our research objective here, armed with our new group action structure developed in the previous chapter, is to construct an identification scheme secure against passive impersonation attacks relative to SDLP. We have already seen that, in the random oracle model, the Fiat-Shamir transform applied to such an identification scheme will yield a secure signature scheme with respect to SDLP as desired.

An obvious starting point is to attempt to adapt Schnorr-type identification protocols, but we are immediately faced with a problem: we can only add in the argument of $s_{g,\phi}()$, whereas classically one can both add and multiply in the exponent of a cyclic group generator g. This problem is not unique to us and indeed applies in some form to any group action: it is solved by Couviegnes in [23] in his paper defining cryptographic group actions, and independently by Rostostev and Stolbunov in [66]. The approach is to appeal to the classic illustrative zero-knowledge proof example of graph isomorphism, which differs from Schnorr identification schemes in that a cheating prover always escapes with probability $1/2$, requiring multiple parallel instances. In his doctoral thesis [71], Stolbunov defines a signature scheme based on this identificaton scheme by applying the Fiat-Shamir transform; the resulting signature, known as the Couveignes-Rostostev-Stolbunov (CRS) signature in deference to its independent discoverers, derives its group action from isogenies between elliptic curves.

CRS signatures did not receive much attention for a number of years, for two key reasons: first, since the scheme is group-action based the subexponential algorithm discussed in the previous chapter due to [18] applies[3]. This might in itself be tolerable; much more troubling is that the original version of CRS signatures are unacceptably slow. There

[3]Indeed, historically speaking [18] is a response to this body of work

has, however, been a resurgence of interest in schemes similar to CRS signatures following the discovery in [15] of a much faster isogeny-based group action; on the other hand, the computation of the class group is in general thought to be computationally difficult. In fact this is quite a significant problem: without random sampling the security proofs, which rely on group elements hiding secrets to have the appropriate distribution, break down. Two approaches to solving this problem have been suggested: in [25], one uses the 'Fiat-Shamir with aborts' technique developed by Lyubashevsky [46], at the cost of rendering the scheme considerably less time efficient; in [9], a state-of-the-art computation of a class group is performed and the resulting group action is used as the platform for a CRS signature. However, it is important to note that here the computation of *a* class group is achieved, and so one is restricted in terms of tweaking parameters. In particular, the introduction of new parameters would require another extremely expensive offline class group computation.

Notice that armed only with the results of the previous chapter, we ostensibly have the same problem. In other words, we know there exists parameters n, r such that a group of size n acts on a set which we can compute with knowledge of n and r - but at the moment we do not have a efficient, classical method of computing these parameters. Various options were trialled to solve this problem, with a positive result eventually yielded by allowing invertibility in the set-up of the group action. It turns out that doing so alters the structure rather dramatically, as we shall now see.

4.2 The Group Action with Invertibility

Fixing a full group G (that is, every element has an inverse), the group action of interest arises as usual from the study of the set $\{s_{g,\phi}(i) : i \in \mathbb{Z}\}$ for pairs $(g, \phi) \in G \ltimes Aut(G)$. The main difference from the group action of the previous chapter is that, since $G \ltimes Aut(G)$ is itself a group by Theorem 2.2, we have $1 \in \{s_{g,\phi}(i) : i \in \mathbb{Z}\}$, since there is some $n \in \mathbb{N}$ such that $(s_{g,\phi}(n), \phi^n) = (g, \phi)^n = (1, id)$. However, one cannot immediately deduce that this is the smallest integer for which $s_{g,\phi}$ is 1. Indeed, even if the order n of (g, ϕ) is the smallest integer such that $s_{g,\phi}(n) = 1$, we are not necessarily guaranteed that every integer up to n is mapped to a distinct elements of G by $s_{g,\phi}$. Before resolving these questions let us introduce some terminology.

Definition 4.1. Let G be a finite group and let $(g, \phi) \in G \ltimes Aut(G)$. The set

$$\mathcal{X}_{g,\phi} := \{s_{g,\phi}(i) : i \in \mathbb{Z}\}$$

is called the *cycle* of (g, ϕ), and its size is called the *period* of (g, ϕ).

In contrast with the non-invertible case we now do not have an 'index', or an analogue for it - put another way, the looping behaviour always starts at 0, rather than for some positive integer, as we will now see.

Recall from Definition 3.2 that we have a function $*$ such that $i * s_{g,\phi}(j) = s_{g,\phi}(i + j)$ for each $i, j \in \mathbb{N}$. As in the non-invertible case the looping behaviour arises as a consequence of the behaviour of $*$; that is, supposing $s_{g,\phi}(n) = 1$ for some $n \in \mathbb{Z}$, one has

$$\begin{aligned} s_{g,\phi}(n+1) = 1 * s_{g,\phi}(n) &= \phi^1(s_{g,\phi}(n))s_{g,\phi}(1) \\ &= \phi(1)s_{g,\phi}(1) \\ &= s_{g,\phi}(1) \end{aligned}$$

Notice that this time the looping behaviour arises slightly differently, since we know the value of $s_{g,\phi}(n)$ outright (whereas in the non-invertible case we know only that $s_{g,\phi}(n + r) = s_{g,\phi}(n)$). Generalising this idea we get a more complete picture of the structure of the cycle.

Theorem 4.2. *Let G be a finite group and fix $(g, \phi) \in G \ltimes Aut(G)$. Let n be the smallest positive integer for which $s_{g,\phi}(n) = 1$. One has that $|\mathcal{X}_{g,\phi}| = n$, and*

$$\mathcal{X}_{g,\phi} = \{1, g, ..., s_{g,\phi}(n-1)\}$$

Proof. First, let us demonstrate that the values

$$s_{g,\phi}(0), s_{g,\phi}(1), ..., s_{g,\phi}(n-1)$$

are all distinct. Suppose to the contrary that there exists $0 \leq i < j \leq n-1$ such that $s_{g,\phi}(i) = s_{g,\phi}(j)$; then some positive $k < n$ must be such that $i + k = j$. Indeed, invoking Definition 3.2, we know that $s_{g,\phi}(i + k) = i * s_{g,\phi}(k)$. In other words:

$$\begin{aligned} i * s_{g,\phi}(k) = s_{g,\phi}(j) &\Rightarrow \phi^i(s_{g,\phi}(k))s_{g,\phi}(i) = s_{g,\phi}(j) \\ &\Rightarrow \phi^i(s_{g,\phi}(k)) = 1 \\ &\Rightarrow s_{g,\phi}(k) = 1 \end{aligned}$$

which is a contradiction, since $k < n$. It remains to show that every integer is mapped by $s_{g,\phi}$ to one of these n distinct values - but this is trivial, since we can write any integer i as $kn + j$ for some integer k and $0 \leq j < n$. It follows that

$$s_{g,\phi}(i) = s_{g,\phi}(j)$$

where $s_{g,\phi}(j)$ is one of the n distinct values.

□

It follows that we can write $i * s_{g,\phi}(j) = s_{g,\phi}(i + j \mod n)$. In fact, the latter part of the above argument demonstrates something slightly stronger: not only is every integer mapped to one of n distinct values by $s_{g,\phi}$, but every member of a distinct residue class modulo n is mapped to the *same* distinct value. It is this basic idea that gives us our group action. The proof is similar to that of Theorem 3.6; we give it for completeness.

Theorem 4.3. *Let G be a finite group and fix $(g, \phi) \in G \ltimes Aut(G)$, and let n be the smallest positive integer such that $s_{g,\phi}(n) = 1$. Define the function $\circledast$ by*

$$\begin{aligned} \circledast : \quad & \mathbb{Z}_n \times \mathcal{X}_{g,\phi} \to \mathcal{X}_{g,\phi} \\ & [i]_n \circledast s_{g,\phi}(j) = i * s_{g,\phi}(j) \end{aligned}$$

The tuple $(\mathbb{Z}_n, \mathcal{X}_{g,\phi}, \circledast)$ is a free, transitive group action.

Proof. First, let us see that $\circledast$ is well-defined. Suppose $i \cong j \mod n$, then $i = j + kn$ for some $k \in \mathbb{Z}$. For some arbitrary $\mathcal{X}_{g,\phi}$, say $s_{g,\phi}(l)$ for $0 \leq l < n$, one has

$$\begin{aligned} i * s_{g,\phi}(l) &= (j + kn) * s_{g,\phi}(l) \\ &= j * s_{g,\phi}(l + kn) \\ &= j * s_{g,\phi}(l) \end{aligned}$$

We also need to verify that the claimed tuple is indeed a group action. In order to check that the identity in $\mathbb{Z}_n$ fixes each $\mathcal{X}_{g,\phi}$, by the well-definedness just demonstrated, it suffices to check that $0 * s_{g,\phi}(l) = s_{g,\phi}(l)$ for each $0 \leq l < n$ - which indeed is the case. For the compatibility of the action with modular addition, note that for $0 \leq i, j, k < n - 1$ one has

$$\begin{aligned} [k]_n \circledast ([j]_n \circledast s_{g,\phi}(i)) &= [k]_n \circledast s_{g,\phi}(i + j \mod n) \\ &= s_{g,\phi}(i + j + k \mod n) \\ &= [j + k]_n \circledast s_{g,\phi}(i) \end{aligned}$$

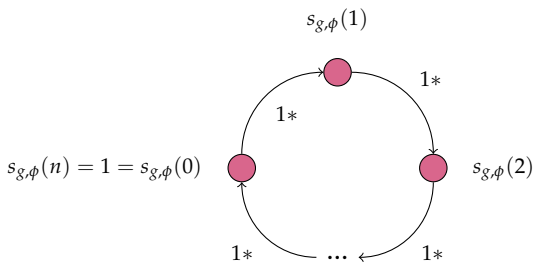


Figure 4.1: Structure of the exponents when invertibility required,, displaying the lack of 'tail' behaviour.

as required. It remains to check that the action is free and transitive. First, suppose $[i]_n \in \mathbb{Z}_n$ fixes each $s_{g,\phi}(j) \in \mathcal{X}_{g,\phi}$. By the above we can assume without loss of generality that $0 \leq i < n-1$, and we have $\phi^j(s_{g,\phi}(i))s_{g,\phi}(j) = s_{g,\phi}(j)$. It follows that $s_{g,\phi}(i) = 1$, so we must have $i = 0$ as required. For transitivity, for any pair $s_{g,\phi}(i), s_{g,\phi}(j)$ we have $[j-i]_n \circledast s_{g,\phi}(i) = s_{g,\phi}(j)$, and we are done.

□

4.2.1 Comparison with Semigroups

Compare Figure 4.1 representing the structure of $\mathcal{X}_{g,\phi}$ when (g,ϕ) is invertible with the corresponding Figure 3.1, when (g,ϕ) is not invertible. In the latter case, the looping can begin at any positive integer argument, but not when the argument of $s_{g,\phi}()$ is 0 since this would imply invertibility. Conversely, requiring invertibility forces the existence of an integer n for which $s_{g,\phi}(n) = 1$, and so the looping behaviour has to start at 0. For this reason we never see a tail when (g,ϕ) is invertible, and indeed we can think of SDLP as having two distinct variants - the case when the pair (g,ϕ) is invertible, and the case when it is not. We treat them as the same, since Theorem 3.13 shows that there is an efficient reduction from the non-invertible case to the invertible case. Put another way, we do not gain any security with the presence of a tail.

4.2.2 Computing Parameters

Recalling that the set $\mathcal{X}_{g,\phi}$ and the period n are a function of the pair (g,ϕ), we have actually shown the existence of a large family of group actions. Nevertheless, we have only really shown the existence of the

crucial parameter n - it is not necessarily clear how this value should be calculated. With this in mind let us conclude the section with a step in this direction:

Theorem 4.4. *Fix a pair $(g, \phi) \in G \ltimes Aut(G)$. Let n be the smallest integer such that $s_{g,\phi}(n) = 1$, then n divides the order of the pair (g, ϕ) as a group element in $G \ltimes Aut(G)$.*

Proof. Suppose $m = ord((g, \phi))$. Certainly $s_{g,\phi}(m) = 1$, and by definition one has $m \geq n$. We can therefore write $m = kn + l$, for $k \in \mathbb{N}$ and $0 \leq l < n$. It is not too difficult to verify that $s_{g,\phi}(x) = \phi^{x-1}(g)...\phi(g)g$ for any $x \in \mathbb{N}$. It follows that

$$s_{g,\phi}(m) = \phi^{kn}(s_{g,\phi}(l))\phi^{(k-1)n}(s_{g,\phi}(n))...\phi^{n}(s_{g,\phi}(n))s_{g,\phi}(n)$$

Since $s_{g,\phi}(m) = s_{g,\phi}(n) = 1$, we must have $s_{g,\phi}(l) = 1$. But $l < n$ and so $l = 0$ by the minimality of n, which in turn implies that $n|m$ as required. □

This, in a sense, is our main result, since the value of n is extremely restricted. To get a full sense of why it is so important, let us define the signature we have discussed more formally.

4.3 SPDH-Sign

4.3.1 Preliminaries

We have given some insight into what is meant by an identification scheme and what is meant by a signature scheme in Section 4.1. Nevertheless, in order to present the security proofs we need to define the appropriate security notions: this is taken care of in the following (rather lengthy) preliminaries section. The reader familiar with these concepts may wish to skip ahead to Section 4.3.2 and refer back to the definitions in what follows as necessary.

Proofs of Knowledge and Identification Schemes

Roughly speaking, the idea of the Fiat-Shamir class of signatures is as follows: we interactively convince an 'honest' party that we possess a certain secret. We can then transform this interactive paradigm to a non-interactive digital signature scheme by applying the Fiat-Shamir

transform. A primary motivation for this approach is that the resulting signature scheme inherits its security at rather low cost from security properties of the underlying interactive scheme - as such, it is necessary for us now to review some of these security notions.

First, let us define exactly what we mean by these interactive proof of knowledge protocols. The idea of communicating a 'secret' is neatly captured by the notion of a binary relation; that is, for two sets $\mathcal{W}$ and $\mathcal{S}$, consider a set $\mathcal{R} \subset \mathcal{W} \times \mathcal{S}$. Given a pair $(w, s) \in \mathcal{R}$, we say s is the *statement* and w is the *witness*. In general, for a given statement a party called the 'prover' wishes to demonstrate their knowledge of a valid witness (that is, given s we wish to prove that we possess a w such that $(w, s) \in \mathcal{R}$) to a party called the *verifier*. Of course, one can do this trivially by simply revealing the witness, so we add the crucial requirement that *no information about the witness is revealed*.

We refer more or less to this idea when discussing identification schemes, with the caveat that the prover should be able to compute an arbitrary pair of the binary relation. If the prover cannot generate an arbitrary pair of the binary relation, and instead is to demonstrate his knowledge of some given element of the binary relation, we have instead a 'zero-knowledge proof'. A notable class of zero-knowledge proofs are the so-called 'sigma protocols'. One can always turn a zero-knowledge proof into an identification scheme by providing the prover with an algorithm capable of generating an arbitrary pair of the binary relation; our definition of identification schemes in fact refers only to those arrived at by transforming a sigma protocol into an identification scheme.

Notice that the idea of a binary relation serves as a neat generalisation of the usual notion of a public and private key pair. The algorithm used by the identification scheme to generate binary relation instances is therefore denoted by `KeyGen`, and produces a pair (sk, pk). We also require, in some sense to be made precise later, that recovering an appropriate witness from a statement is computationally difficult.

Definition 4.5 (Identification Scheme)**.** Let $\mathcal{R} \subset \mathcal{S} \times \mathcal{P}$ be a binary relation. An identification scheme is a triple of algorithms (`KeyGen`,`P`,`V`), where

- `KeyGen` takes as input a security parameter n and generates a pair $(sk, pk) \in \mathcal{R}$, publishes pk, and passes sk to `P`
- `P` is an interactive algorithm initialised with a pair $(sk, pk) \in \mathcal{R}$

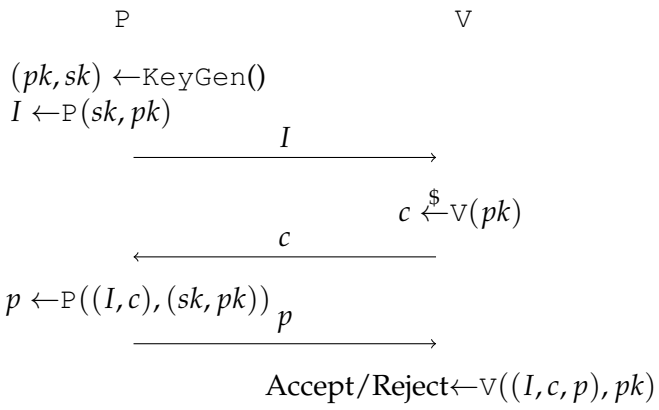


Figure 4.2: An identification scheme.

- V is an interactive algorithm initialised with a statement $pk \in \mathcal{P}$. After the interaction, V outputs a decision 'Accept' or 'Reject'.

The interaction of P and V runs as follows:

1. P generates a random value I, called the *commitment*, from the space of all possible commitments $\mathcal{I}$ and sends it to V
2. Upon receipt of I, V chooses a random value c, called a *challenge*, from the space of all possible challenges $\mathcal{C}$ at random and sends it to P
3. P responds with some value p, called a *response*
4. V calculates an 'Accept' or 'Reject' response as a function of (I, c, p) and the statement pk.

The interaction of P and V is depicted in Figure 4.2.

Definition 4.6. Let (KeyGen,P,V) be an identification scheme. The triple (I, c, p) of exchanged values between P and V is called a 'transcript'; if a prover (resp. verifier) generates I, p (resp c) with the algorithm P (resp. V), they are called 'honest'. An identification scheme is 'complete' if a transcript generated by two honest parties is always accepted by the verifier.

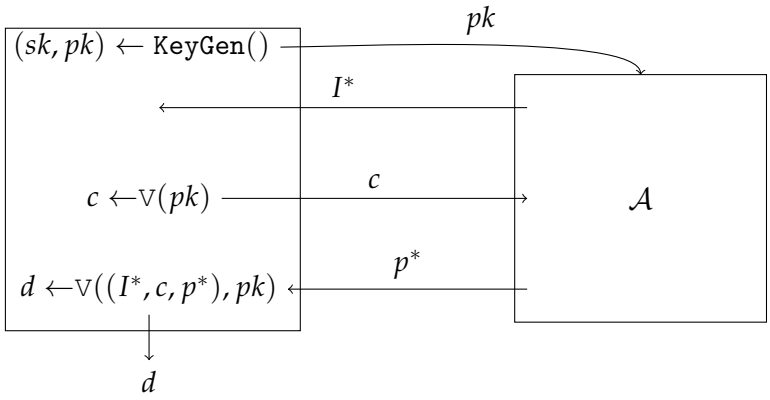


Figure 4.3: The direct attack game.

Turning our attention to the security of identification protocols, let us define the framework we wish to work with. As we will see later, it suffices for signature security to only consider identification schemes for which we have an honest verifier - in other words, it suffices to consider only a cheating prover. Let us do so in the form of the following attack games, which are [10, Attack Game 18.1] and [10, Attack Game 18.2] respectively.

Definition 4.7 (Direct Attack Game)**.** Let ID=(KeyGen,P,V) be an identification scheme and $\mathcal{A}$ be an adversary. Consider the following game:

1. The challenger obtains $(sk, pk) \leftarrow$KeyGen and passes pk to $\mathcal{A}$.

2. The adversary interacts with the challenger who generates responses with V. At the end, the challenger outputs 'Accept/Reject' as a function of the generated transcript and pk; the adversary wins the game if V outputs 'Accept'.

The Direct Attack game is depicted in Figure 4.3. We denote the advantage of the adversary in this game with ID as the challenger by dir-adv($\mathcal{A}$,ID).

Definition 4.8 (Eavesdropping Attack)**.** Let ID=(KeyGen,P,V) be an identification scheme and $\mathcal{A}$ be an adversary. Consider the following game:

1. The challenger obtains $(sk, pk) \leftarrow$KeyGen and passes pk to $\mathcal{A}$.

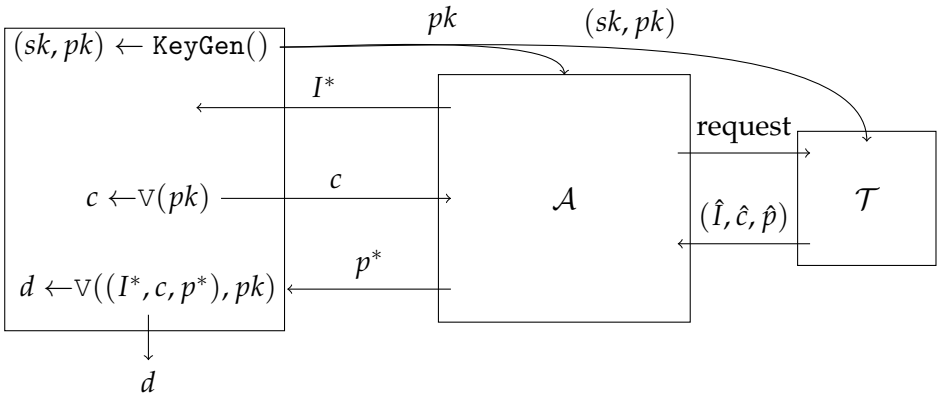


Figure 4.4: The eavesdropping attack game.

2. The adversary enters into an 'eavesdropping' phase, whereby they can request honestly-generated transcripts from a transcript oracle $\mathcal{T}$ possessing the same (sk, pk) pair generated in the previous step.

3. The adversary interacts with the challenger who generates responses with V. At the end, the challenger outputs 'Accept/Reject' as a function of the generated transcript and pk; the adversary wins the game if V outputs 'Accept'.

The Eavesdropping Attack game is depicted in Figure 4.4. We denote the advantage of the adversary in this game with ID as the challenger by eav-adv($\mathcal{A}$,ID).

In practice, given a concrete identification scheme it is possible to bound the advantage of an adversary in these games provided one can prove the following two properties hold for the identification scheme:

Definition 4.9. Let (KeyGen,P,V) be an identification scheme.

- The scheme has 'special soundness' if two transcripts with the same commitment and different challenges allow recovery of the witness sk; that is, if $(I, c, p), (I, c^*, p^*)$ are two transcripts generated with $(sk, pk) \leftarrow$KeyGen, there is an efficient algorithm taking these transcripts as input that returns sk.

- The scheme has 'special honest verifier zero knowledge' if, given a statement pk and a challenge c, there is an efficient algorithm to generate a passing transcript (I^*, c, p^*) with the same distribution as a legitimately generated transcript.

Before moving on there is one final security notion to explore. Notice that if the underlying binary relation of an identification scheme is such that one can easily recover a valid witness from the public statement, an adversary can easily succeed in either of the above games simply by honestly generating the proof p with the appropriate value of sk. We have loosely discussed the notion that recovering a witness should therefore be difficult; it is nevertheless so far not clear how precisely this difficulty is accounted for. In fact, there are a number of ways to get round this. For our purposes, and in our application of the Fiat-Shamir transform, we will invoke the system outlined in [10, Section 19.6]. The idea is basically thus: provided the properties in Definition 4.9 hold, it is possible to set up the security proof such that all the difficulty of recovering a witness is 'priced in' to the key generation algorithm. Again, we will need a precise definition to make this rigorous later on: the following is [10, Attack Game 19.2]

Definition 4.10 (Inversion Attack Game)**.** Let `KeyGen` be a key generation algorithm for a binary relation $\mathcal{R} \subset \mathcal{S} \times \mathcal{P}$ and $\mathcal{A}$ be an adversary. Consider the following game:

1. A pair (sk, pk) is generated by running `KeyGen`, and the value pk is passed to the adversary $\mathcal{A}$.
2. $\mathcal{A}$ outputs some $\hat{sk} \in \mathcal{S}$. The adversary wins if $(\hat{sk}, pk) \in \mathcal{R}$.

We denote the advantage of the adversary in this game with `kg` as the challenger by `inv-adv`$(\mathcal{A},$`kg`$)$.

Finally, we codify the obvious definition of a security game modelling SDLP:

Definition 4.11 (SDLP Game)**.** Let G be a finite group and fix a pair $(g, \phi) \in G \ltimes Aut(G)$ and $\mathcal{A}$ be an adversary. For some randomly selected integer x in $\{1, ..., n\}$, the adversary wins the Semidirect Discrete Log Game if they can recover x from (g, ϕ) and $s_{g,\phi}(x)$. The advantage of $\mathcal{A}$ in this game is denoted `sdlp-adv`$(\mathcal{A}, (g, \phi))$.

Signature Schemes

A 'signature scheme' is a triple of algorithms (`KeyGen, Sg, Vf`), where `KeyGen()` outputs a private-public key pair (sk, pk) upon input of a security parameter. For some space of messages $\mathcal{M}$, `Sg` takes as input sk

and some $m \in \mathcal{M}$, producing a 'signature' σ. Vf takes as input pk and a pair (m, σ), and outputs either 'Accept' or 'Reject'. We have the obvious correctness requirement that for a key pair (sk, pk) generated by KeyGen we can expect, for any $m \in \mathcal{M}$, that one has

$$\mathtt{Vf}(pk, (m, \mathtt{Sg}(sk, m))) = \text{Accept}$$

The security of a signature scheme is defined with respect to the following attack game, which is [10, Attack Game 13.1] (but is widely available).

Definition 4.12 (Chosen Message Attack)**.** Let S=(KeyGen,Sg,Vf) be a signature scheme and $\mathcal{A}$ be an adversary. Consider the following game:

1. The challenger obtains $(sk, pk) \leftarrow$KeyGen and passes pk to $\mathcal{A}$.
2. The adversary enters into an 'querying' phase, whereby they can obtain signatures $\sigma_i = \mathtt{Sg}(sk, m_i)$ from the challenger, for the adversary's choice of message m_i. The total number of messages queried is denoted Q.
3. The adversary submits their attempted forgery - a message-signature pair (m^*, σ^*) - to the challenger. The challenger outputs $\mathtt{Vf}(pk, (m^*, \sigma^*))$; the adversary wins if this output is 'Accept'.

The Chosen Message Attack game is depicted in Figure 4.5. We denote the advantage of the adversary in this game with S as the challenger by cma-adv($\mathcal{A}$,S).

A signature scheme S for which cma-adv($\mathcal{A}$,S) is bounded favourably[4] from above for any efficient adversary $\mathcal{A}$ is sometimes called euf-cma secure, or 'existentially unforgeable under chosen message attacks'.

It remains to briefly define the well-known notion of the Fiat-Shamir transform, initially presented in [29]:

Definition 4.13 (Fiat-Shamir)**.** Let ID=(KeyGen,P,V) be an identity scheme with commitment space $\mathcal{I}$ and $\mathcal{C}$. We define a signature scheme FS(ID) = (KeyGen,Sg,Vf) on the message space $\mathcal{M}$ given access to a public function $H : \mathcal{M} \times \mathcal{I} \to \mathcal{C}$:

1. KeyGen is exactly the key generation algorithm of ID and outputs a pair (sk, pk), where pk is made public

[4] 'Favourably' here usually means as a negligible function of a security parameter.

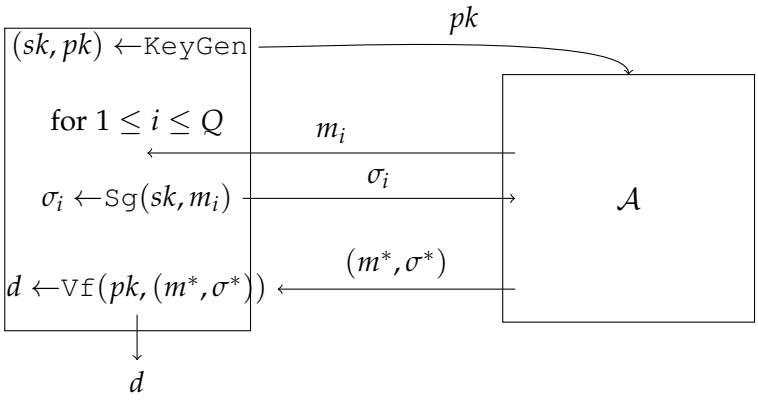


Figure 4.5: The chosen message attack game.

2. Sg takes as input $m \in \mathcal{M}$ and the key pair (pk, sk) and outputs a signature (σ_1, σ_2):
   $I \leftarrow$P$((sk, pk))$
   $c \leftarrow H(m, I)$
   $p \leftarrow$P$((I, c), (sk, pk))$
   $(\sigma_1, \sigma_2) \leftarrow (I, p)$
   return (σ_1, σ_2)

3. Vf takes as input a message-signature pair $(m, (\sigma_1, \sigma_2))$ and outputs a decision d, which is 'Accept' or 'Reject':
   $c \leftarrow H(I, \sigma_1)$
   $d \leftarrow$V$((\sigma_1, c, \sigma_2), pk)$
   return d

Intuitively, we can see that Sg is simulating an interactive protocol non-interactively with a call to the function H; in order to inherit the security properties of the identification scheme, this function H should have randomly distributed outputs on fresh queries and should be computationally binding - that is, it should be difficult to find a value $I' \neq I$ such that $H(m, I) = H(m, I')$; and given a commitment $c \in \mathcal{C}$ it should be difficult to find a message m and commitment $I \in \mathcal{I}$ such that $H(m, I) = c$. On the other hand, for correctness we need H to be deterministic on previously queried inputs. Such a function is modelled by a hash function under the random oracle model: in this model, it was famously demonstrated in [1] that a relatively modest security notion for the underlying identification scheme gives strong security proofs for the resulting

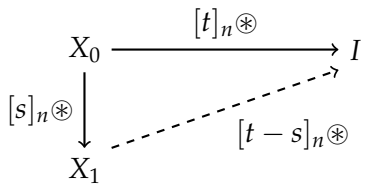


Figure 4.6: Paths to the commitment.

signature scheme. In our own security proof we use the slightly more textbook exposition presented in [10].

4.3.2 An Identification Scheme

Recall that our strategy is to set up an honest-verifier identification scheme, to which we can apply the well-known Fiat-Shamir heuristic and obtain strong security guarantees in the ROM. The central idea of this identification scheme is as follows: suppose we wish to prove knowledge of some secret $\mathbb{Z}_n$ element, say $[s]_n$. We can select an arbitrary element of $\mathcal{X}_{g,\phi}$, say X_0, and publish the pair $X_0, X_1 := [s]_n \circledast X_0$. An honest party wishing to verify our knowledge of the secret $[s]_n$ might invite us to commit to some group element $[t]_n$, for $[t]_n$ sampled uniformly at random from $\mathbb{Z}_n$. We can do this by sending the element $I = [t]_n \circledast X_0$ - note that as a consequence of the free and transitive properties, $[t]_n$ is the unique group element such that $I = [t]_n \circledast X_0$. However, with our knowledge of the secret $[s]_n$ and the commitment $[t]_n$, we can calculate the element $[p]_n = [t-s]_n$ such that $[p]_n \circledast X_1 = I$, where this equation holds by the group action axioms: one has $[t-s]_n \circledast ([s]_n \circledast X_0) = [t]_n \circledast X_0 = I$.

Interpreted graph-theoretically (as depicted in Figure 4.6), an honest verifier can ask to see one of two paths to the commitment value. Consider a dishonest party attempting to convince the verifier that they possess the secret $[s]_n$. In attempting to impersonate the honest prover, our dishonest party can generate their own value of $[t]_n$, and so can certainly provide the correct path in one of the two scenarios. Assuming, however, that recovering the appropriate group element is difficult, without knowledge of the secret $[s]_n$ this party succeeds in their deception with low probability.

This intuition gives us the following non-rigorous argument of security in the framework described in Section 4.3.1. First, recall that we are

in the honest verifier scenario, and so a challenge bit c will be 0 with probability $1/2$, in which case a cheating prover succeeds with probability 1. Supposing that ε is the probability of successfully recovering the value $[t-s]_n$, it follows that a cheating prover succeeds with probability $(1+\varepsilon)/2$ - that is, with probability larger than $1/2$. We can quite easily counter this by requiring that N instances are run at the same time. In this case, if N zeroes are selected the prover wins with probability 1 by revealing their dishonestly generated values of $[t]_n$ - otherwise, they are required to recover at least 1 value of $[t-s]_n$. Assuming for simplicity that the probability of doing so remains consistent regardless of the number of times such a value is to be recovered, since the honest verifier selects their challenges uniformly at random the cheating prover succeeds with probability

$$\frac{1}{2^N} + \sum_{i=1}^{2^N-1} \frac{\varepsilon}{2^N} = \frac{1}{2^N} + \varepsilon \frac{2^N - 1}{2^N}$$

which tends to ε as $N \to \infty$.

The actual proof of security operates within the security games defined in the preliminaries. As a step towards this formalisation, we need to specify the binary relation our identification scheme is based on. Choose some finite non-abelian group G: given a fixed pair $(g, \phi) \in G \times Aut(G)$ we are interested, by Theorem 4.3, in a subset $\mathcal{R}$ of $\mathbb{Z}_n, \mathcal{X}_{g,\phi}$, where n is the smallest integer such that $s_{g,\phi}(n) = 1$. In fact, legislating for N parallel executions of the proof of knowledge, to each tuple $(X_1, ..., X_N)$ is associated a binary relation

$$\mathcal{R} \subset \mathbb{Z}_n^N \times \mathcal{X}_{g,\phi}^N$$

where $(([s_1]_n, ..., [s_N]_n), (Y_1, ..., Y_N)) \in \mathcal{R}$ exactly when $(Y_1, ..., Y_N) = ([s_1]_n * X_1, ..., [s_N]_n * X_N)$.

With all this in mind let us define our identification scheme. The more rigorous presentation should not distract from the intuition that we describe N parallel executions of the game in Figure 4.6.

Protocol. SPDH-ID Let G be a finite non-abelian group and $(g, \phi) \in G \ltimes Aut(G)$. Suppose also that $n \in \mathbb{N}$ is the smallest integer such that $s_{g,\phi}(n) = 1$. The identification scheme $\texttt{SPDH-ID}_{g,\phi}(N)$ is a triple of algorithms

$$(\texttt{KeyGen}_{g,\phi}, \texttt{P}_{g,\phi}, \texttt{V}_{g,\phi})$$

such that

1. $\texttt{KeyGen}_{g,\phi}$ takes as input some $N \in \mathbb{N}$.
   $(X_1, ..., X_N) \leftarrow \mathcal{X}_{g,\phi}^N$
   $([s_1]_n, ..., [s_N]_n) \leftarrow \mathbb{Z}_n^N$
   $(Y_1, ..., Y_N) \leftarrow ([s_1]_n \circledast X_1, ..., [s_N]_n \circledast X_N)$

   $\texttt{KeyGen}_{g,\phi}$ outputs the public key $((X_1, ..., X_N), (Y_1, ..., Y_N))$ and passes the secret key $([s_1]_n, ..., [s_N]_n)$ to the prover $\texttt{P}_{g,\phi}$. The public key and the value of N used is published.

2. $\texttt{P}_{g,\phi}$ and $\texttt{V}_{g,\phi}$ are interactive algorithms that work as depicted in Figure 4.7:

Security

In this section we demonstrate that SPDH-ID is secure against eavesdropping attacks in the following sense: the advantage of an adversary in the eavesdropping attack game can be bounded by that of the adversary in the SDLP game. First, let us check that the desirable properties of an identification scheme hold:

Theorem 4.14. SPDH-ID *has the following properties:*

1. *Completeness*
2. *Special soundness*
3. *Special honest-verifier zero knowledge.*

Proof. Note that in order to prove each of these properties on the N-tuples comprising the transcripts generated by SPDH-ID, we need to prove that the properties hold for each component of the tuple; but since each component is independent of all the others, it suffices to demonstrate the stated properties for a single arbitrary component. In other words, we show that the stated properties hold when $N = 1$, and the general case immediately follows. For ease of notation in this single case we will write $S_0 = X, S_1 = Y$.

1. If $b = 0$ then $[p]_n = [t]_n$, and trivially we are done. If $b = 1$ then $[p]_n = [t - s]_n$; doing the bookkeeping we get that

$$\begin{aligned}[p]_n \circledast S_1 &= [p]_n \circledast ([s]_n \circledast S_0) \\ &= ([t - s]_n + [s]_n) \circledast S_0 \\ &= ([s]_n \circledast X) = I\end{aligned}$$

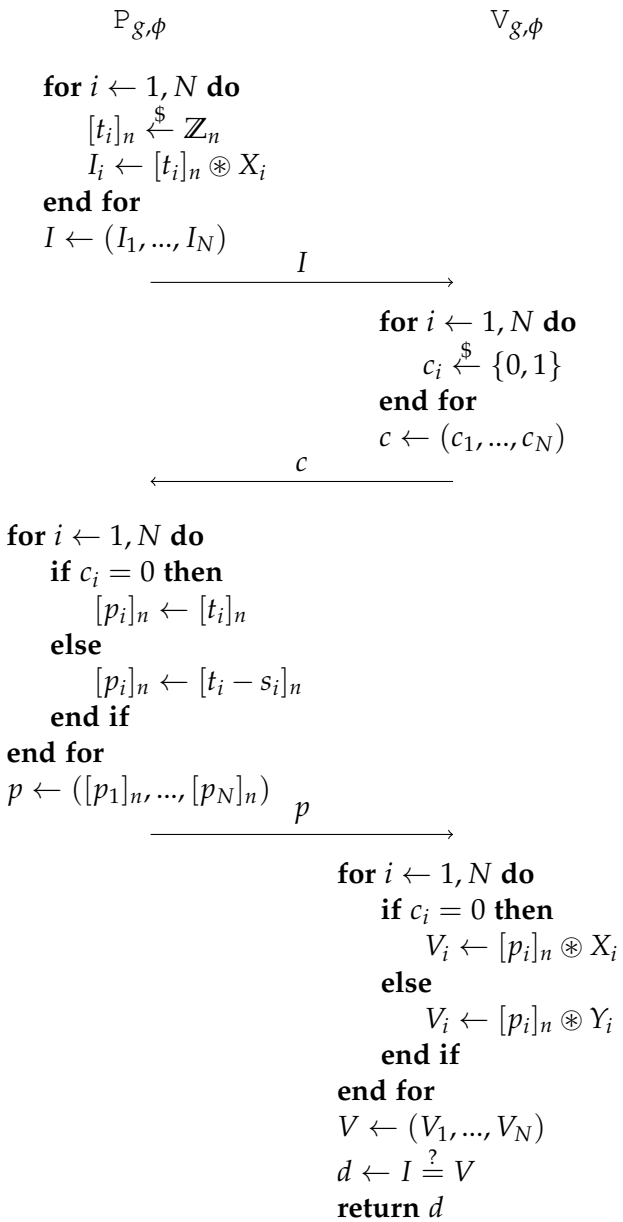


Figure 4.7: SPDH-ID

2. Two passing transcripts with the same commitment are $(I, 0, [t]_n)$ and $(I, 1, [t-s]_n)$. Labelling the two responses $[a]_n, [b]_n$, we recover the secret as $[a]_n - [b]_n$.

3. It suffices to show that one can produce passing transcripts with the same distribution as legitimate transcripts, but without knowledge of $[s]_n$. We have already discussed how to produce these transcripts; if a simulator samples $[t]_n$ uniformly at random, then the transcript $([t]_n \circledast S_b, b, [t]_n)$ is valid regardless of the value of b. Moreover, if $b = 0$, trivially the transcripts have the same distribution; if $b = 1$, since $[s]_n$ is fixed and $[t]_n$ is sampled uniformly at random, the distribution of a legitimate passing transcript is also uniformly random.

□

We are now ready to bound on the security of our identification scheme.

Theorem 4.15. *Let G be a finite abelian group and let $(g, \phi) \in G \ltimes Aut(G)$. For some $N \in \mathbb{N}$, consider the identification scheme $\texttt{SPDH-ID}_{g,\phi}(N)$ and an efficient adversary $\mathcal{A}$. There exists an efficient adversary $\mathcal{B}$ with $\mathcal{A}$ as a subroutine, such that with ε the advantage of the adversary $\mathcal{B}$ in the SDLP game, we have*

$$\texttt{eav-adv}(\mathcal{A}, \texttt{SPDH-ID}_{g,\phi}(N)) \leq \sqrt{\varepsilon} + \frac{1}{2^N}$$

Proof. This is just a straightforward application of two results in [10]. By [10, Theorem 19.14], since $\texttt{SPDH-ID}_{g,\phi}(N)$ has honest verifier zero knowledge, there exists an efficient adversary $\mathcal{B}'$ with $\mathcal{A}$ as a subroutine such that

$$\texttt{eav-adv}(\mathcal{A}, \texttt{SPDH-ID}_{g,\phi}(N)) = \texttt{dir-adv}(\mathcal{B}', \texttt{SPDH-ID}_{g,\phi}(N))$$

Moreover, let

$$\delta = \texttt{inv-adv}(\mathcal{B}', \texttt{KeyGen}_{g,\phi})$$

Since $\texttt{SPDH-ID}_{g,\phi}(N)$ has special soundness, [10, Theorem 19.13] gives

$$\texttt{dir-adv}(\mathcal{B}, \texttt{SPDH-ID}_{g,\phi}(N)) \leq \sqrt{\delta} + \frac{1}{M}$$

where M is the size of the challenge space. It is easy to see that $M = 2^N$; it remains to relate the quantities ε and δ. We do so eschewing some of the detail since the argument is straightforward; note that by

definition of the binary relation underpinning $\texttt{KeyGen}_{g,\phi}$, we can think of the inversion attack game as a security game in which one solves N independent SDLP instances in parallel. Call the advantage in this game $\texttt{N-sdlp-adv}(\mathcal{B}', (g, \phi))$, and suppose an adversary $\mathcal{B}$ in the standard SDLP attack game runs $\mathcal{B}'$ as an adversary. $\mathcal{B}$ can simply provide $\mathcal{B}'$ with N copies of its challenge SDLP instance, and succeeds whenever $\mathcal{B}'$ does. It follows that $\delta \leq \varepsilon$, and we are done.

□

4.3.3 A Digital Signature Scheme

It remains now to apply the Fiat-Shamir transform to our identification scheme.

Protocol (SPDH-Sign)**.** Let G be a finite non-abelian group and let $(g, \phi) \in G \times Aut(G)$ be such that n is the smallest integer for which $s_{g,\phi}(n)$=1. For any $N \in \mathbb{N}$ and message space $\mathcal{M}$, suppose we are provided a hash function $H : \mathcal{X}_{g,\phi}^N \times \mathcal{M} \to \{0,1\}^N$. We define the signature scheme

$$\texttt{SPDH-Sign}_{g,\phi}(N) = (\texttt{KeyGen}, \texttt{Sg}, \texttt{Vf})$$

as in Figure 4.8.

It is easy to see that given the identification scheme $\texttt{SPDH-ID}_{g,\phi}(N)$, the signature scheme $\texttt{SPDH-Sign}_{g,\phi}(N)$ is exactly FS($\texttt{SPDH-ID}_{g,\phi}(N)$). Before we can use this fact to prove the security of the signature, we require that the hash function gives outputs distributed at 'random', in some sense. This is accounted for by the 'Random Oracle Model': every time we wish to compute the hash function H, we suppose that an oracle function of the appropriate dimension selected at random is queried. Any party can query the random oracle at any time, and the number of these queries is kept track of. We also note that we do not in this paper account for the quantum-accessible random oracle model required for post-quantum security - equivalent security proofs in the quantum-accessible random oracle model are provided, for example, in [9].

With this heuristic in place we can prove the security of our signature scheme relative to SDLP with a simple application of [10, Theorem 19.15] and its corollaries:

$\texttt{KeyGen}(N)$:
for $i \leftarrow 1, N$ **do**
$X_i \xleftarrow{\$} \mathcal{X}_{g,\phi}$
$[s_i]_n \xleftarrow{\$} \mathbb{Z}_n$
$Y_i \leftarrow [s_i]_n \circledast X_i$
end for
$sk \leftarrow ([s_1]_n, ..., [s_N]_n)$
pk $\leftarrow$
$((X_1, ..., X_N), (Y_1, ..., Y_N))$
return (sk, pk)

$\texttt{Sg}(m, (sk, pk))$:
for $i \leftarrow 1, N$ **do**
$[t_i]_n \xleftarrow{\$} \mathbb{Z}_n$
$I_i \leftarrow [t_i]_n \circledast X_i$
end for
$I \leftarrow (I_1, ..., I_N)$
$c \leftarrow H(I, m)$
for $i \leftarrow 1, N$ **do**
if $c_i = 0$ **then**
$p_i \leftarrow [t_i]_n$
else
$p_i \leftarrow [t_i - s_i]_n$
end if
end for
$p \leftarrow (p_1, ..., p_N)$
$(\sigma_1, \sigma_2) \leftarrow (I, p)$
return (σ_1, σ_2)

$\texttt{Vf}(m, (\sigma_1, \sigma_2), pk)$:
$c \leftarrow H(\sigma_1, m)$
for $i \leftarrow 1, N$ **do**
if $c_i = 0$ **then**
$V_i \leftarrow p_i \circledast X_i$
else
$V_i \leftarrow p_i \circledast Y_i$
end if
end for
$V \leftarrow (V_1, ..., V_N)$
$d \leftarrow V \stackrel{?}{=} I$
return d

Figure 4.8: SPDH-Sign

Theorem 4.16. *Let G be a finite non-abelian group; $(g, \phi) \in G \ltimes Aut(G)$; and $n \in \mathbb{N}$ be the smallest integer such that $s_{g,\phi}(n) = 1$. Consider the chosen message attack game in the random oracle model, where Q_s is the number of signing queries made and Q_{ro} is the number of random oracle queries. For any efficient adversary $\mathcal{A}$ and $N \in \mathbb{N}$, there exists an efficient adversary $\mathcal{B}$ running $\mathcal{A}$ as a subroutine such that the signature scheme $\texttt{SPDH-Sign}_{g,\phi}(N)$ has*

$$\delta \leq \frac{Q_s}{n}(Q_s + Q_{ro} + 1) + \frac{Q_{ro}}{2^N} + \sqrt{(Q_{ro} + 1)\texttt{sdlp-adv}(\mathcal{B}, (g, \phi))}$$

where $\delta = \texttt{cma-adv}^{\texttt{ro}}(\texttt{SPDH-Sign}_{g,\phi}(N), \mathcal{A})$ is the advantage of the signature scheme in the random oracle model version of the chosen message attack game.

Proof. Applying [10, Theorem 19.15] and [10, Equation 19.21], since the underlying identification scheme has honest verifier zero knowledge there is an efficient adversary $\mathcal{B}'$ running $\mathcal{A}$ as a subroutine such that

$$\delta \leq \gamma Q_s(Q_s + Q_{ro} + 1) + \frac{Q_{ro}}{|\mathcal{C}|} + \sqrt{(Q_{ro} + 1)\texttt{inv-adv}(\mathcal{B}, \texttt{KeyGen}_{g,\phi})}$$

where γ is the probability that a given commitment value appears in a transcript, and $\texttt{KeyGen}_{g,\phi}$ is the key generation algorithm of the underlying identification scheme. Since choosing a random group element corresponds to choosing a random element of $\mathcal{X}_{g,\phi}$, each commitment value in $\mathcal{X}_{g,\phi}$ has probability $1/|\mathcal{X}_{g,\phi}| = 1/n$ of being selected. We have already seen in the proof of Theorem 4.15 that the advantage of an adversary in the inversion attack game against this key generation algorithm is bounded by the advantage in an SDLP attack game, and the result follows.

□

The above theorem provides a concrete estimate on the advantage of an adversary in the chosen message attack game; nevertheless, a plain English rephrasing is a useful reflection on these results. Essentially, we now know that the euf-cma security of our signature scheme is reliant on the integer n corresponding to the pair (g, ϕ), the size of N, and the difficulty of SDLP relative to the pair (g, ϕ). We can discount the reliance on N, which can be 'artificially' inflated as we please; note also that we can intuitively expect the size of n and the difficulty of SDLP for (g, ϕ) to be at least somewhat correlated, since a small value of n trivially renders the associated SDLP instance easy by brute force. In essence, then, we have shown that we can expect the signature scheme corresponding to (g, ϕ) to be secure provided the associated SDLP instance is difficult.

The Quantum Random Oracle Model

The *quantum random oracle model* is similar to the random oracle model, but allows the query of quantum states. In [11] it is shown that indeed there is a gap between these two models: there exist cryptosystems secure against adversaries with access to classical random oracles that become insecure when the adversary is given quantum access to the random oracle. Conditions for security of Fiat-Shamir type schemes in the quantum random oracle model are given in [28]; this framework is used for the security proof of the related signature scheme [9]. The proof is very similar to the one given above, in that the same security properties are used to bound signature security in terms of the difficulty of recovering a witness from a satement in a binary relation - which, again, is exactly SDLP The choice of which to present, then, is mainly stylistic.

4.4 A Candidate Group

We propose the following group of order p^3, where p is an odd prime, for use with `SPDH-Sign`.

Definition 4.17. Let p be an odd prime. The group G_p is defined by

$$G_p = \left\{ \begin{pmatrix} a & b \\ 0 & 1 \end{pmatrix} : a, b \in \mathbb{Z}_{p^2}, a \equiv 1 \mod p \right\}$$

As discussed in [22], this group is one of two non-abelian groups of order p^3 for an odd prime up to isomorphism. It has presentation

$$G_p = \langle x, y : y^p = 1, [x, y] = x^p =: z \in Z(G_p), z^p = 1 \rangle$$

as described in [47]; moreover, its automorphism group is known and has size $(p-1)p^3$ by [24, Theorem 3.1].

With respect to the various matters discussed in this chapter, we briefly present the advantages of employing such a group.

4.4.1 Sampling

Recall that our security proof for `SPDH-Sign` relied heavily on the underlying identification scheme being honest-verifier zero knowledge, which in turn relied on the 'fake' transcripts to have the same distribution as

honestly generated transcripts. For a pair (g, ϕ), it is therefore important to be able to sample uniformly at random from the group $\mathbb{Z}_n$, where n is the smallest integer for which $s_{g,\phi}(n) = 1$ - in our case, to do so it clearly suffices to compute n.

Here we recall Theorem 4.4, which tells us basically that, thinking of (g, ϕ) as a member of the semidirect product group $G \ltimes Aut(G)$, n must divide the order of (g, ϕ). We therefore have the following

Theorem 4.18. *Let $(g, \phi) \in G_p \times Aut(G_p)$, where p is an odd prime. Suppose n is the smallest integer for which $s_{g,\phi}(n) = 1$. Then*

$$n \in \{p^i(p-1)^j : 0 \leq i \leq 6, j \in \{0, 1\}\}$$

Proof. By Theorem 4.4 we know that $n | ord((g, \phi))$, and it is standard that

$$ord((g, \phi)) \quad | \quad |G_p \ltimes Aut(G)|$$

. We know from the discussion at the outset of this section that $|G_p| = p^3$ and $|Aut(G_p)| = p^3(p-1)$. It follows that $n | p^3 p^3 (p-1)$. Since p is prime, and assuming that (g, ϕ) is not the identity, the claimed set is a complete list of divisors of $p^6(p-1)$ - excluding $p^6(p-1)$ itself, since this would imply $G_p \ltimes Aut(G_p)$ is cyclic. □

It follows that for an arbitrary pair (g, ϕ) in $G_p \ltimes Aut(G_p)$, in order to compute the smallest n for which $s_{g,\phi}(n) = 1$, and therefore the group $\mathbb{Z}_n$, one has to compute $s_{g,\phi}(i)$ for at most 12 values of i. Moreover, by square-and-multiply each such computation requires $\mathcal{O}(\log p)$ applications of the group operation in the semidirect product group. In other words, we can compute a complete description of $\mathbb{Z}_n$ efficiently.

4.4.2 Security

SDLP

Trivially one can carry out successful forgeries if one can solve SDLP with respect to G_p, since one can simply recover the secret key associated to the public key. The results on solving the semigroup version of SDLP in the previous chapter apply here in the obvious way: since we have no tail to deal with as in the previous case, SDLP is precisely GADLP here. There is therefore (since the group acting on the cycle has size n) a quantum algorithm solving SDLP with respect to G_p in time at worst

$2^{\mathcal{O}(\sqrt{\log n})} = 2^{\mathcal{O}(\sqrt{\log p})}$, where the equality follows since we know n is polynomial in p by Theorem 4.18.

Taking the security parameter to be the length of an input, we can represent a pair $(g, \phi) \in G_p \ltimes Aut(G_p)$ with a bitstring of length $\mathcal{O}(\log p^2) = \mathcal{O}(\log p)$. Asymptotically, then, with k as the security parameter we estimate the time complexity of the main quantum attack on SDLP as $2^{\mathcal{O}(\sqrt{k})}$. On the other hand, in order to derive a concrete estimate for specific security parameters - say, those required by NIST - one would have to check the associated constants much more carefully. Although this is outside the scope of this thesis, we refer the reader to [15, Section 7.2 'Subexponential vs Practical'] for an idea of type of spirited research carried out in pursuit of a satisfactory resolution to deriving concrete security estimates - one should note, however, that this exposition deals with specific artefacts of the isogeny framework.

SCDH

As we have discussed, classically speaking there appears to be a gap between the difficulty of SDLP and SCDH. Nevertheless, should this prove not to be the case, we point out the resistance to some of the typical attacks of SCDH.

The Dimension Attack. It is clear from the presentation of G_p that it must possess an element of order p^2, and so by Corollary 2.18 we have that the dimension attack runs in time $\mathcal{O}((p+1)^2)$. Again taking the security parameter to be input length, since the G_p elements can be represented by a bitstring of order $4 \log p^2 = 8 \log p$, the dimension attack runs in time $\mathcal{O}(2^{2k/8}) = \mathcal{O}(2^{k/4})$.

The Telescoping Attack. In general, the explicit method of deducing $s_{g,\phi}(x+y)$ from $s_{g,\phi}(y)$ and $\phi^x(g)$ relies on the group G being the abelian group of a matrix alegbra over a field under addition. In particular, an extension outside of this linear context is not known - we would expect, however, that such an extension would rely on equation solving techniques available only in an algebra over a field, rather than over a ring, and therefore that arguments on the efficiency of a representation discussed above would also apply.

4.4.3 Efficiency.

Multiplication in G_p consists of 8 multiplication operations and 4 addition operations in $\mathbb{Z}_{p^2}$, for a total of $\mathcal{O}(8\log p^2) = \mathcal{O}(\log p)$ operations. Assuming that applying an automorphism ϕ has about the same complexity as multiplication[5]. It follows by standard square-and-multiply techniques that calculating $s_{g,\phi}$ and evaluating the group action is very roughly of complexity $\mathcal{O}((\log p)^2)$.

The signatures consist of N elements of $\mathcal{X}_{g,\phi}$ and N elements of $\mathbb{Z}_n$. Since $\mathcal{X}_{g,\phi} \subset G_p$ we can represent $\mathcal{X}_{g,\phi}$ elements as bitstrings of length $4\log\left(p^2\right) = 8\log p$; and since $n = p^i(p-1)^j$ for some $1 \leq i \leq 5$ and $0 \leq j \leq 1$, $\mathbb{Z}_n$ elements can be represented by bitstrings of length $\log p^i(p-1)^j$. It follows that we get signatures of length

$$N((8+i)\log p + j\log(p-1))$$

4.5 Conclusion

We have given a constructive proof that a few elementary definitions give rise to a free, transitive group action; such a group action naturally gives rise to an identification scheme and a signature scheme. Moreover, well-known tools allow us to phrase the security of this signature scheme in terms of the semidirect discrete logarithm problem, which is itself a special case of Couveignes' Vectorisation Problem.

Our main contributions are as follows: firstly, the generality of the construction gives an unusually diverse family of signature schemes - indeed, a signature scheme of the `SPDH-Sign` type is defined for each finite group. Much further study on the relative merits of different choices of finite non-abelian group in different use cases is required to fully realise the potential of this diversity.

Second, our Theorem 4.4 essentially gives us information about how to compute the group in our group action. In Theorem 4.18, we saw one particular case where the result was enough to completely describe how to efficiently compute the group, thereby yielding an example of a group-action based key exchange in which efficient sampling is possible from the whole group, without appealing to techniques inducing additional overhead, most notably the 'Fiat-Shamir with aborts' technique of Lyubashevsky.

[5]This is indeed the case if the automorphism is inner as we have seen in Chapter 2.

The chapter notably has not addressed concrete security estimates, nor recommended parameter sizes for a signature scheme. In order to do so we would need to carefully check the constants in the asymptotic security estimates - we consider the scale of this task, along with that of providing an implementation of the scheme, as sufficient to merit a separate paper.

5 Conclusion

> The wheel is come full circle: I am here.
>
> *King Lear*, Act V

Let us recall the research questions set out at the start of this thesis.

1. For which choices of group is SDPKE secure?
2. For which choices of group do schemes related to SDPKE have useful applications?

In a way we have achieved both more and less than we set out to. With respect to the first question we have consolidated and augmented the literature surrounding the complexity of SCDH, which certainly can be thought of as heavily related, if not precisely equivalent, to the security of SDPKE.

More than this, we have systematised the analysis of SCDH. We have corrected the misconception common to the literature that "one does not have to solve the underlying algorithmic problem to break the scheme"[1] - in fact, algorithms are given precisely to solve SCDH, and the lack of algorithms solving SDLP in the same group demonstrates only that there appears to be a gap between the difficulty of these two problems, in contrast with the classical case. We have also made rigorous the nascent idea that the search for a choice of group for SDPKE is precisely the search for groups, or semigroups, admitting only large-dimensional faithful representations. A good case study in the lack of awareness of this fact is the case of MOBS [60]: a switch to a semigroup intended to defeat the telescoping attack failed, but accidentally recommended a semigroup admitting only large-dimensional faithful representations.

On the other hand, we have not addressed the decisional variant of SCDH, which by modern standards is what should be thought of as

[1] This particular claim is of [65], but similar sentiments are expressed throughout the cryptanalytic literature.

the 'security' of SDPKE. The experimental evidence presented in [35] suggesting that this decisional variant may indeed be difficult, compared with the effectiveness of the dimension attack against the semigroup chosen therein, suggests that there may be a gap between this decisional variant and SCDH. Really, this warrants its own separate line of research. We also have developed a much less rich classification of the telescoping attack compared to the dimension attack, and really we still do not have a comparable method of applying the telescoping attack in general. We should therefore exercise caution before concluding that certain types of p-group, despite their resistance to the telescoping attack, are a 'secure' choice of group for SDPKE.

Evaluating our contribution with respect to the second question is more complicated. In a sense, Section 4.4 tells us that the answer is a certain group of order p^3; but this does not reflect the full picture of how such a result was derived. Indeed, we have answered a more involved question than that which was originally posed: namely, 'how can we use the structure associated to SDPKE to derive different types of cryptosystem?'. It turned out that one can do so by showing that the structure associated to SDPKE occurs as an example of a cryptographic group action; this had the twin effect of providing the first dedicated analysis of SDLP.

Key theoretical questions remain here, too: for example, there is currently no classical way in the semigroup version of the group action to compute the index and period, precluding efficient sampling in the relevant group action. Moreover, in order to derive concrete parameter estimates for the signature scheme, one would have to carefully check the constants implied by the asymptotic estimates of efficiency and security - particularly with respect to the quantum algorithm of Kuperberg.

How ready is semidirect product cryptography for an approaching post-quantum world? This thesis represents the most comprehensive investigation of the subject yet written; despite important theoretical progress, any standardisation or commercialisation of techniques within would require the efforts of a much larger team. Indeed, as well as further research on the various technical open problems generated in the course of this thesis, a large amount of implementation work remains.

We close with some slightly more 'big picture' remarks relevant to the field of post-quantum cryptography at large. Firstly, the 'back-and-forth' type papers making up the bulk of the previous literature in this area, while important, were not the most effective in gaining insight into the nature of the problem itself. In other words, one should remain analytical rather than competitive. Secondly, insofar as post-quantum cryptography

consists of distinct subfields, dialogue between these subfields is important. Our relatively little-known area of research has been effectively revolutionised by borrowing techniques from the more mainstream area of isogeny-based cryptography; conversely, applying these techniques allowed us to contribute an alternative solution to a problem inherent to isogeny-based cryptography. In the opinion of this author, this same spirit of collaboration and open dialogue should serve as our blueprint for navigating the coming post-quantum world.

Bibliography

[1] Michel Abdalla et al. 'From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security'. In: *Advances in Cryptology—EUROCRYPT 2002: International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam, The Netherlands, April 28–May 2, 2002 Proceedings 21*. Springer. 2002, pp. 418–433.

[2] Christopher Battarbee, Delaram Kahrobaei and Siamak F Shahandashti. 'Semidirect Product Key Exchange: the State of Play'. In: *arXiv preprint arXiv:2202.05178, to appear in Journal of Algebra and Applications* (2022).

[3] Christopher Battarbee, Delaram Kahrobaei and Siamak F. Shahandashti. 'Cryptanalysis of Semidirect Product Key Exchange Using Matrices Over Non-Commutative Rings'. In: *Mathematical Cryptology* 1.2 (Mar. 2022), pp. 2–9. URL: https://journals.flvc.org/mathcryptology/article/view/130528.

[4] Christopher Battarbee et al. 'A Subexponential Quantum Algorithm for the Semdirect Discrete Logarithm Problem'. In: *4th NIST PQC Standardization Conference 2022 https://csrc.nist.gov/csrc/media/Events/2022/fourth-pqc-standardization-conference/documents/papers/a-subexpoenential-quantum-algorithm-pqc2022.pdf* (2022), pp. 1–27.

[5] Christopher Battarbee et al. 'On the efficiency of a general attack against the MOBS cryptosystem'. In: *Journal of Mathematical Cryptology* 16.1 (2022), pp. 289–297.

[6] Christopher Battarbee et al. 'SPDH-Sign: towards Efficient, Post-quantum Group-based Signatures'. In: *PQCrypto 2023: 14th Int'l Conference on Post-Quantum Cryptography (to appear)*. available as arXiv preprint arXiv:2304.12900. 2023.

[7] Michael Ben-Or et al. 'Everything provable is provable in zero-knowledge'. In: *Advances in Cryptology—CRYPTO'88: Proceedings 8*. Springer. 1990, pp. 37–56.

[8] Ward Beullens. *Breaking Rainbow Takes a Weekend on a Laptop*. Cryptology ePrint Archive, Paper 2022/214. https://eprint.iacr.org/2022/214. 2022. URL: https://eprint.iacr.org/2022/214.

[9] Ward Beullens, Thorsten Kleinjung and Frederik Vercauteren. 'CSI-FiSh: efficient isogeny based signatures through class group computations'. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2019, pp. 227–247.

[10] Dan Boneh and Victor Shoup. 'A graduate course in applied cryptography'. In: *Draft 0.5* (2020).

[11] Dan Boneh et al. 'Random oracles in a quantum world'. In: *Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings 17*. Springer. 2011, pp. 41–69.

[12] Stephen Boyd and Lieven Vandenberghe. *Introduction to applied linear algebra: vectors, matrices, and least squares*. Cambridge university press, 2018.

[13] Daniel Brown, Neal Koblitz and Jason Legrow. 'Cryptanalysis of 'MAKE''. In: *eprint.iacr.org.2021.465* (2021).

[14] Wouter Castryck and Thomas Decru. 'An efficient key recovery attack on SIDH (preliminary version)'. In: *Cryptology ePrint Archive* (2022).

[15] Wouter Castryck et al. 'CSIDH: an efficient post-quantum commutative group action'. In: *Advances in Cryptology–ASIACRYPT 2018: 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2–6, 2018, Proceedings, Part III 24*. Springer. 2018, pp. 395–427.

[16] David Chaum, Jan-Hendrik Evertse and Jeroen Van De Graaf. 'An improved protocol for demonstrating possession of discrete logarithms and some generalizations'. In: *Advances in Cryptology—EUROCRYPT'87: Workshop on the Theory and Application of Cryptographic Techniques Amsterdam, The Netherlands, April 13–15, 1987 Proceedings 6*. Springer. 1988, pp. 127–141.

[17] Lily Chen, D Moody and YK Liu. 'NIST post-quantum cryptography standardization'. In: *Transition* 800.131A (2017), p. 164.

[18] Andrew Childs, David Jao and Vladimir Soukharev. 'Constructing elliptic curve isogenies in quantum subexponential time'. In: *Journal of Mathematical Cryptology* 8.1 (2014), pp. 1–29.

[19] Andrew M Childs and Gábor Ivanyos. 'Quantum computation of discrete logarithms in semigroups'. In: *Journal of Mathematical Cryptology* 8.4 (2014), pp. 405–416.

[20] Andrew M Childs and Wim Van Dam. 'Quantum algorithms for algebraic problems'. In: *Reviews of Modern Physics* 82.1 (2010), p. 1.

[21] Jae Cha Choon and Jung Hee Cheon. 'An identity-based signature from gap Diffie-Hellman groups'. In: *Public Key Cryptography—PKC 2003: 6th International Workshop on Practice and Theory in Public Key Cryptography Miami, FL, USA, January 6–8, 2003 Proceedings 6*. Springer. 2002, pp. 18–30.

[22] Keith Conrad. *Groups of Order p^3*. URL: https://kconrad.math.uconn.edu/blurbs/grouptheory/groupsp3.pdf.

[23] Jean-Marc Couveignes. 'Hard homogeneous spaces'. In: *Cryptology ePrint Archive, https://eprint.iacr.org/2006/291.pdf* (2006).

[24] MJ Curran. 'The automorphism group of a nonsplit metacyclic p-group'. In: *Archiv der Mathematik* 90 (2008), pp. 483–489.

[25] Luca De Feo and Steven D Galbraith. 'SeaSign: compact isogeny signatures from class group actions'. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2019, pp. 759–789.

[26] Whitfield Diffie and Martin Hellman. 'New directions in cryptography'. In: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654.

[27] Jintai Ding and Dieter Schmidt. 'Rainbow, a new multivariable polynomial signature scheme'. In: *International conference on applied cryptography and network security*. Springer. 2005, pp. 164–175.

[28] Jelle Don et al. 'Security of the Fiat-Shamir transformation in the quantum random-oracle model'. In: *Advances in Cryptology–CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part II 39*. Springer. 2019, pp. 356–383.

[29] Amos Fiat and Adi Shamir. 'How to Prove Yourself: Practical Solutions to Identification and Signature Problems.' In: *Crypto*. Vol. 86. Springer. 1986, pp. 186–194.

[30] Eduarda S V Freire et al. 'Non-interactive key exchange'. In: *Public-Key Cryptography–PKC 2013: 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26–March 1, 2013. Proceedings 16*. Springer. 2013, pp. 254–271.

[31] Oded Goldreich, Silvio Micali and Avi Wigderson. 'Proofs that yield nothing but their validity and a methodology of cryptographic protocol design'. In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. 2019, pp. 285–306.

[32] Shafi Goldwasser and Yael Tauman Kalai. 'On the (in) security of the Fiat-Shamir paradigm'. In: *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.* IEEE. 2003, pp. 102–113.

[33] Shafi Goldwasser, Silvio Micali and Chales Rackoff. 'The knowledge complexity of interactive proof-systems'. In: *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. 2019, pp. 203–225.

[34] Dima Grigoriev and Vladimir Shpilrain. 'Tropical cryptography II: extensions by homomorphisms'. In: *Communications in Algebra* 47.10 (2019), pp. 4224–4229.

[35] Maggie Habeeb et al. 'Public key exchange using semidirect product of (semi) groups'. In: *International Conference on Applied Cryptography and Network Security*. Springer. 2013, pp. 475–486.

[36] Jiao Han and Jincheng Zhuang. 'DLP in semigroups: Algorithms and lower bounds'. In: *Journal of Mathematical Cryptology* 16.1 (2022), pp. 278–288.

[37] Mika Hirvensalo. *Quantum computing*. Springer Science & Business Media, 2003.

[38] John Mackintosh Howie. *Fundamentals of semigroup theory*. 12. Oxford University Press, 1995.

[39] Steve Isaac and Delaram Kahrobaei. 'A closer look at the tropical cryptography'. In: *International Journal of Computer Mathematics: Computer Systems Theory* (2021), pp. 1–6.

[40] Gordon James and Martin W Liebeck. *Representations and characters of groups*. Cambridge university press, 2001.

[41] GJ Janusz. 'Faithful Representations of p-Groups at Characteristic p'. In: *Representation Theory of Finite Groups and Related Topics* 21 (1971), p. 89.

[42] David Jao and Luca De Feo. 'Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies'. In: *International Workshop on Post-Quantum Cryptography*. Springer. 2011, pp. 19–34.

[43] Delaram Kahrobaei and Vladimir Shpilrain. 'Using semidirect product of (semi) groups in public key cryptography'. In: *Conference on Computability in Europe*. Springer. 2016, pp. 132–141.

[44] Ki Hang Kim and Fred W Roush. 'Linear representations of semigroups of Boolean matrices'. In: *Proceedings of the American Mathematical Society* 63.2 (1977), pp. 203–207.

[45] Greg Kuperberg. 'A subexponential-time quantum algorithm for the dihedral hidden subgroup problem'. In: *SIAM Journal on Computing* 35.1 (2005), pp. 170–188.

[46] Vadim Lyubashevsky. 'Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures'. In: *International Conference on the Theory and Application of Cryptology and Information Security*. Springer. 2009, pp. 598–616.

[47] Ayan Mahalanobis. 'The MOR cryptosystem and extra-special p-groups'. In: *Journal of Discrete Mathematical Sciences and Cryptography* 18 (2015), pp. 201–208.

[48] Luciano Maino and Chloe Martindale. 'An attack on SIDH with arbitrary starting curve'. In: *Cryptology ePrint Archive* (2022).

[49] Anja Moldenhauer. 'A group theoretical ElGamal cryptosystem based on a semidirect product of groups and a proposal for a signature protocol'. In: *AMS Contemporary Mathematics* 633 (2015), pp. 97–113.

[50] Chris Monico. 'Remarks on MOBS and cryptosystems using semidirect products'. In: *arXiv preprint arXiv:2109.11426* (2021).

[51] Christopher J Monico. *Semirings and semigroup actions in public-key cryptography*. University of Notre Dame, 2002.

[52] Dustin Moody. *The Beginning of the End: The First NIST PQC Standards*. 2022. URL: https://csrc.nist.gov/csrc/media/Presentations/2022/the-beginning-of-the-end-the-first-nist-pqc-standa/images-media/pkc2022-march2022-moody.pdf.

[53] Alexei Myasnikov and Vitaliĭ Roman'kov. 'A linear decomposition attack'. In: *Groups Complexity Cryptology* 7.1 (2015), pp. 81–94.

[54] Michael A Nielsen and Isaac L Chuang. 'Quantum computation and quantum information'. In: *Phys. Today* 54.2 (2001), p. 60.

[55] *NIST PQC*. https://csrc.nist.gov/projects/post-quantum-cryptography. Accessed: 2023-06-21.

[56] Ryan O'Donnell. *Quantum Computation and Quantum Information*. 2018. URL: https://www.cs.cmu.edu/~odonnell/quantum18/.

[57] Tatsuaki Okamoto. 'Provably secure and practical identification schemes and corresponding signature schemes.' In: *Crypto*. Vol. 92. Springer. 1992, pp. 31–53.

[58] David Pointcheval and Jacques Stern. 'Security proofs for signature schemes'. In: *Advances in Cryptology—EUROCRYPT'96: International Conference on the Theory and Application of Cryptographic Techniques Saragossa, Spain, May 12–16, 1996 Proceedings 15*. Springer. 1996, pp. 387–398.

[59] *PQC Standardization Process: Announcing Four Candidates to be Standardized, Plus Fourth Round Candidates*. URL: https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4#new-call.

[60] Nael Rahman and Vladimir Shpilrain. 'MOBS: Matrices Over Bit Strings public key exchange'. In: *https://eprint.iacr.org/2021/560* (2021).

[61] Nael Rahman and Vladimir Shpilrain. 'MAKE: A matrix action key exchange'. In: *Journal of Mathematical Cryptology* 16.1 (2022), pp. 64–72.

[62] Oded Regev. 'A subexponential time algorithm for the dihedral hidden subgroup problem with polynomial space'. In: *arXiv preprint quant-ph/0406151* (2004).

[63] Eleanor G Rieffel and Wolfgang H Polak. *Quantum computing: A gentle introduction*. MIT Press, 2011.

[64] Ronald L Rivest, Adi Shamir and Leonard Adleman. 'A method for obtaining digital signatures and public-key cryptosystems'. In: *Communications of the ACM* 21.2 (1978), pp. 120–126.

[65] Vitaliĭ Roman'kov. 'Linear decomposition attack on public key exchange protocols using semidirect products of (semi) groups'. In: *arXiv preprint arXiv:1501.01152* (2015).

[66] Alexander Rostovtsev and Anton Stolbunov. 'Public-key cryptosystem based on isogenies'. In: *Cryptology ePrint Archive* (2006). URL: https://eprint.iacr.org/2006/145.

[67] Dylan Rudy and Chris Monico. 'Remarks on a tropical key exchange system'. In: *Journal of Mathematical Cryptology* 15.1 (2021), pp. 280–283.

[68] Claus-Peter Schnorr. 'Efficient identification and signatures for smart cards'. In: *Advances in Cryptology—CRYPTO'89 Proceedings 9*. Springer. 1990, pp. 239–252.

[69] Peter W Shor. 'Algorithms for quantum computation: discrete logarithms and factoring'. In: *Proceedings 35th annual symposium on foundations of computer science*. Ieee. 1994, pp. 124–134.

[70] Anton Stolbunov. 'Constructing public-key cryptographic schemes based on class group action on a set of isogenous elliptic curves'. In: *Advances in Mathematics of Communications* 4.2 (2010), pp. 215–235.

[71] Anton Stolbunov. 'Cryptographic Schemes Based on Isogenies'. PhD thesis. Jan. 2012. DOI: 10.13140/RG.2.2.20826.44488.