The Semigroup Action Problem in Cryptography

Oliver Wilhelm Gnilke

UCD Student Number: 10278869

The thesis is submitted to University College Dublin in fulfillment of the requirements for the degree of Doctor of Philosophy

School of Mathematical Sciences
Head of School: Prof. Gary McGuire

Principal Supervisor:
Dr. habil. Marcus Greferath

Doctoral Studies Panel:
Dr. Eimear Byrne
Prof. Gary McGuire

December 2014

Contents

1 Preliminaries **4**
- 1.1 Semigroups, Semirings . . . 4
- 1.2 Semigroup Actions . . . 9

2 Public-Key Cryptography **12**
- 2.1 Key Exchange Protocols . . . 12
- 2.2 Key-Exchange Protocols on Non-Commutative Structures . . . 21
- 2.3 Other Applications of Public-Key Cryptography . . . 23
  - 2.3.1 Public-Key Encryption . . . 23
  - 2.3.2 Digital Signatures . . . 24
- 2.4 Generic Attacks . . . 25
  - 2.4.1 Shanks's Algorithm . . . 26
  - 2.4.2 Pollard's Rho . . . 27
- 2.5 Pohlig-Hellman . . . 30
- 2.6 Quantum Algorithms . . . 33

3 Key-Exchange based on Semigroups **37**
- 3.1 Semigroup Discrete Logarithm . . . 37
- 3.2 Key-Exchange Protocols based on Semigroup Actions . . . 39
- 3.3 Comparison of Diffie-Hellman and Anshel-Anshel-Goldfeld . . . 41
- 3.4 Monico-Maze-Rosenthal Protocol . . . 43
  - 3.4.1 Statistical Analysis of the Monico-Maze-Rosenthal protocol . . . 46
  - 3.4.2 Steinwandt Suárez-Corona Attack . . . 50

4 Attacks on Semigroup Actions **52**
- 4.1 Brute-Force . . . 52
- 4.2 Time-Memory trade-offs for Semigroup Action Problems . . . 53
- 4.3 Pohlig-Hellman Reductions . . . 57

5 Design of a Key-Exchange Protocol **67**
- 5.1 A Non-Commutative Semigroup Action Key-Exchange Protocol . . . 67
- 5.2 Semigroup Semirings . . . 69
- 5.3 Reductions of Semigroup Semirings . . . 70

Für meinen Vater

Abstract

In this dissertation we address the use of semigroup actions for cryptographic purposes. The necessary algebraic background on semigroup and semiring theory is provided and several different key-exchange protocols based on semigroup actions presented, most prominently a construction by Monico, Maze and Rosenthal based on the Diffie-Hellman key exchange utilizing commutative semigroups. We introduce our own key-exchange protocol based on an idea by Anshel, Anshel and Goldfeld that overcomes several shortcomings of earlier constructions and no longer requires the semigroups to be commutative.
Several different time-memory trade-offs are presented and it is shown that semigroup action problems can be secure when considering these types of attack only.
We then analyze the security of general semigroup action problems and develop a framework for treating them similar to the Pohlig-Hellman reductions for the discrete logarithm problem. It is shown that most semigroups are susceptible to these attacks and we give several examples of how these reductions can be applied to specific semigroups. Furthermore for any proper semigroup that is secure against these attacks it is implausible that it has applications in cryptography. We conclude by proving that even our own construction is not secure when considering these reductions.

Declaration of Authorship

I hereby certify that the submitted work is my own work, was completed while registered as a candidate for the degree stated on the Title Page, and I have not obtained a degree elsewhere on the basis of the research presented in this submitted work

_____

Acknowledgements

I want to thank my supervisor Dr. habil. Marcus Greferath for his confidence in me, his support, and his friendship. Dr. Jens Zumbrägel has been an invaluable source of knowledge and answered many of my questions with patience and diligence.
I am thankful for my mom's constant support during my studies, all of this would not have been possible without her.
To Ais, for countless lifts, all the encouraging conversations, and cakes, I am forever indebted. My friends in Hamburg and elsewhere who have been there for me whenever I needed them, thank you for making sure I never felt like I was too far away or too long gone. Especially Markus and Mickey, who have provided all forms of support and nourishment and Bober, whose friendship and encouragement mean a lot to me.
I am particularly grateful for the chance to visit Assistant Prof. Dr. Akiko Manada, and Prof. Hiro Morita at the University of Electro-Communications in Tokyo, they have introduced me to many interesting applications and have been most hospitable.
The Greferath/Rößing family has been most welcoming and host to many joyful evenings that I am glad to have been part of. I will always remember the time Sara spent with our group in Dublin with great fondness.
My gratitude belongs to my examination committee, Dr. Elisa Gorla, Dr. Eimear Byrne, and Prof. Gary McGuire, for many insightful comments on this thesis. Prof. Richard Stanley has been kind enough to answer my question on mathexchange, turning Open Problem 77 into Lemma 77.
Andreas has been a great desk neighbor and group member and I am appreciative of his help and encouragement during the early stages of my PhD. A sincere thanks goes to the friendly staff at KC Peaches, foremost Katie, who have provided me with lunches that have made me the envy of many at UCD and Paulina of Ariosa Coffee, who never failed to provide a friendly smile and chat with a great latte.
Many thanks to @mrandmrsstevens for their @akillersandwich which have made so many Friday afternoons so much better, best of luck with your endeavors.

Oliver W. Gnilke December 2014

Introduction

Contemporary public-key cryptography relies mainly on two different computational problems, the factorization of integers and the discrete logarithm in groups. Both have withstood years of attempts of solving them efficiently, but nonetheless it seems advisable to consider alternatives.

When choosing an instance of a cryptosystem, one has to balance between key sizes and the extent of security it provides. Especially the discrete logarithm on algebraic varieties, mainly elliptic curves, has proven to provide the best ratio between these two conflicting aims. However in all of these settings generic attacks of computational effort in the order of the square root of the size of the problem chosen exist. This means that the security level, measured in bits, is at most half as big.

To remedy this fact, semigroup actions were suggested as a generalization of the exponentiation in groups. Rosenthal and his team [18] suggested a first protocol for a key-exchange. Instead of working in $(\mathbb{Z}_n, \cdot)$ they use the multiplicative semigroup of matrices over a semiring. The square root attacks mentioned earlier rely on the linearity of the exponentiation, i.e. $g^a \cdot g^b = g^{a+b}$, and on $\mathbb{Z}_n$ having many units. In fact it is strongly suggested to use groups of prime order for the discrete logarithm, in which case one works in the field $(\mathbb{Z}_p, \cdot)$. Seeing as neither the linearity condition nor the invertibility is given in general in the case of a semigroup action we see that the square root attacks might be prevented.

This thesis aims to provide an extensive analysis of the security of semigroup action problems against generic attacks. We start in chapter one by providing the reader with the algebraic background on semigroups, semirings and semigroup actions needed to follow the examples in the later chapters. Chapter two is an overview of public-key cryptography with a focus on key-exchange protocols; we present the Diffie-Hellman protocol as well as the Anshel-Anshel-Goldfeld protocol, and briefly explain other applications of public-key cryptography. We then give a detailed account of the generic square root attacks on the discrete logarithm problem before we consider Shor's quantum algorithm that solves the so-called Hidden Subgroup Problem.

We start chapter three by presenting a rather simple proposal of using semigroups for

a discrete logarithm problem and prove that this construction is most secure when it is equivalent to the classical discrete logarithm. We then proceed to present the proposed generalization of the Diffie-Hellman protocol by Rosenthal and his team using commuting semigroups. With this generalization it is easy to describe the Ko-Lee protocol as a Diffie-Hellman protocol for a particular group action and we recall a comparison of it to the Anshel-Anshel-Goldfeld protocol for a very similar group action by Shpilrain and Ushakov. We conclude chapter three with an account of the Monico-Maze-Rosenthal protocol and a statistical analysis as well as a description of the attack by Steinwandt and Suárez Corona.

Chapter four is exclusively concerned with attacks on semigroup action problems, we start by re-evaluating the brute-force approach in this new setting and go on providing a thorough analysis of possible time-memory trade-offs. We show that semigroup actions can indeed be very resistant against these attacks if no additional structure is assumed and only few units are present. The last section of chapter three covers a new line of attack that we suggest for treating semigroup action problems. We present reductions that transform the given problem to a new problem in smaller semigroups and work for general semigroup actions. Our reductions are similar to the ideas of Pohlig and Hellman and they actually depend on the semigroup having non-units. We show for several examples how these reductions can be applied in practice.

Chapter five presents our own take on designing a key-exchange protocol. The attack by Steinwandt and Suárez Corona revealed severe problems with the construction by Monico, Maze and Rosenthal. We overcame these by using a non-commutative key-exchange based on the protocol by Anshel, Anshel and Goldfeld. We introduce an extensive class of semirings that could be used for our protocol. However, considering our reductions from chapter four shows that our protocol will always be susceptible to these means of attacks.

"All the king's horses and all the king's men,
Couldn't put Humpty together again."
Nursery Rhyme

1 Preliminaries

1.1 Semigroups, Semirings

We begin with the definition of one of the most basic structures in modern algebra.

Definition 1
A **semigroup** is a pair $(S, \circ)$, where S is a set and $\circ$ an associative binary operation on S. If the set S contains an element 1_S such that $1_S \circ s = s \circ 1_S = s$ for all $s \in S$ we call $(S, \circ, 1_S)$ a **monoid** and refer to the element 1_S as the one or the identity. A semigroup that contains an absorbing element 0_S, i.e. $0_S \circ s = s \circ 0_S = 0_S$ for all $s \in S$, is called a **nulloid** and 0_S is referred to as the zero element.
For a monoid we define the subset $S^* := \{s \in S : \exists s' \in S, \ \ s \circ s' = s' \circ s = 1\}$ to be the subgroup of units in S. If $S = S^*$ then $(S, \circ, 1_S)$ is a **group**. If S is not a group we call it a **proper** semigroup.
A **morphism of semigroups** is a map $\varphi : (S, \circ_S) \to (T, \circ_T)$ such that

$$\varphi(s \circ_S s') = \varphi(s) \circ_T \varphi(s')$$

for all $s, s' \in S$. If S and T are monoids with ones 1_S and 1_T then a **morphism of monoids** is a semigroup morphism ϕ such that $\varphi(1_S) = 1_T$.

Example 2

i) $(\mathbb{Z}, +, 0)$ is a group.

ii) $(\mathbb{Z}, \cdot, 1, 0)$ is a monoid and a nulloid.

iii) $(\mathbb{N}, \max, 1)$ is a monoid, but has no absorbing element.

iv) $(\mathbb{Z}_{\leq m}, \max, m)$ is a nulloid, but it has no neutral element.

v) $(\mathrm{Mat}_{n\times n}(\mathbb{R}), \cdot, E_{n\times n}, 0_{n\times n})$ is a monoid and a nulloid.

It is common to omit the operation (especially in multiplicatively written semigroups) and hence we will sometimes write st for $s \circ t$. In finite monoids left invertibility implies right

invertibility and vice-versa and we therefore only refer to invertible and non-invertible elements.

Definition 3
Let $(S, \circ)$ be a semigroup. A subset $T \subseteq S$ is called a **subsemigroup** if $T \circ T \subseteq T$, i.e. if T is closed under the operation of S. A subsemigroup $I \subseteq S$ is called a **right ideal** if $I \circ S \subseteq I$, which is reminiscent of the case of ideals of rings.

There is rich theory on semigroups, for more information on this topic the reader is referred to [9].
An important extension of Definition 3 that will be of use later on is given by an additional structure on the set.

Definition 4 (Partial/Total Order)
Given a set X a **partial order** $\leq$ on X is a binary relation on X such that the following hold for all $x, y, z \in X$:

i) **Reflexivity:** $x \leq x$

ii) **Antisymmetry:** $x \leq y$ and $y \leq x \Rightarrow x = y$

iii) **Transitivity:** $x \leq y$ and $y \leq z \Rightarrow x \leq z$.

As usual for relations the infix notation $x \leq y$ is used for $(x, y) \in \leq$. If additionally for all $x, y \in X$ it holds that $x \leq y$ or $y \leq x$ then $\leq$ is called a **total order**, sometimes also referred to as linear order.

A convenient graphic representation of finite partial orders are **Hasse diagrams**. A partial order $\leq$ on a set S can be used to describe a loop-free directed graph (S, E) with vertex set S and edge set $E := \{(s, s') : s < s' \text{ and } \nexists t \in S,\ s < t < s'\}$. The Hasse diagram is an illustration of this graph where for every edge $(s, s') \in E$ the direction of edge is represented by s' being higher on the vertical axis than s. Figure 1.1 shows two examples of Hasse diagrams. On the left, the partial order is given by divisibility $a \leq b :\Leftrightarrow a|b$ and on the right it is set inclusion $S \leq T :\Leftrightarrow S \subseteq T$.

Definition 5 (Partially Ordered Semigroup)
Let $(S, \circ)$ be a semigroup with a partial order $\leq$ on its elements. If $\leq$ is compatible with the operation $\circ$, meaning

$$x \leq y \Rightarrow z \circ x \leq z \circ y \quad \text{and} \quad x \leq y \Rightarrow x \circ z \leq y \circ z$$

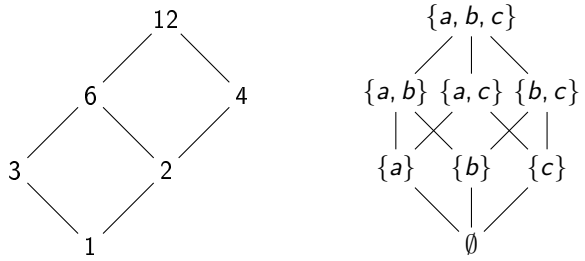

Figure 1.1: Hasse diagrams

for all $x, y, z \in S$ then $(S, \circ, \leq)$ is called a partially ordered semigroup, or a **posemigroup**.

Example 6

i) $(\mathbb{Z}, +, \leq)$ is a totally ordered group.

ii) $(\mathbb{N}, \max, \leq)$ is a posemigroup and $\leq$ is a total order.

We will see some more examples later on.

As in the case of groups we can extend the definition of a semigroup to include a second, compatible operation.

Definition 7 (Semiring)

A **semiring** is a triple $(S, +, \cdot)$ where S is a set and $+$ and $\cdot$ are binary operations, such that $+$ is commutative, both $(S, +)$ and $(S, \cdot)$ are semigroups and the following distributive laws hold for all $x, y, z \in S$

$$\begin{aligned} & x \cdot (y + z) = (x \cdot y) + (x \cdot z) \\ \text{and} \quad & (x + y) \cdot z = (x \cdot z) + (y \cdot z). \end{aligned}$$

If $(S, \cdot)$ is a monoid then $(S, +, \cdot)$ is a semiring with one. If $(S, \cdot)$ is a nulloid with zero 0_S and it holds that 0_S is neutral with respect to $+$, then $(S, +, \cdot)$ is a semiring with zero. We will only consider semirings with zero from now on.

The morphisms of semirings are the maps that are compatible with the operations.

Definition 8 (Morphism of Semirings)

Let $(S, +_S, \cdot_S)$ and $(R, +_R, \cdot_R)$ be semirings. A map $f : S \longrightarrow R$ is a semiring morphism iff it is a semigroup (or monoid) morphism between the additive semigroups $(S, +_S)$ and

$(R, +_R)$ and the multiplicative semigroups $(S, \cdot_S)$ and $(R, \cdot_R)$. A bijective morphism is called an isomorphism.

Semirings, in contrast to other algebraic structures, allow for several distinct notions of simplicity. Here we will define the absence of non-trivial congruence relations as (congruence-) simple.

Definition 9
An equivalence relation $\sim$ on a semiring $(R, +, \cdot)$ is called a congruence if it is compatible with both operations,

$$x \sim y \Rightarrow s + x \sim s + y$$
$$x \sim y \Rightarrow s \cdot x \sim s \cdot y$$
$$x \sim y \Rightarrow x \cdot s \sim y \cdot s.$$

for all $s, x, y \in S$. A semiring S is called **congruence simple** (or just simple), if the only congruences are the identity $=$ and the all congruence $R \times R$.

There is a one-to-one relation between congruences on a semiring S and kernels of semiring morphisms $f : S \longrightarrow R$. A classification of finite (congruence-) simple semirings with zero can be found in [34] and we will recall the theorem therein.

Theorem 10 *Let $(R, +, \cdot)$ be a finite semiring with zero which is not a ring and $|R| > 2$. Then the following are equivalent:*

i) *$(R, +, \cdot)$ is congruence simple*

ii) *$(R, +, \cdot)$ is isomorphic to a dense subsemiring of $\mathrm{End}(M)$ where $(M, +)$ is a finite commutative monoid such that $m + m = m$ for all $m \in M$.*

A subsemiring $(S, +, \circ)$ of endomorphisms of an idempotent commutative monoid $(M, +)$ is called dense if it contains the elements

$$e_{a,b}(x) := \begin{cases} 0 & \text{if } x + a = a \\ b & \text{otherwise} \end{cases}$$

for all $a, b \in M$.

Example 11
We want to revisit some of the examples given in [35]

i) The semiring $R_{2,a} := (\{0,1\}, \max, 0)$ is congruence simple.

ii) The semiring $R_{2,b} := (\{0,1\}, \max, \min)$ is called the **boolean semifield**, since its set of non-zero elements forms (trivially) a group with respect to multiplication.

iii) The ring $R_6 := (\{0,1,2,3,4,5\}, +, \cdot)$ defined by the following operation tables

$+$	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	1	1	1	1	5
2	2	1	2	1	2	5
3	3	1	1	3	3	5
4	4	1	2	3	4	5
5	5	5	5	5	5	5

$\cdot$	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	2	0	0	5
3	0	3	4	3	4	3
4	0	4	4	0	0	3
5	0	5	2	5	2	5

is up to isomorphism the only congruence-simple semiring with 6 elements.

iv) For any semiring R the matrix ring $\mathrm{Mat}_{n\times n}(R)$ is a semiring. If R is a semiring with one then $\mathrm{Mat}_{n\times n}(R)$ is congruence-simple iff R is congruence-simple, see [17].

The two semirings $R_{2,a}$ and $R_{2,b}$ are up to isomorphism the only congruence-simple semirings of order 2 and therefore complete the classification in Theorem 10.

We recall an interesting corollary of the classification of congruence-simple semirings (see Proposition 3.1 and Remark 2.1 in [34]).

Corollary 12 *Every congruence-simple semiring S has idempotent addition and the addition induces a partial order on S via*

$$x \leq y :\Leftrightarrow x + y = y.$$

If S is finite it follows that S is a lattice with join operation $x \vee y := x + y$ and meet operation $x \wedge y := \sum_{z \leq x, z \leq y} z$.

In other words, if $(S, +, \cdot)$ is a simple semiring then $(S, +)$ is a commutative posemigroup with respect to above partial order. For the semiring R_6 in Example 11 the Hasse diagram of the partial order is shown in Figure 1.2.

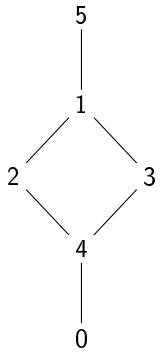

Figure 1.2: Hasse diagram for R_6

1.2 Semigroup Actions

Definition 13 (Semigroup Action)
Let S be a semigroup and X be a set. We say a mapping $\varphi : S \times X \to X,\ (s,x) \mapsto s\,.\,x$ is a semigroup action of S on X, or simply X is an S-set if it has the following flow property $\varphi\,(s, \varphi(t,x)) = \varphi(st,x)$ or equivalently $s\,.\,(t\,.\,x) = (st)\,.\,x$.
Equivalently a semigroup action is given by a morphism $\Phi : S \to X^X,\ s \mapsto \Phi_s : x \mapsto s\,.\,x = \varphi(s,\cdot)$, where X^X is the semigroup of all mappings of X into itself. We will also refer to the map $\Psi_x := \varphi(\cdot, x)$ as being **induced** by the element x.

Example 14
Given a group $(G,\cdot)$ of order n we can define a semigroup action of $(\mathbb{Z}_n,\cdot)$ on G by

$$\varphi : \mathbb{Z}_n \times G \to G$$
$$(z,g) \mapsto \begin{cases} g^z := \underbrace{g \cdot \cdots \cdot g}_{z \text{ times}} & \text{for } z \neq 0 \\ e_G & \text{for } z = 0 \end{cases}.$$

Using Lagrange's theorem one can prove that this is indeed a semigroup action.

Definition 15 (Orbit, Image)
For X an S-set. For $x \in X$ and $s \in S$ we define the following two sets:

a) $S.x := \mathrm{im}(\Psi_x) = \{s\,.\,x : s \in S\}$, the **orbit** of x

b) $s.X := \mathrm{im}(\Phi_s) = \{s\,.\,x : x \in X\}$, the **image** of s.

The second definition is unique to semigroup actions since in the case of S being a group it follows trivially that the image of every element is X since $s \,.\, (s^{-1} \,.\, x) = x$ for all $x \in X$.

The following Lemma shows another property that makes semigroup actions different from group actions and has to be addressed when considering cryptographic uses of semigroup actions.

Lemma 16 *For every action of a proper finite monoid S on a set X, there exists $x \in X$ such that the induced map $\Psi_x : S \to X$ is non-injective.*

PROOF Since S is a proper finite monoid not every element is invertible. Let $s \in S$ be non-invertible then there exist $u, v \in S$ with $u \neq v$ such that $us = vs$. It then follows that for any $x \in X$

$$\Psi_{s \,.\, x}(u) = u \,.\, (s \,.\, x) = v \,.\, (s \,.\, x) = \Psi_{s \,.\, x}(v).$$

Hence all $y \in s \,.\, X$ give rise to non-injective maps Ψ_y. ∎

Example 17
A semigroup action we will consider later was first presented in [18]. Consider a semiring $(R, +, \cdot)$ and let $(\mathcal{M}, +, \cdot) := \mathrm{Mat}_{n \times n}(R)$ the ring of $n \times n$ matrices over R. Define the opposite ring $\mathcal{M}^{op} := (\mathcal{M}, +, \cdot^{op})$ where $A \cdot^{op} B := B \cdot A$ for all $A, B \in \mathcal{M}$. Then by associativity of matrix multiplication it follows that

$$\begin{aligned} \varphi : (\mathcal{M} \times \mathcal{M}^{op}) \times \mathcal{M} &\longrightarrow \mathcal{M} \\ ((A, B), X) &\longmapsto AXB \end{aligned}$$

is a semigroup action.

A useful notion is presented in the following definition.

Definition 18
Given a semigroup action of S on X, every element $x \in X$ naturally induces a left congruence $\sim_x := \sim_{\Psi_x}$ on the semigroup by

$$s \sim_x t \;:\Leftrightarrow\; s \,.\, x = t \,.\, x.$$

Using this notation the size of the orbit of an element $x \in X$ is given as the number of congruence classes

$$|S \,.\, x| = \left| S \big/ \sim_x \right|.$$

Similar to free group actions we use the following convention

Definition 19
Let X be an S-set. We call $x \in X$ free (with respect to the semigroup action) iff the induced congruence is trivial, i.e. $\sim_x = \mathrm{id}$ or equivalently

$$s \,.\, x = t \,.\, x \implies s = t.$$

This is equivalent to saying that the map Ψ_x is injective and the orbit of x is of maximal size $|S \,.\, x| = |S|$.

Example 20
Consider the action of $\mathbb{Z}_n$ on a group G of order n by exponentiation as in Example 14. The free elements are the generators of the group, hence iff G is cyclic it has $\phi(n)$ different free elements.

2 Public-Key Cryptography

2.1 Key Exchange Protocols

A fundamental principle of cryptography was formulated by Auguste Kerckhoffs in 1883 and is now known as Kerckhoffs's principle [24]. It is more commonly known in a reformulation by Claude Shannon

"The enemy knows the system"

which is also known as Shannon's maxim. For security analysis of cryptosystems this implies that the only unknown to an attacker is the key used. Systems designed with this attacker model in mind tend to be drastically more secure, but rely on changing the key on a regular basis. Organizations that wanted to employ a cryptographic system were suddenly faced with a different problem of seemingly administrative nature, the **Key Distribution Problem**. For almost a century the Achilles heel of any secured communication was the need to equip both parties with the same secret key beforehand. In practice this meant that military units, like submarines, had to be given the keys that would be used months in advance. The keys were printed in key tables (sometimes also referred to as codebooks) and a lot of effort was put into protecting them. Furthermore in a system with n users there would need to be $\frac{n(n-1)}{2}$ different keys to allow every user to securely communicate with every other user. For more information on the history of key tables and Kerckhoffs's contributions to cryptography see [10].

The first solution to the key distribution problem is often credited to Whitfield Diffie and Martin Hellman (see Protocol 30) who presented their protocol in 1976. It should be mentioned though that Ralph Merkle had developed another, less efficient, protocol already in 1974. In 2002 it was disclosed that employees of the GCHQ had as early as 1973 come up with the Diffie Hellman protocol and the RSA encryption algorithm. In this work we will not present the Merkle Puzzles or the RSA encryption scheme but solely focus on the Diffie Hellman protocol and related mathematical problems.

It should be noted though that the solutions to the key distribution problem are not a panacea but introduce other problems. Most significant is the lack of authentication. When a secret key is used, every party is certain about who is at the other end of the line

assuming the key had not been compromised. The key exchange protocols we want to discuss do not provide any authentication and are therefore susceptible to man-in-the-middle attacks. This detriment can be remedied by using a public-key infrastructure with digital signatures and trusted third parties like certificate authorities. We will restrict our attention to the key-exchange protocols and ignore authentication issues for now.

We define a two-party key-exchange protocol as a sequence of calculations and transmissions between two parties, most commonly referred to as Alice and Bob.

Definition 21 (Key-Exchange Protocol (KEP))

0.) Setup:
An initial handshake is performed, and protocol parameters specified.

1.) Generation of public/private keys:
Both parties generate ephemeral key pairs (k_s^A, k_p^A) and (k_s^B, k_p^B) respectively.

2.) Exchange of public keys:
The parties exchange their public keys k_p^A and k_p^B.

3.) Calculating the shared key:
Alice uses the received public key k_p^B and her own key pair to calculate a shared key (shared secret) K_A. Bob uses k_p^A and his own key pair to calculate K_B.

Correctness of a protocol is given if $K_A = K_B$ for all possible key pairs.

More complex definitions would be possible to allow for several communication and computation steps between the parties involved or more than two parties to participate. But we will restrict our attention to the simple case described in the definition above. For simplicity we sometimes assume that the generated shared key $K_A = K_B$ is a bit sequence in $\{0,1\}^*$.

There are two different attacker models for our framework, but will only consider the weaker type for the rest of this dissertation.

Definition 22 (Attacker Models)

i) Passive Attacker / Eavesdropper:
A passive attacker gathers all the information that is *sent* between the parties involved in the protocol and tries to infer information about the shared secret. This attacker has no means of interfering with the transmissions and can not alter or disrupt them.

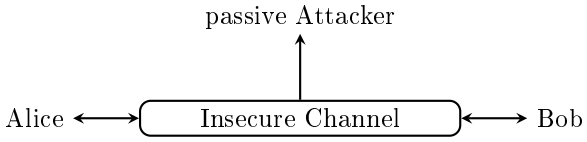

Figure 2.1: passive attacker model

ii) Active Attacker / Man-in-the-Middle:
An active attacker not only sees all the transmissions between the parties but also has the (sometimes limited) ability to alter or disrupt the information in transit or inject his own information into the channel.

Figure 2.2: active attacker model

It is immediately clear that in the second case an attacker could even prevent the parties from communicating altogether. Usually one assumes that the attacker instead will try to hide his presence and convince the communicating parties that they have established a secure shared secret. This will enable the attacker to listen in on the conversation.
Protocols that consider this case are mostly concerned with the detection of modifications to the transmissions and therefore the detection of an active attacker. This is usually achieved by adding some form of authentication to the messages sent.

The security of a KEP against eavesdroppers is formally defined using the following experiment (see [12] for more details).

Definition 23 (KEP indistinguishability experiment)
Let $\mathcal{A}$ be a probabilistic polynomial-time (PPT) algorithm that takes as input the transcript generated by one run of a KEP and an element from the space of possible shared secrets. We then define the following experiment

1. Two parties perform a KEP as described above and generate a shared secret K. The transcript of transmissions (step 2.) along with the protocol specifics and the security parameter n (step 0.) are recorded.

2. With equal probabilities the adversary $\mathcal{A}$ is provided with the transcript and the shared secret K or with the transcript and a random element from the space of

possible shared secrets.

3. We say $\mathcal{A}$ succeeded if it can correctly decide whether the element it received with the transcript is the shared secret or not.

We see that an algorithm $\mathcal{A}$ that randomly guesses will succeed with probability $\frac{1}{2}$ but can not be considered a threat to the KEP. In fact an algorithm that succeeds with just little higher probability than $\frac{1}{2}$ should still not be considered a break of the KEP. The next definition gives the term *'just little'* a formal framework.

Definition 24 (Negligible Function)
A function $\text{negl} : \mathbb{N} \longrightarrow \mathbb{R}_0^+$ is called **negligible** if for every positive polynomial $p \in \mathbb{N}[x]$ there exists $N \in \mathbb{N}$ such that for all integers $n > N$ it holds that

$$\text{negl}(n) < \frac{1}{p(n)}$$

An examples of a negligible functions is 2^{-n}, which corresponds to the probability of randomly guessing the right preimage of a injective function $f : \{0,1\}^n \to \{0,1\}^*$.
This now allows for a formal definition of security of a KEP against eavesdroppers.

Definition 25 (Security of a KEP)
A KEP is considered secure in the presence of a passive adversary if for every probabilistic polynomial-time algorithm $\mathcal{A}$ there exists a negligible function $\text{negl}(n)$ such that

$$\Pr[\mathcal{A} \text{ succeeds}] \leq \frac{1}{2} + \text{negl}(n)$$

where the probability is taken uniformly over all possible choices in the execution of the protocol, and n is the security parameter.

This definition is a very strong requirement on the KEP since an attacker is only asked to decide between two possible shared secrets. In an application an adversary actually would need to recover the shared secret K by himself from the transcript of the communications to compromise the security. In general this can be achieved in two different ways.

1. An adversary might be able to deduce the shared key K directly from the public keys that he overheard without ever getting hold of on of the secret keys.

2. The adversary could recover one of the secret keys k_s^* and then calculate K the same way the original holder of k_s^* does.

The relation between these two problems depends on the protocol used and is an active topic of research.

In most KEPs, key pairs are generated by uniformly at random choosing a secret key k_s and then using a function f to calculate the corresponding public key $k_p := f(k_s)$ as a function[1] of k_s. Recovering the secret key from the public key is then equivalent to finding preimages for f. This problem motivates one of the most fundamental concepts in modern cryptography (we loosely follow the definitions given in [12]).

Definition 26 (One-Way function)
A function $f : \{0,1\}^* \to \{0,1\}^*$ is a one-way function if the following two conditions hold:

1.) There exists a polynomial-time algorithm to compute f on arbitrary values x.

2.) For all probabilistic polynomial-time algorithms $\mathcal{A}$ the probability of inverting f is negligible in the length n of the input $x \in \{0,1\}^n$

$$\Pr\left[f\left(\mathcal{A}(f(x))\right) = f(x)\right] \leq \text{negl}(n).$$

It would be desirable for the function that maps the private keys to the public keys to be one-way. But proving that any particular function is one-way appears difficult so far. In fact the existence of a one-way function would imply that $P \neq NP$ [33, Theorem 6.6], therefore solving one of the Millennium Problems.
In practice one works with functions that are assumed to be one-way and have so far resisted attempts to find a polynomial algorithm that retrieves preimages. We will in the following discuss a well known candidate for a one-way function that is widely used in practice.

At the foundation of many modern cryptographic protocols stands an implementation of the following problem

Definition 27 (Discrete Logarithm Problem)
For any cyclic group G of order n and generator b of G there is a natural isomorphism

[1] In contrast, identity based encryption chooses the public key and then calculates the corresponding secret key.

from $(\mathbb{Z}_n, +)$ into G,

$$\begin{aligned} \varepsilon_b : \mathbb{Z}_n &\longrightarrow G \\ z &\longmapsto b^z. \end{aligned}$$

Its inverse $\log_b : G \to \mathbb{Z}_n$ is called the discrete logarithm with respect to basis b. The **discrete logarithm problem** is the explicit calculation of values of $\log_b(g)$ for $g \in G$.

Remark 28
To be precise, we would have to consider a class of discrete logarithm problems in groups of sizes n. This class would be a candidate for a one-way function since we expect the effort of an attacker to grow faster than polynomial in the size of the group. For the sake of legibility and simplicity we will restrict our considerations to one group instead.

A purely mathematical view on this problem does not reveal any serious problem: the solution always exists and is uniquely defined. But for an evaluation of its cryptographic hardness, computational complexity questions have to be considered. We will therefore discuss several algorithms and compare them in terms of memory and computational requirements. A few examples show that the representation of the group G is of paramount importance.

Example 29

i) For $G = \mathbb{Z}_n$ consider the isomorphism $\varepsilon_b(z) := z \cdot b$ where $\gcd(b, n) = 1$, hence b is a generator of $(\mathbb{Z}_n, +)$. It is obvious that $\log_b(x) = x \cdot b^{-1}$ where b^{-1} is the inverse of b in $\mathbb{Z}_n^*$ and is easily calculated by the Euclidean algorithm.

ii) Let $G = \mathbb{F}_q^*$ and $b \in G$ an element of order n. For $\varepsilon_b(z) := b^z$ the associated DLP can be computationally hard and some instances of this problem are used in modern cryptographic applications, see [14].

iii) Let $E(\mathbb{F}_q)$ be the group of points on an elliptic curve, $G = \langle b \rangle$ a subgroup generated by an element b. Then the associated DLP to $\varepsilon_b(z) := z \cdot b$ is believed to be computationally hard for suitably chosen curves [2].

Using this candidate for a one-way function we are ready to describe the Diffie-Hellman key exchange protocol. It has several advantages over the first KEP to achieve secure

[2] A good resource on secure curves and related implementation issues is [3]

transmission of a key over an insecure channel, the so called Merkle Puzzles. While in the construction of Merkle [19] a multitude of possible keys have to be transmitted and the effort of the attacker lies in the order of the square of the effort of the legitimate parties the Diffie-Hellman protocol has far lower transmission cost and provides exponentially higher security[3].

Protocol 30 (Diffie-Hellman key exchange protocol (DHKEP))

0.) Setup:
The protocol parameters are negotiated. These include the group G of order n with generator g.

1.) Generation of public/private keys:
Both parties, Alice and Bob, choose secret elements $a, b \in \mathbb{Z}_n$ respectively as their secret keys and calculate their public keys as

$$p_A = g^a$$
$$p_B = g^b.$$

2.) Exchange of public keys:
Alice and Bob exchange their public keys p_A and p_B.

3.) Calculating the shared key:
Alice, upon receiving p_B from Bob, calculates

$$K_A := p_B^a$$

Bob similarly computes

$$K_B := p_A^b.$$

Correctness follows from commutativity in $\mathbb{Z}_n$, since $K_A = (g^b)^a = g^{ab} = (g^a)^b = K_B$.

An attacker aims to gather information on the shared secret $K_A = K_B$. Several problems related to this task have been studied and names have been coined in the standard literature. A direct retrieval of the shared secret from the intercepted public keys relates to the following problem.

[3]Here, we assume that the chosen instance of the protocol is secure, i.e. an attacker has at most a square root attack at her disposal.

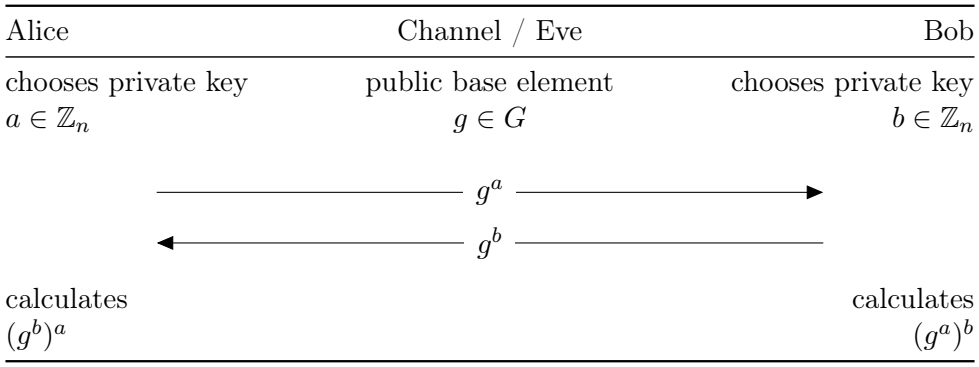

Figure 2.3: Protocol 30

Definition 31 (Computational Diffie-Hellman Problem (CDHP))
Given a triple of elements (g, g^a, g^b) of G compute g^{ab}.

Recovering the secret key of one of the parties is equivalent to the DLP described in Definition 27. As seen in Example 29, the hardness of the DLP depends heavily on the the choice of the group G. A collection of groups that are used in practice can be found in [14] and a good overview of elliptic curve standards is given in [3].
Another problem that relates to Definition 25, where security is defined as a form of indistinguishability, is the following.

Definition 32 (Decision Diffie-Hellman Problem (DDHP))
We call a tuple $(g, g^a, g^b, h) \in G^4$ a valid Diffie-Hellman tuple if $h = g^{ab}$. Given a tuple of group elements (g, g^a, g^b, h) that is with probability $\frac{1}{2}$ a valid Diffie-Hellman tuple and in all other cases h is chosen uniformly random from G, decide whether the tuple is valid.

This leads to the formal definition of DDHP hardness.

Definition 33 (DDHP security)
A discrete logarithm problem is considered DDHP secure if for all probabilistic polynomial-time algorithms $\mathcal{A}$ that attempt to solve the DDHP, there exists a negligible function $\text{negl}(n)$ in the security parameter n such that

$$\Pr[\mathcal{A} \text{ succeeds}] \leq \frac{1}{2} + \text{negl}(n)$$

where the probability is taken over all possible choices of triples for the DDHP as described above.

The hardness relations between these problems can be described using what are called Turing reductions.

Definition 34 (Turing Reduction)
Let X and Y be two computational problems. We say there is a Turing reduction from X to Y or equivalently $X \alpha_T Y$ if there exists a polynomial time algorithm $\mathcal{A}$ that solves X by using a theoretical algorithm $\mathcal{B}$ that solves Y.

The algorithm $\mathcal{B}$ is often termed an **oracle** for Y. Given a reduction $X \alpha_T Y$ it follows that if there exists a polynomial time algorithm that solves the problem Y then the problem X can also be solved in polynomial time. In other words the problem X is at most as hard as the problem Y.

Proposition 35 *There exist Turing reductions* DDHP α_T CDHP α_T DLP.

Proof Given an algorithm $\mathcal{C}$ that solves the CDHP, i.e. $\mathcal{C}(g, g^a, g^b) = g^{ab}$ we can construct an algorithm $\mathcal{D}$ that solves the DDHP. Given an instance (g, g^a, g^b, h) of the DDHP the algorithm $\mathcal{D}$ calls the subroutine $\mathcal{C}$ on (g, g^a, g^b) and receives g^{ab} as a result. If $h = g^{ab}$ then $\mathcal{D}$ is certain that the triple is valid, otherwise it is not valid.
Given an algorithm $\mathcal{L}$ that solves the DLP, i.e. $\mathcal{L}(g, g^x) = \log_g(g^x) = x$ we can design an algorithm $\mathcal{C}$ that solves the CDHP. Given an instance (g, g^a, g^b) of the CDHP the algorithm $\mathcal{C}$ calls $\mathcal{L}$ once on (g, g^a) obtaining a and then calculates $(g^b)^a = g^{ab}$ and hence solves the CDHP. ∎

In [16], the authors take a closer look at the relation between the CDHP and the DLP. They can show that CDHP is equivalent to the DLP given some conditions on the order of the group G and the existence of certain auxiliary groups.

The DDHP is less often used to estimate the security of a given instance of the Diffie-Hellman KEP, in fact there are groups in which the DDHP is solvable while the CDHP and the DLP are still considered hard. In $\mathbb{Z}_p^*$ it is easy to decide whether a given element is a quadratic residue, and the shared secret K is a quadratic residue iff at least one of the public keys p_A, p_B is. Hence an adversary can certainly distinguish K from a random bit string in half the cases. This situation can be remedied by working in a prime order subgroup of $\mathbb{Z}_p^*$ or by working in the subgroup of quadratic residues.
Similarly for elliptic curves, pairings can be used to solve the DDHP by identifying many non-valid tuples.

2.2 Key-Exchange Protocols on Non-Commutative Structures

The Anshel-Anshel-Goldfeld protocol is a two party key-exchange protocol and instead of commutativity as in the case of the Diffie-Hellman protocol it requires linearity of certain functions. We will see how associative algebras naturally lend themselves to be used in such a protocol later on.

In their paper [1] the authors introduce a protocol based on parametrized morphisms. Assume we have two additive (semi-) groups S, T and three maps $\beta : S \times S \to T$ and $\gamma_{1,2} : S \times T \to T$ such that the following equations hold.

a) $\beta(s, t + t') = \beta(s, t) + \beta(s, t')$

b) $\gamma_1(s, \beta(s', s)) = \gamma_2(s', \beta(s, s'))$

Unlike the Diffie-Hellman protocol the Anshel-Anshel-Goldfeld protocol is not entirely symmetric. The two parties need to be cast into two different roles. We assume from now on that the first party, Alice, has role "1" and Bob has role "2".

Protocol 36 (Anshel-Anshel-Goldfeld key-exchange protocol)

0.) Setup:
The two parties agree on the protocol specifics, these include the (semi-)group S, a generating set (s_i) and the maps $\beta, \gamma_{1,2}$.

1.) Generation of public/private keys:
Both parties choose secret keys $s_A = \sum_J s_j$ and $s_B = \sum_K s_k$ as sums of the generating elements. Their public keys are the maps $p_A = \beta(s_A, \cdot)$ and $p_B = \beta(s_B, \cdot)$ represented as the images of the generators, i.e. $p_A = (\beta(s_A, s_i))_I$ and $p_B = (\beta(s_B, s_i))_I$.

2.) Exchange of public keys:
Alice and Bob exchange the public keys p_A and p_B.

3.) Calculating the shared key:
Upon receiving the morphism p_B from Bob, Alice uses her knowledge of the decomposition of s_A into generators to calculate $\sum_J \beta(s_B, s_j) = \beta(s_B, \sum_J s_j) = \beta(s_B, s_A)$. Bob similarly calculates $\sum_K \beta(s_A, s_k) = \beta(s_A, \sum_K s_k) = \beta(s_A, s_B)$. Alice now calculates the shared secret as $k_A = \gamma_1(s_A, \beta(s_B, s_A))$ and Bob uses $k_B = \gamma_2(s_B, \beta(s_A, s_B))$. Property b guarantees correctness of the protocol.

Alice	Channel / Eve	Bob
chooses private key $s_A \in S$		chooses private key $s_B \in S$
	exchanging morphisms	
	$\longrightarrow \beta(s_A, \cdot) \longrightarrow$	
	$\longleftarrow \beta(s_B, \cdot) \longleftarrow$	
calculates $\gamma_1(s_A, \beta(s_B, s_A))$		calculates $\gamma_2(s_B, \beta(s_A, s_B))$

Figure 2.4: Protocol 36

In their paper the authors suggest using inner automorphisms of a multiplicative group as the parametrized morphisms, i.e.

$$\beta(s, x) := s^{-1}xs.$$

The functions $\gamma_{1,2}$ are defined as

$$\gamma_1(a, b) := a^{-1}b$$
$$\gamma_2(a, b) := b^{-1}a.$$

It then follows that

$$\begin{aligned}\gamma_1(a, \beta(b, a)) &= \gamma_1(a, b^{-1}ab) \\ &= a^{-1}(b^{-1}ab) \\ &= (a^{-1}b^{-1}a)b \\ &= (a^{-1}ba)^{-1}b \\ &= \gamma_2(b, a^{-1}ba) = \gamma_2(b, \beta(a, b)),\end{aligned}$$

as desired.
Several groups have been suggested as a base structure for this protocol. Most prominently were braid groups since they possess an easily calculable normal form and the associated conjugation problem was thought to be hard enough. The braid group on n strands, B_n,

can be defined as

$$B_n := \left\langle \sigma_1, \ldots, \sigma_{n-1} \;\middle|\; \begin{matrix} \sigma_i\sigma_j = \sigma_j\sigma_i & \text{if } |i-j| \geq 2 \\ \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j & \text{if } |i-j| = 1 \end{matrix} \right\rangle$$

A good survey on cryptography using braid groups can be found in [4].

2.3 Other Applications of Public-Key Cryptography

Public-Key Cryptography or Asymmetric Cryptography has several other applications of which we will mention the two most important.

2.3.1 Public-Key Encryption

The key-exchange protocols described above establish a shared secret between two parties over a public channel, but this shared secret does not carry any information. Public-key encryption allows parties to securely send messages in one direction.

Given a key-exchange protocol and a symmetric encryption function one can build a public-key encryption scheme. We will present the Elgamal encryption scheme [6] which is based on the Diffie-Hellman KEP. We assume that the message $m = (m_i)_{1 \leq i \leq n}$ has been encoded as a list of group elements $m_i \in G$.

Protocol 37 (Elgamal Encryption)
The public parameters include a cyclic group G of order n together with a generator g.

i.) Alice generates a static key pair by uniformly at random choosing the secret key $a \in \mathbb{Z}_n$ and calculating the public key $g^a \in G$. She publishes her public key g^a.

ii.) For every m_i Bob uniformly at random chooses an element $b_i \in \mathbb{Z}_n$ and calculates the pair $(g^{b_i}, m_i \cdot (g^a)^{b_i})$ using Alice's public key.

iii.) Upon receiving a pair $(g^{b_i}, m_i \cdot g^{ab_i})$, Alice uses her secret key to calculate $(g^{b_i})^a$ and multiplies the second component by its inverse, hence attaining m_i.

For a passive attacker the task of retrieving a message from the transmissions of this encryption scheme is equivalent to the CDHP. The scheme is therefore secure iff the CDHP in the group G is hard. Furthermore if the DDHP is hard in the group the Elgamal encryption achieves semantic security, meaning that the cipher-texts do not leak

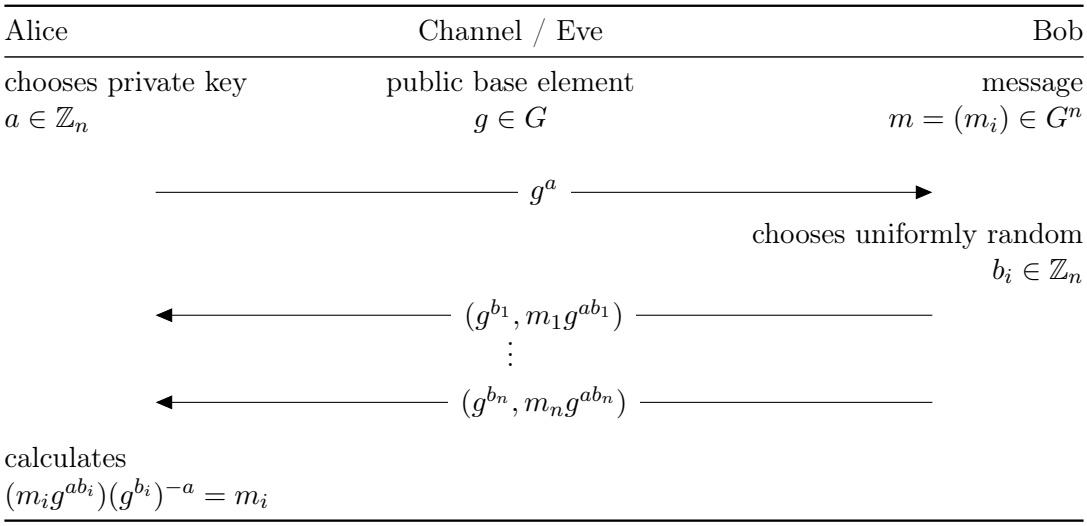

Figure 2.5: Protocol 37

information about the message. An obvious disadvantage of the Elgamal encryption scheme, apart from the computational complexity, is the transmission requirement of twice the message size.

2.3.2 Digital Signatures

Another interesting application of asymmetric cryptography are digital signatures that enable someone who has access to some public key to determine whether a message was signed by the holder of the corresponding secret key. A necessary prerequisite for these schemes are hash functions $H : \{0,1\}^* \to \mathbb{Z}_n$ that map bit strings or arbitrary length to elements in $\mathbb{Z}_n$. For the security of the digital signature it is crucial that H is preimage and collision resistant, for more information on hash functions see [12]. We also assume that group elements have a unique representation as bit strings.

In the following we assume that Alice holds a message $m \in \{0,1\}^*$ that she wants to sign and send to Bob, who then verifies the signature.

Protocol 38 (Schnorr Signature Scheme)
The public parameters include a cyclic group G of order n together with a generator g and a hash function H.

i.) Alice generates a key pair by uniformly at random choosing the secret key $a \in \mathbb{Z}_n$ and calculating the public key $p_k := g^a \in G$. She publishes her public key p_k.

ii.) Alice chooses uniformly at random an element $r \in \mathbb{Z}_n$ and calculates $e := H(m\|g^r)$ and $s := r - ae$, where $m\|g^r$ is the concatenation of the bit strings.

iii.) Alice sends Bob the message m together with the signature (e, s).

iv.) Upon receiving the message and the signature Bob verifies that $e = H(m\|(g^s p_k^e))$.

Alice	Channel / Eve	Bob
chooses private key $a \in \mathbb{Z}_n$	public base element $g \in G$	
	$p_k := g^a$ ⟶	
chooses $r \in \mathbb{Z}_n$ calculates $e := H(m\|g^r)$		
	$(m, e, r - ae)$ ⟶	
		verifies $e = H(m\|(g^{r-ae} p_k^e))$

Figure 2.6: Protocol 38

It can be proven that this signature scheme is secure if the hash function H is secure (assumed to be a random oracle) and if the DLP in the group G is hard.

2.4 Generic Attacks

In this section we would like to present several (generic) algorithms that treat the DLP. We will not discuss specialized algorithms for the DLP in finite fields like index-calculus or number-field sieve attacks since these rely heavily on special properties of finite fields and therefore are unlikely to extend to the SAP.

The DLP on a cyclic group G can be interpreted as a SAP with the semigroup $(\mathbb{Z}_n, \cdot)$ acting on the set G. But it comes with additional properties, some of which are exploited in algorithms described below.
Most prominent is the following 'linearity' of the semigroup action of $\mathbb{Z}_n$ on a cyclic group.

$$g^a \cdot g^b = g^{a+b}$$

As a baseline, we will shortly discuss the expected computational effort a brute-force attack on the DLP would require. Given a DLP $x = \log_g h$ the brute force approach would consist of testing elements $z \in \mathbb{Z}_n$ by calculating g^z till it equals h and $x \in \mathbb{Z}_n$ is found. We assume the choices are made uniformly at random and independently. Then the probability that the brute-force attack will succeed at the t-th step is $\left(1 - \frac{1}{n}\right)^{t-1} \frac{1}{n}$. Therefore the expected number of tests can be calculated as

$$\mathrm{E}[t] = \frac{1}{n} \sum_{t=1}^{\infty} t \left(1 - \frac{1}{n}\right)^{t-1} = \frac{1}{n} \frac{1}{(1 - (1 - \frac{1}{n}))^2} = n.$$

If we assume that the choices are not independent but all distinct then we see that after testing $\frac{n}{2}$ elements we have a probability of $\frac{1}{2}$ that x was among the tested elements. A brute force attempt therefore would require $\mathcal{O}(n)$ operations.

2.4.1 Shanks's Algorithm

The first attack on the DLP was presented by Shanks [28] in 1969. Shanks's algorithm is sometimes referred to as "baby-step giant-step" and yields in its essence a time-memory trade-off. A precomputed list of group elements and indices is used to reduce the computation time.

Algorithm 39
Given a DLP $\log_g h$ in a group G of order n the algorithm proceeds in two stages.

Precalculation:
Let $s := \lfloor \sqrt{n} \rfloor$, a list $L := \left((g^{is}, i) : 0 \leq i \leq \lceil \sqrt{n} \rceil\right)$ of so called *giant-steps* is generated.

Solving the DLP:
The elements $hg^j = g^{x+j}$ are successively calculated for $0 \leq j \leq \lfloor \sqrt{n} \rfloor$ and compared to the first entries in elements in L. When a collision with the list occurs a solution can be calculated since

$$g^{x+j} = g^{si} \;\Rightarrow\; x = si - j.$$

Generating the list L of giant-steps uses $\lceil \sqrt{n} \rceil$ exponentiations. The algorithm is deterministic and terminates successfully after at most $\lfloor \sqrt{n} \rfloor$ baby-steps. Therefore, even when considering sorting the list and lookups, we have a computational effort of $\mathcal{O}(\sqrt{n} \cdot \text{ polylog}(n))$ and a memory requirement of $\mathcal{O}(\sqrt{n})$, where $\text{polylog}(n)$ is a polynomial in $\log(n)$.

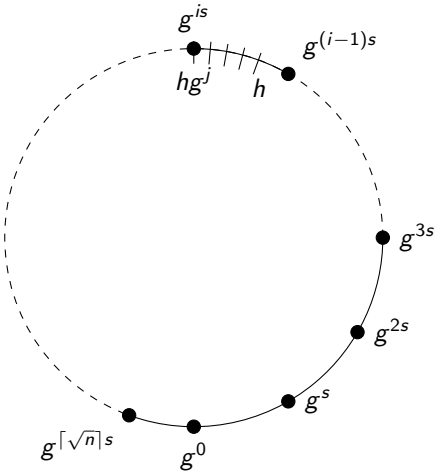

Figure 2.7: Shanks's baby-step giant-step

2.4.2 Pollard's Rho

John Pollard described a probabilistic factoring algorithm in a 1975 paper [26]. In 1978 he published an adaption of his algorithm to the DLP [27]. It uses ideas of Floyd's cycle finding algorithm to minimize storage requirements.
It is optimal in the generic case since it asymptotically achieves the lower bound of $\Omega(\sqrt{p})$.
We will begin by describing the cycle finding problem and Floyd's algorithm for solving it.

Definition 40 (Cycle finding)
Given a function $F : S \to S$ on a finite set S and an element $s \in S$ find minimal $r, p \in \mathbb{N}$ such that $F^p(s) = F^{p+r}(s)$.

We refer to p as the preperiod and r as the period. Algorithms that find the period and the preperiod are called cycle finding algorithms, since they find the length of the cycle contained in the path starting in s in the directed graph induced by the function F on the vertices S.
We present Floyd's algorithm since Pollard's rho is based on similar ideas. The algorithm first searches for collisions of the type

$$F^i(s) = F^{2i}(s)$$

and then determines the preperiod and the period.

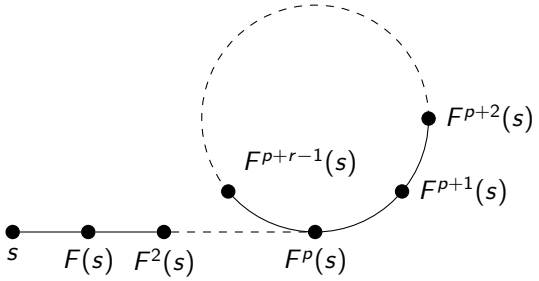

Figure 2.8: Period and Preperiod of a function F

Algorithm 41 (Floyd's Cycle finding)
This algorithm proceeds in three distinct stages.

I: A multiple of the period is calculated

i) Set $i := 1$, $a := s$, $b := F(s)$.

ii) While $a \neq b$ set $i := i + 1$ and $a := F(a) = F^i(s)$ and $b := F^2(b) = F^{2i}(s)$.

iii) If $a = b$ then i is a multiple of r.

II: The preperiod is determined

i) Set $p := 0$, $c := s$, $d := a = F^i(s)$.

ii) While $c \neq d$ set $p := p + 1$ and $c := F(c) = F^p(s)$ and $d := F(d) = F^{i+p}(s)$.

iii) When $c = d$ then p is the preperiod.

III: The period is exactly determined

i) Set $r := 1$, $e := c = F^p(s)$, $f := F(c) = F^{p+r}(s)$.

ii) While $e \neq f$ set $r := r + 1$ and $f := F(f) = F^{p+r}(s)$.

iii) When $e = f$ then r is the period.

This algorithm is very memory efficient and only uses two variables at any given state apart from the two values that are returned, it uses $\mathcal{O}(p + r)$ operations.

It was shown in [7] that if F is chosen uniformly random from the set of all mappings from a set of size n onto itself and for uniformly random chosen starting point s the expected value for the preperiod p and the period r both can be estimated by $\sqrt{\frac{\pi n}{8}}$. The first i such that $F^i(s) = F^{2i}(s)$ is called the **epact** and it is easily checked that

$i = \lceil \frac{p}{r} \rceil r$ is the epact when s is not a fixpoint of F. Pollard states that the expected value for the epact can be estimated by

$$E(\text{epact}) \approx \frac{1}{12}\sqrt{\frac{\pi^5 n}{2}}. \tag{2.1}$$

Now given a DLP $\log_g h$ on a group $G := \langle g \rangle$ of order n, consider a function $f : G \to G$ defined by choosing a partition of $G = G_1 \sqcup G_2 \sqcup G_3$ into roughly equally sized sets.

$$f(j) = \begin{cases} jg, & \text{for } j \in G_1 \\ j^2, & \text{for } j \in G_2 \\ jh, & \text{for } j \in G_3. \end{cases}$$

For the analysis of the running time we assume that the epact of f is indeed of the magnitude given in (2.1). Should a particular f not have sufficiently small epact for a given s then it can easily be altered by choosing a different partition of G.

The function is chosen such that it can be extended to a special subset $X \subset G \times \mathbb{Z}_n \times \mathbb{Z}_n$ where $(j, a, b) \in X \Leftrightarrow j = g^a h^b$. We define a function F on X by

$$F\left((j, a, b)\right) = \begin{cases} (jg, a+1, b), & \text{for } j \in G_1 \\ (j^2, 2a, 2b), & \text{for } j \in G_2 \\ (jh, a, b+1), & \text{for } j \in G_3 \end{cases}.$$

Every element $f \in G$ has a unique representation $f = g^e$ where $e \in \mathbb{Z}_n$ as a power of g. But if we allow for representations of the form $f = g^a h^b = g^{a+bx}$ then we see that every element has n different representations. If we obtain two different representations of the same element, say

$$g^a h^b = g^{a'} h^{b'} \tag{2.2}$$

with $(a, b) \neq (a', b')$ then it follows that

$$g^{\frac{a-a'}{b'-b}} = h$$

therefore

$$x = \frac{a - a'}{b' - b}$$

where calculations in the exponent are done in $\mathbb{Z}_n$, provided $b' - b \in \mathbb{Z}_n^*$. Let φ denote Euler's phi function, then the element $b' - b$ is invertible with probability $\frac{\varphi(n)}{n-1}$ (since we can exclude the case where $b = b'$). It should be noted that in the case that is of most interest, when n is a prime, the probability is 1 and every collision leads to a solution of the DLP.

We are now ready to describe Pollard's Rho algorithm.

Algorithm 42 (Pollard's Rho)
Given a DLP $\log_g h$ in a group G of order n choose random $a, b \in \mathbb{Z}_n$, calculate $j = g^a h^b$ and let $s = (j, a, b) \in X$. Calculate $F^i(s) = (f^i(j), a_i, b_i)$ and $F^{2i}(s) = (f^{2i}(j), a_{2i}, b_{2i})$ for increasing values of i till the epact e of f for starting value j is reached, i.e. $f^e(j) = f^{2e}(j)$. If $b_{2i} - b_i$ is invertible in $\mathbb{Z}_n$ then

$$\log_g h = \frac{a_i - a_{2i}}{b_{2i} - b_i}.$$

If the algorithm fails because $b_{2i} - b_i$ is not invertible then it can be restarted with a different partition of G and different values for a and b. The algorithm has a minimal memory requirement of only two elements of the set X. Using the assumption that the epact for f is as stated in (2.1), we expect a collision after $\mathcal{O}(\sqrt{n})$ steps, comparable to Shank's algorithm. With high probability, F will have a different epact from f and this will lead to two different representations of the same element as in (2.2). A drawback compared to Shank's algorithm is that the algorithm no longer is deterministic and might not succeed at all.

2.5 Pohlig-Hellman

Only two years after the seminal paper by Diffie and Hellman [5], Pohlig and Hellman published an attack [25] on the DLP that is very powerful for groups of smooth order n.

Definition 43 (B-smooth)
Let $B \in \mathbb{N}$ then a number $n \in \mathbb{N}$ is said to be B-smooth if for its prime factor decomposition

$$n = \prod_{i=1}^{k} p_i^{e_i}$$

it holds that $p_i \leq B$ for all $1 \leq i \leq k$. We call a number smooth if it is B-smooth for $B \ll n$.

We will first describe the attack on a toy example and then give a general description.

Example 44
Find $x \in \mathbb{Z}_{28}$ such that $2^x = 18$ in $\mathbb{Z}_{29}^*$, i.e. compute $\log_2(18)$ in $\mathbb{Z}_{29}$.
We split this into smaller problems to obtain partial information about x. The base element 2 is a generator for the multiplicative group of $\mathbb{Z}_{29}$ and therefore has order 28. The Chinese remainder theorem (CRT) describes an isomorphism

$$\begin{aligned} \varphi : \mathbb{Z}_{28} &\to \mathbb{Z}_7 \times \mathbb{Z}_4 \\ z &\mapsto (z_7, z_4) \\ \text{where } z_i = z \mod i & \end{aligned}$$

and we identify $x \in Z_{28}$ with its image $x = (x_7, x_4)$. It is then obvious that

$$(x_7, x_4) \cdot (1, 0) = x \cdot 8 = (x_7, 0)$$

where the factor 8 can easily calculated by applying φ^{-1} to $(1, 0)$. Using the commutativity of the exponents we see that

$$2^x = 18 \quad \Rightarrow \quad (2^x)^8 = 24^x = 16 = 18^8.$$

This new DLP $\log_{24}(16)$ is much simpler since 24 has order 7. In the worst case 6 different exponents have to be tried to determine that $24^4 \equiv 16 \mod 29$, this is equivalent to $8 \cdot x \equiv 8 \cdot 4 \mod 28$ and hence $x \equiv 4 \mod 7$. Similarly it can be determined after at most three tries that $17^3 \equiv 12 \mod 29$ and therefore $x \equiv 3 \mod 4$.

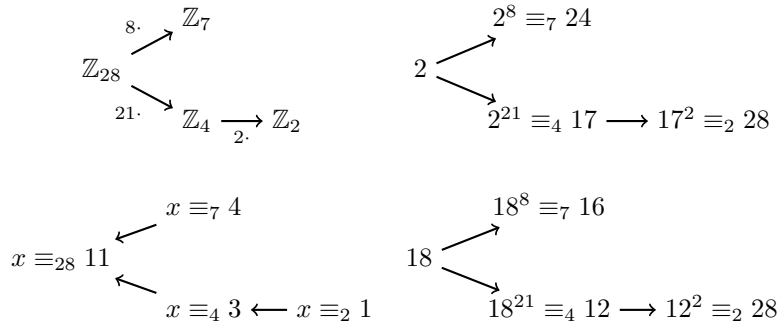

Figure 2.9: Pohlig-Hellman Example

For general $\mathbb{Z}_n$ with $n = a \cdot b$ and $\gcd(a, b) = 1$ it seems as if the attack is based on the CRT or the isomorphism $\mathbb{Z}_n \cong \mathbb{Z}_a \times \mathbb{Z}_b$. But this explanation falls short when considering the reduction from $\mathbb{Z}_4$ to $\mathbb{Z}_2$ in the above example. This reduction is nicely explained by using a p-adic representation of the exponent, but we would like to introduce a different more general explanation that will be easier to extend to semigroup actions.
The following Lemma provides the basis for the Pohlig-Hellman attack.

Lemma 45 *Let $n \in \mathbb{N}$ and $0 < m < n$ such that $\gcd(m, n) = h$. Then the following hold:*

i) The map $\mu_m(z) = mz$ is a group morphism on $(\mathbb{Z}_n, +)$.

ii) $\operatorname{im}(\mu_m) \cong \mathbb{Z}_{\frac{n}{h}}$.

iii) $\ker(\mu_m) \cong \mathbb{Z}_h$.

If n is not prime then we can find elements m such that the $gcd(m, n) > 1$ and therefore $|\operatorname{im}(\mu_m)| < |\mathbb{Z}_n|$. Given a DLP $x = \log_g h$ where $x \in \mathbb{Z}_n$ we can reduce the problem by choosing such an m and transforming the problem into the problem of computing $\mu_m(x) = \log_g h^m$. This problem is easier than the original problem since the search space is smaller. Given a solution $\mu_m(x) = z$ to this smaller problem we can restrict the original problem to $\mu_m^{-1}(z)$. Effectively this transformation allows us to replace a problem of size n with two problems of sizes $\frac{n}{h}$ and h.
The strength of this approach is twofold:

a) The reduction can be applied recursively. The smaller problem is a DLP in $\mathbb{Z}_{\frac{n}{h}}$. If $\frac{n}{h}$ has a non-trivial divisor h' another reduction can be applied, splitting the problem of size $\frac{n}{h}$ into two problems of sizes $\frac{n}{hh'}$ and h'.

b) The reductions can be applied in parallel. Given several divisors $h_1, \ldots, h_i$ of n a DLP in $\mathbb{Z}_n$ can be reduced to i different smaller problems of sizes $\frac{n}{h_1}, \ldots, \frac{n}{h_i}$. The solution to the original problem then has to lie in the intersection of all preimages which is of size $\frac{n}{\operatorname{lcm}(h_1, \ldots, h_i)}$.

For a DLP in $\mathbb{Z}_n$ the size of the biggest problem that can not be split further is of size p where p is the biggest prime dividing n. This DLP of size p can be treated by one of the algorithms presented in the previous section with complexity $\mathcal{O}(\sqrt{p})$. Given the prime decomposition of $n = \prod_{i=1}^{m} p_i^{a_i}$ it follows that the computational effort is of order $\mathcal{O}(ma\sqrt{p})$ where p is the biggest prime dividing n and a the biggest exponent in the

decomposition. It should be noted though that the key length is $\sum_i a_i \lceil \log_2 p_i \rceil$, hence it follows that the computational effort grows linear wrt. the keylength in a while it grows exponential in the prime p. Therefore the complexity relates directly to how smooth the number n is.

2.6 Quantum Algorithms

The search for alternatives to the factoring and the discrete logarithm problem has gained a lot of attention since Shor presented his algorithm [29] in 1994 that enables quantum computers to solve both these problems efficiently. A whole new field of so-called post quantum cryptography has been established and as of now, four different candidates of computational problems have emerged: coding based cryptography, lattice based systems, multivariate polynomials and hash functions.
In this section we give a short overview of how quantum algorithms can be used to attack the DLP and provide some intuition as to why the same line of attack will not work in the semigroup setting, unless further breakthroughs in the field of quantum algorithms are achieved.

Quantum algorithms work on what are called qubits as opposed to boolean bits. A qubit is a unit vector in $\mathbb{C}^2$ represented by basis vectors $|0\rangle$ and $|1\rangle$. The space of all possible states is denoted as $\mathbb{H} := \{\alpha|0\rangle + \beta|1\rangle \mid |\alpha|^2 + |\beta|^2 = 1,\ \alpha, \beta \in \mathbb{C}\}$. Combining m qubits through entanglement leads to a space state that is the tensor product of m copies of the space $\mathbb{H}$ and is denoted by $\mathbb{H}_m := \bigotimes_{i=1}^{m} \mathbb{H}$. The basis vectors of this space have different notations, e.g. in $\mathbb{H}_3$ the following represent the same vector $|1\rangle \otimes |0\rangle \otimes |1\rangle = |101\rangle = |5\rangle$ where the last equality is given by the binary representation of 5.
A collection of entangled qubits is called a quantum register and may be manipulated in one of two ways:

i) The state can be changed by applying a unitary transformation. (In implementations one needs to consider the task of approximating these transformations)

ii) The state of the system can be measured. This measurement will only return one of the basis vectors. For a register of m qubits in state $\sum_{i=0}^{2^m-1} \alpha_i |i\rangle$, where $(\alpha_i) \in \mathbb{C}^{2^m}$ is a unit vector, the probability of reading the basis vector $|i\rangle$ is given by $|\alpha_i^2|$. Upon measuring the system collapses into the state $|i\rangle$ for the basis vector $|i\rangle$ that was read.

One particular unitary transformation of interest to us is the following

Definition 46 (Quantum Fourier Transform)
For a positive $n \geq 1$ a unitary transformation can be defined through

$$\begin{aligned} \mathrm{QFT}_n : \mathbb{C}^n &\to \mathbb{C}^n \\ |a\rangle &\mapsto \frac{1}{\sqrt{n}} \sum_{b=0}^{n-1} \omega^{ab} |b\rangle \end{aligned}$$

where ω is a n-th root of unity, e.g. $\omega = e^{\frac{2\pi i}{n}}$. In matrix form QFT_n is represented as

$$\mathrm{QFT}_n = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & & \cdots & & 1 \\ & \omega & \omega^2 & \cdots & \omega^{n-1} \\ \vdots & \omega^2 & \omega^4 & \cdots & \omega^{2(n-1)} \\ & \vdots & \vdots & & \vdots \\ 1 & \omega^{n-1} & \omega^{2(n-1)} & \cdots & \omega^{(n-1)^2} \end{pmatrix}.$$

It should be noted that QFT_n can be applied to any quantum register of at least $m := \lceil \log_2(n) \rceil$ qubits by having QFT_n act on n of the 2^m basis vectors and leaving the other coefficients untouched.
This transformation can be effectively approximated in quantum computers [23].

We now investigate a class of problems that can be effectively solved by quantum algorithms.

Definition 47 (Hidden Subgroup Problem (HSP))
Let $G > H$ be groups, X a finite set, and $f : G \to X$ such that f is constant on cosets of H but takes on distinct values on each coset. Let $m := \lceil log_2(n) \rceil$ and assume that the unitary transformation

$$\begin{aligned} U : \mathbb{H}_{2m} &\longrightarrow \mathbb{H}_{2m} \\ |g\rangle|x\rangle &\longmapsto |g\rangle|x \oplus f(g)\rangle, \end{aligned}$$

where $x \in X$ and $\oplus$ is the componentwise addition mod 2 on binary representations of X, can be efficiently performed. Find a generating set for H.

Most problems, that can be solved efficiently with quantum algorithms, translate into instances of the HSP. In these cases the group is a finite abelian group and a quantum algorithm need about $\log(|G|)$ operations to solve the problem, see [23] for more details.

For non-commutative groups the problem seems to be much harder and only little progress has been made, see for example [21] for information on treating the symmetric group.
The DLP $\ell = \log_g h$ in a group $\langle g \rangle$ of order p translates into a HSP in the group $(\mathbb{Z}_p, +) \times (\mathbb{Z}_p, +)$. The subgroup in question is the group H generated by $(\ell, 1)$. We will present the algorithm for the DLP in Algorithm 49 after this lemma.

Lemma 48 *Let $\omega \neq 1$ be a n-th root of unity, then $\sum_{j=0}^{n-1} \omega^j = 0$.*

PROOF i) Assume ω is a primitive n-th root. Then the powers of ω generate all possible n-th roots and the sum in question is the coefficient for x^{n-1} in the product $\prod_{j=0}^{n-1}(x - \omega^j) = x^n - 1$ and therefore 0.

ii) If ω is not primitive then $\omega = \zeta^k$ for a primitive root ζ with $\gcd(k, n) = d > 1$. It follows that ω is of order $m = \frac{n}{d}$ and therefore a primitive m-th root. Furthermore $\sum_{j=0}^{n-1} \omega^j = d \sum_{j=0}^{m-1} \omega^j$ and by i) $\sum_{j=0}^{m-1} \omega^j = 0$. ∎

Algorithm 49 (Shor's Algorithm [29])
We want to solve the DLP $\ell = \log_g h$ in the group $\langle g \rangle$ of prime order p and let $m := \lceil log_2(p) \rceil$. We therefore define a function $f(x, y) := g^x h^y = g^{x+\ell y}$. Let T be the subspace of $(\mathbb{Z}_p, +) \times (\mathbb{Z}_p, +)$ generated by $(1, \ell)$.

i).. Start with a quantum system in state $|0, 0, 0\rangle \in \bigotimes^3 \mathbb{H}_m$.

ii).. Apply QFT_p to the first two registers, this brings the system into the uniform state $\frac{1}{p} \sum_{x=0}^{p-1} \sum_{y=0}^{p-1} |x, y, 0\rangle$.

iii).. Apply f to our quantum state and get $\frac{1}{p} \sum_{x=0}^{p-1} \sum_{y=0}^{p-1} |x, y, f(x, y)\rangle$.

iv).. Perform QFT_p on the first two registers.

v).. A measurement of the first two registers will output an element of $T^\perp$.

PROOF After step *iii)* the system is in the state

$$\frac{1}{p} \sum_{x=0}^{p-1} \sum_{y=0}^{p-1} |x, y, f(x, y)\rangle.$$

We sort by the last entry $f(x,y) = g^e$ it then follows that $x = e - \ell y$

$$\frac{1}{p}\sum_{e=0}^{p-1}\sum_{y=0}^{p-1}|e-\ell y, y, g^e\rangle.$$

Step *iv*) applies QFT_p to the first two registers leading to

$$\frac{1}{p}\sum_{e=0}^{p-1}\sum_{y=0}^{p-1}\frac{1}{\sqrt{p}}\sum_{s=0}^{p-1}\omega^{(e-\ell y)s}\frac{1}{\sqrt{p}}\sum_{t=0}^{p-1}\omega^{ty}|s,t,g^e\rangle = \frac{1}{p^2}\sum_{e=0}^{p-1}\sum_{y=0}^{p-1}\sum_{s=0}^{p-1}\sum_{t=0}^{p-1}\omega^{(e-\ell y)s+ty}|s,t,g^e\rangle.$$

We rearrange

$$\frac{1}{p^2}\sum_{e=0}^{p-1}\sum_{s=0}^{p-1}\omega^{es}\sum_{t=0}^{p-1}\sum_{y=0}^{p-1}\omega^{(t-\ell s)y}|s,t,g^e\rangle$$

and use Lemma 48 to see that $\sum_{y=0}^{p-1}\omega^{(t-\ell s)y} \neq 0$ iff $t - \ell s = 0$. It follows that only vectors with $t = \ell s$ remain in the sum

$$\frac{1}{p}\sum_{e=0}^{p-1}\sum_{s=0}^{p-1}\omega^{es}|s,\ell s,g^e\rangle.$$

Now measuring the first two registers will with probability $\frac{p-1}{p}$ return a non-zero element $(x,y) = (s,\ell s)$ of $T^\perp$. We calculate the discrete logarithm as $\ell = yx^{-1}$. ∎

Conclusion We see that Algorithm 49 relies heavily on additional properties of the exponentiation, i.e. the semigroup action of $(\mathbb{Z}_n,\cdot)$ on a cyclic group. For fixed $g \in G$ the map $\Psi_g : (\mathbb{Z}_n,+) \to G,\ z \mapsto g^z$ is a group isomorphism of cyclic, hence abelian, groups. Only this makes the treatment with the quantum Fourier transform possible. The treatment of non-abelian groups is an ongoing topic in research and only partial results have been achieved. It has been shown [22] that for the symmetric group an approach using Fourier sampling will not solve the hidden subgroup problem. For a general semigroup action we are not aware of a translation into a hidden subgroup problem and it is unlikely that this is possible unless additional structure of the action and the semigroup are assumed.

3 Key-Exchange based on Semigroups

3.1 Semigroup Discrete Logarithm

The Diffie-Hellman protocol as described in Protocol 30 is based on the hardness of the DLP. Pollard's Rho attack relies on the fact that the DLP is a $\mathbb{Z}_n$ action on a group G and it is possible to calculate linear functions in the exponents by $(g^x)^a g^b = g^{ax+b}$.
A naive approach to counter this attack is to replace G with a semigroup S with few units. A key exchange based on this idea has been proposed in [11]. An analysis of this protocol by Banin and Tsaban [2] showed that it is vulnerable in a quantum setting and can be reduced to a standard DLP in a group.

The authors of [11] suggest the semigroup $\mathrm{Mat}_{3\times 3}(\mathbb{Z}_7[S_5])$ and the following variation on the Diffie-Hellman protocol (Protocol 30).

Protocol 50 (Semigroup Diffie-Hellman Protocol)

0.) Setup:
The protocol parameters are negotiated. These include the semigroup S and an element $\alpha \in S$.

1.) Generation of public/private keys:
Both parties, Alice and Bob, choose secret elements $s_A, s_B \in \mathbb{N}$ respectively as their secret keys and calculate their public keys as $p_A = \alpha^{s_A}, p_B = \alpha^{s_B}$.

2.) Exchange of public keys:
Alice and Bob exchange their public keys p_A and p_B.

3.) Calculating the shared key:
Alice, upon receiving p_B from Bob, calculates $k_A := p_B^{s_A} = \alpha^{s_B s_A}$.
Bob similarly computes $k_B := p_A^{s_B} = \alpha^{s_A s_B}$.

Correctness of the protocol follows by commutativity in $\mathbb{N}$.

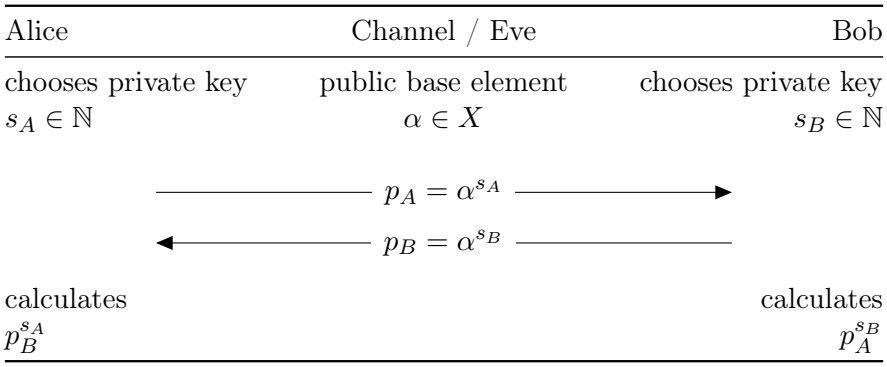

Figure 3.1: Protocol 50

In the aforementioned paper the authors suggest that Alice chooses the base element α at the beginning of the protocol. Since the order of the element α might not be known the two parties choose their secrets in $\mathbb{N}$.
The preperiod and period of α determine the size of the set of possible keys and are crucial to the hardness of the following problem.

Definition 51 (Semigroup DLP)
Given a semigroup S and elements $\alpha, \beta \in S$ such that $\beta = \alpha^x$ for some $x \in \mathbb{N}$ determine $y \in \mathbb{N}$ such that $\beta = \alpha^y$.

If the preperiod and period p, r of α are known, we can restrict the answer to the unique value $0 \leq y \leq p + r - 1$ that solves the semigroup DLP $\alpha^y = \beta$. We will refer to this value as $y = \operatorname{slog}_\alpha \beta$.

Theorem 52 (see [2]) *A semigroup* DLP $\operatorname{slog}_\alpha \beta$ *can either be reduced to a* DLP *in a group of order* r *or solved in* $\mathcal{O}(\log(p))$ *steps if the preperiod* p *and period* r *of* α *are known.*

Proof Split the semigroup generated by α into two sets $L := \{\alpha^i : 1 \leq i < p\}$ and $C := \{\alpha^i : p \leq i < p + r\}$. The set C is in fact a group of order r isomorphic to $\mathbb{Z}_r$ by the morphism $\varphi : \alpha^i \mapsto i \mod r$. Given an element $\gamma \in \langle\alpha\rangle$ we realize that $\gamma \in C$ iff $\gamma\alpha^r = \gamma$ (this can be exchanged with $\gamma^{r+1} = \gamma$). If $\beta \in C$ then the problem has been reduced to a DLP in a group of order r. The generator for the group is the element α^g where g is the unique integer such that $p \leq g < p + r$ and $g = 1 \mod r$.
If $\beta = \alpha^x \in L$ then we know that $0 \leq x < p$. Assume we have an element $\gamma = \alpha^y$ and we know that $a \leq y < b \leq p$. If $\gamma\alpha^{p-b+\lfloor\frac{b-a}{2}\rfloor} \in C$ then it follows that $y + \lfloor\frac{b-a}{2}\rfloor \geq p$

which implies $b - \lfloor \frac{b-a}{2} \rfloor \leq y < b$. If $\gamma\alpha^{p-b+\lfloor \frac{b-a}{2} \rfloor} \in L$ then we get $y + \lfloor \frac{b-a}{2} \rfloor < p$ which leads to $a \leq y < b - \lfloor \frac{b-a}{2} \rfloor$. We can use this method repeatedly starting with β and the interval $0 \leq x < p$ till x is found after $\lceil \log_2 p \rceil$ steps. ∎

We summarize this section in the following corollary.

Corollary 53 *Assuming that preperiod and period of the involved elements are known or computable, a semigroup* DLP *only achieves maximal hardness if* $p = 0$ *and* r *is prime.*

In the case $p = 0$ the semigroup DLP is equivalent to a DLP in a group of order r hence making this approach weaker or at most as hard as currently known methods.

3.2 Key-Exchange Protocols based on Semigroup Actions

A different and more promising approach follows from the realization that the exponentiation of group elements can be more generally described as a semigroup action. This naturally leads to a new description of the Diffie-Hellman protocol in terms of semigroup actions

Protocol 54 (Diffie-Hellman with Semigroup Action)

0.) Setup:
The protocol parameters are negotiated. These are the semigroup S and an S-set X together with a base element $\alpha \in X$.

1.) Generation of public/private keys:
Both parties, Alice and Bob, choose secret elements $s_A, s_B \in S$ respectively as their secret keys and calculate their public keys as $p_A = s_A \,.\, \alpha$ and $p_B = s_B \,.\, \alpha$.

2.) Exchange of public keys:
Alice and Bob exchange their public keys p_A and p_B.

3.) Calculating the shared key:
Alice, upon receiving p_B from Bob, calculates $k_A := s_A \,.\, p_B = (s_A \cdot s_B) \,.\, \alpha$.
Bob similarly computes $k_B := s_B \,.\, p_A = (s_B \cdot s_A) \,.\, \alpha$.

To ensure correctness, i.e. $k_A = k_B$, Alice and Bob might have to be restricted to choosing their secret keys from commuting subsemigroups $S_A, S_B \subseteq S$.

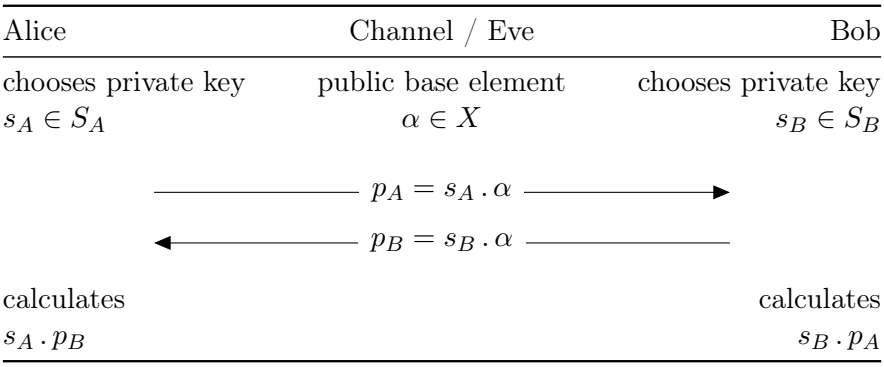

Figure 3.2: Protocol 54

The security of this protocol relates to similar problems as in the case of Protocol 30. The task of retrieving the secret key from the public key is equivalent to the following problem.

Definition 55 (Semigroup action problem (SAP))
Given an S-set X and two elements $x \in X$ and $y \in S \,.\, x$, find an element $s \in S$ such that $s \,.\, x = y$.
The solution to this problem is not necessarily unique and we define the set of all solutions as $\operatorname{sap}_x y := \{s \in S \,:\, s \,.\, x = y\}$.

If Eve has access to an algorithm that solves the SAP in the semigroup S_A she can retrieve Alice's secret key s_A from her public key p_A, or at least an equivalent element $s_E \sim_\alpha s_A$ under the equivalence relation introduced in Definition 18. With this element Eve can compute the shared secret as

$$s_E \,.\, (s_B \,.\, \alpha) = (s_E s_B) \,.\, \alpha = (s_B s_E) \,.\, \alpha = s_B \,.\, (s_E \,.\, \alpha) = s_B \,.\, (s_A \,.\, \alpha) = k_B.$$

We point out that Eve has to solve the SAP in the subsemigroup S_A to ensure commutativity with s_B.
Calculating the shared secret from the public keys is described by the following problem.

Definition 56 (Semigroup Action Diffie-Hellman Problem (SADHP))
Let X be an S-set. Given the elements $\alpha, s \,.\, \alpha$ and $t \,.\, \alpha$ calculate $st \,.\, \alpha$.

Remark 57
There exists a Turing reduction SADHP α_T SAP if Protocol 54 achieves correctness.

PROOF Given an algorithm $\mathcal{L}$ that solves the SAP in S_A, i.e. $\mathcal{L}(\alpha, \beta) \in \operatorname{sap}_\alpha \beta \cap S_A$ we can design an algorithm $\mathcal{C}$ that solves the SADHP in polynomial time. Given an instance $(\alpha, s\,.\,\alpha, t\,.\,\alpha)$ of the SADHP the algorithm $\mathcal{C}$ calls $\mathcal{L}$ on $(\alpha, s\,.\,\alpha)$ obtaining $s' \sim_\alpha s$ and then calculates $s'\,.\,(t\,.\,\alpha) = s't\,.\,\alpha$. Since s' could have been Alice's secret as far as Bob could tell from the execution of the protocol, it follows that $s't\,.\,\alpha = st\,.\,\alpha = k_A$ assuming correctness of the protocol. ■

The indistinguishability of the shared secret from random elements is captured in the definition below.

Definition 58 (SADDHP)
Let X be an S-set. We call a tuple $(\alpha, s\,.\,\alpha, t\,.\,\alpha, \beta) \in X^4$ a valid Diffie-Hellman tuple if $\beta = st\,.\,\alpha$. Given a tuple of group elements $(\alpha, s\,.\,\alpha, t\,.\,\alpha, \beta)$ that is with probability $\frac{1}{2}$ a valid Diffie-Hellman tuple and in all other cases β is chosen uniformly random from $S\,.\,\alpha$, decide whether the tuple is valid.

Remark 59
A similar Turing reduction SADDHP α_T SADHP as in the classical case applies.

PROOF Given an algorithm $\mathcal{C}$ that solves the SADHP, i.e. $\mathcal{C}(\alpha, s\,.\,\alpha, t\,.\,\alpha) = st\,.\,\alpha$ we can construct an algorithm $\mathcal{D}$ that solves the SADDHP. Given an instance $(\alpha, s\,.\,\alpha, t\,.\,\alpha, \beta)$ of the $SADDHP$ the algorithm $\mathcal{D}$ calls the subroutine $\mathcal{C}$ on $(\alpha, s\,.\,\alpha, t\,.\,\alpha)$ and receives $st\,.\,\alpha$ as a result. If $\beta = st\,.\,\alpha$ then $\mathcal{D}$ knows that the triple is valid, otherwise it is not. ■

We will study the SAP for several examples of semigroup actions in the following sections.

3.3 Comparison of Diffie-Hellman and Anshel-Anshel-Goldfeld

An interesting comparison of the Diffie-Hellman protocol and the Anshel-Anshel-Goldfeld protocol was conducted in [30]. More precisely, they compared the Ko-Lee protocol from [15] to Protocol 36. The Ko-Lee protocol is in our notation an instance of Protocol 54. They propose to use conjugation in a group G as a group action of G on itself. Hence for $g, x \in G$ the action is given as $\varphi : (g, x) \mapsto gxg^{-1}$. In both cases the authors suggest using braid groups. Although the use of inner automorphisms is a common feature of both protocols, there are differences in the problems an adversary faces.

Protocol	Secret Keys	Public Keys
Ko-Lee	$s_A \in LB_\ell$	$p_A := s_A x s_A^{-1}$
	$s_B \in RB_r$	$p_B := s_B x s_B^{-1}$
Anshel-Anshel-Goldfeld	$s_A \in B_n$	$p_A := (s_A x s_A^{-1})_{x \in S}$
	$s_B \in B_n$	$p_B := (s_B x s_B^{-1})_{x \in S}$

Table 3.1: Comparison Ko-Lee, Anshel-Anshel-Goldfeld

For the Ko-Lee protocol the choices for the secret keys are restricted to commuting subsets $LB_\ell := \langle \sigma_1, \ldots, \sigma_{\ell-1} \rangle$ and $RB_r := \langle \sigma_{\ell+1}, \ldots, \sigma_n \rangle$, where $\ell + r = n$, of B_n. The public key is the image of the corresponding inner automorphism of one element $x \in B_n$. In the Anshel-Anshel-Goldfeld protocol the secret keys are chosen from B_n. While the public keys are the images of a generating subset $S \subset B_n$ under the corresponding inner automorphism.

In [30] the authors investigate the relation between a special instance of the SAP that they call the (subgroup) conjugacy search problem and the task of retrieving the shared secret from the public keys, i.e. the SADHP.

Definition 60 (Subgroup Conjugacy Search Problem (CSP))
Given a group G and two elements $g, h \in G$, find $j \in J < G$ such that $jgj^{-1} = h$.

For the Ko-Lee protocol it is sufficient to solve the CSP and find $j \in LB_n$ such that $jxj^{-1} = p_A$. An adversary then calculates the shared secret the same way Alice does $jp_Bj^{-1} = js_Bxs_B^{-1}j^{-1} = s_Bjxj^{-1}s_B^{-1} = s_Bs_Axs_A^{-1}s_B^{-1} = K$. But in fact it suffices for an adversary to solve a simpler, less restrictive problem.

Definition 61 (Decomposition Problem)
Given a group G and two elements $g, h \in G$ find $j, k \in J < G$ such that $jgk = h$.

This problem is equivalent to the CSP with the additional requirement that $k = j^{-1}$. But even without this condition, given $j, k \in LB_n$ such that $jxk = s_Axs_A^{-1}$ the shared secret can be computed as $jp_Bk = js_Bxs_B^{-1}k = s_Bjxks_B^{-1} = s_Bs_Axs_A^{-1}s_B^{-1} = K$. Hence the SADHP for the Ko-Lee protocol is at most as hard as the CSP.

The situation is quite different for the Anshel-Anshel-Goldfeld protocol. An adversary is given several simultaneous conjugates $(s_Axs_A^{-1})_{x \in S}$ instead of just one. But finding an

element $k \in B_n$ such that $kxk^{-1} = s_A x s_A^{-1}$ for all $x \in S$ is not sufficient to calculate the shared secret. An adversary would further need to be able to describe the solution as a composition of the elements of S.
Given $j, k \in B_n$ such that $jxk = s_A x s_A^{-1}$ for all $x \in S$ and a decomposition of $k = \prod_{i=1}^{m} x_i$ into generators $x_i \in S$ the shared secret can be computed as

$$j \prod_{i=1}^{m} s_B x_i s_B^{-1} = j s_B k s_B^{-1} = s_A s_B s_A^{-1} s_B^{-1} = K.$$

The SADHP is therefore at least as hard as the (simultaneous) CSP in the Anshel-Anshel-Goldfeld protocol.

3.4 Monico-Maze-Rosenthal Protocol

The semigroup action from Example 17 has been suggested for application in a Diffie-Hellman-like key-exchange protocol in [18]. Let as before S a finite congruence-simple semiring, $\mathcal{M} := \mathrm{Mat}_{n\times n}(S)$ with one. As shown in [35] the center of any finite congruence-simple semiring with one is just $C = \{0, 1\}$. Let $L, R \in \mathcal{M}$ be two fixed matrices, then for the protocol the semigroup was restricted to the commutative subsemigroup

$$P_{L,R} := \{(f(L), g(R)) : f, g \in C[x]\} \subseteq \mathcal{M} \times \mathcal{M}^{op}.$$

The authors further suggest limiting the degrees of the involved polynomials, effectively working in a subset of $P_{L,R}$

$$P_{L,R}^{(d_L,d_R)} := \{(f(L), g(R)) : f, g \in C[x], \deg(f) \leq d_L, \deg(g) \leq d_R\} \subseteq P_{L,R}.$$

Protocol 62 (Monico-Maze-Rosenthal Protocol (MMR))

0.) Setup:
The protocol parameters are negotiated. These are the semiring S, the matrices $L, R, M \in \mathcal{M}$ and $d_L, d_R \in \mathbb{N}$.

1.) Generation of public/private keys:
Both parties, Alice and Bob, choose secret elements $s_A, s_B \in P_{L,R}^{(d_L,d_R)}$ respectively as their secret keys and calculate their public keys as $p_A = s_A \,.\, M = f_A(L) \cdot M \cdot g_A(R)$ and $p_B = s_B \,.\, M = f_B(L) \cdot M \cdot g_B(R)$.

2.) Exchange of public keys:
Alice and Bob exchange their public keys p_A and p_B.

3.) Calculating the shared key:
Alice, upon receiving p_B from Bob, calculates $K_A := s_A \,.\, p_B = f_A(L) \cdot f_B(L) \cdot M \cdot g_B(R) \cdot g_A(R) = (s_A \cdot s_B) \,.\, M$.
Bob similarly computes $K_B := s_B \,.\, p_A = f_B(L) \cdot f_A(L) \cdot M \cdot g_A(R) \cdot g_B(R) = (s_B \cdot s_A) \,.\, M$.

Correctness follows from commutativity of $P_{L,R}^{(d_L,d_R)}$.

Alice	Channel / Eve	Bob
chooses private key $(f_A(L), g_A(R)) \in P_{L,R}^{(d_L,d_R)}$	public base element $M \in \mathcal{M}$ and semiring $P_{L,R}^{(d_L,d_R)}$	chooses private key $(f_B(L), g_B(R)) \in P_{L,R}^{(d_L,d_R)}$
	$\longrightarrow\ f_A(L)Mg_A(R)\ \longrightarrow$	
	$\longleftarrow\ f_B(L)Mg_B(R)\ \longleftarrow$	
calculates $f_A(L)f_B(L)Mg_B(R)g_A(R)$		calculates $f_B(L)f_A(L)Mg_A(R)g_B(R)$

Figure 3.3: Protocol 62

Remark 63
In [18] the authors state that "For cryptographic purposes it is important that the involved semirings are simple to avoid a Pohlig-Hellman type reduction of the SAP". This statement is unfounded for several reasons that we would like to discuss.

1.) The authors choose matrix rings over semirings since these are simple iff the base semiring is [18, Lemma 5.5]. In particular they find the semiring R_6 (see Example 11) by computation and choose to work in $\mathrm{Mat}_{20\times 20}(R_6)$. Although this semiring is simple, their protocol actually uses the subsemiring $P_{L,R}$ instead of $\mathrm{Mat}_{20\times 20}(R_6)$.

2.) Furthermore they consider simple semirings although the cryptographic application only uses the multiplicative semigroup structure. Therefore the simplicity of the *ring* does not relate to the SAP, but the simplicity of the multiplicative semigroup would be of importance.

We will illustrate the Remark 63 with the following examples.

Example 64
Although R_6 is simple as a semiring, neither of the semigroups $(R_6, +)$ and $(R_6, \cdot)$ are.

a) For $(R_6, +)$ the following partitions are examples of congruence classes of an associated congruence relation.

$$C_1 = \{\{0, 4\}, \{1, 2, 3\}, \{5\}\}$$
$$C_2 = \{\{0\}, \{1, 2, 3, 4\}, \{5\}\}$$
$$C_3 = \{\{0, 4\}, \{1, 2, 5\}, \{3\}\}$$

b) On $(R_6, \cdot)$ the relation that separates 1 from all other elements is a congruence relation, as the relation that separates units from non-units always is.

To discuss the simplicity of the multiplicative matrix group proposed we can use the structure theorems on congruence-simple semigroups. Finite simple semigroups can be classified using the two following theorems (see [9]).

Theorem 65 *Let S be a finite congruence-simple semigroup with $|S| > 2$ without a zero element then S is a finite simple group.* □

Theorem 66 *Let $S := I \times J \cup \{0\}$ for I, J finite sets of cardinalities m and n. Let P be a $n \times m$ matrix of ones and zeros such that no row or column is identically zero and no two rows or columns are identical. Define a binary operation on S by*

$$(i, j)(k, l) = \begin{cases} (i, l) & \text{if } p_{j,k} = 1 \\ 0 & \text{if } p_{j,k} = 0 \end{cases}$$
$$0(i, j) = (i, j)0 = 0.$$

Together with this operation S becomes a simple semigroup and it can be shown that any semigroup T with zero is simple iff it is isomorphic to a semigroup of this type. □

In fact Theorem 66 guarantees that $(P_{L,R}, \cdot)$ (or $(C[L], \cdot)$) is not a congruence simple semigroup whenever $L \neq E_n$ or $R \neq E_n$. The semigroups $(P_{L,R}, \cdot)$ contain an absorbing element and are constructed to be commutative. In contrast the construction above can never be commutative except for the case when $m = n = 1$. To see this choose indices

i, j, k, l such that $p_{j,k} = 1$ and $p_{i,l} = 0$, this is always possible. Then it follows that

$$(i, j)(k, l) = (i, l)$$
$$\text{but } (k, l)(i, j) = 0.$$

Hence they can not be isomorphic to a semigroup of the type described in Theorem 66 and are therefore not congruence simple.
The semigroup $(\mathrm{Mat}_{n\times n}(R), \cdot)$ for any semiring R with one is neither simple since it has a neutral element and units. The relation that separates units from non-units is therefore a non-trivial congruence on $\mathrm{Mat}_{n\times n}(R)$.

3.4.1 Statistical Analysis of the Monico-Maze-Rosenthal protocol

We investigate the distribution on secret keys for the above protocol, using the challenge parameters from [18].

$$L := \begin{bmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&2&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0
\end{bmatrix}
\quad
R := \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&4\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&3&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&2&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&2&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0
\end{bmatrix}$$

The matrix L has preperiod and period 34 and 420, i.e. $L^{34} = L^{454}$, while for R the preperiod and period are 20 and 60, i.e. $R^{20} = R^{80}$.

$$M := \begin{bmatrix}
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&2&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&5&1&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&2&0&0&0&0&0&0&0&0&0&1\\
0&0&2&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0
\end{bmatrix}$$

For a matrix X denote by $N(r;X) := |\{(i,j) : X_{i,j} = r\}|$ the number of entries that are r. It is easy to see that for random uniform 20×20 matrices X over R_6 the probability for N is given as

$$\Pr\left[N(r;X) = i\right] = \frac{\binom{400}{i}5^{400-i}}{6^{400}}$$

for any $r \in R_6$.

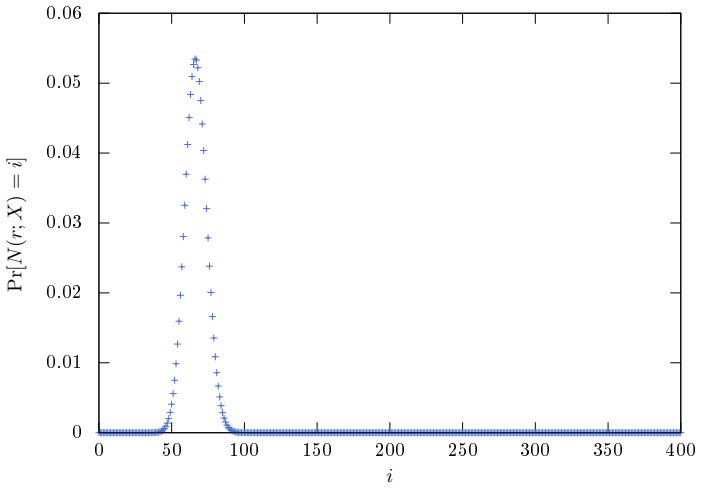

Figure 3.4: Probability of exactly i occurrences of r for uniformly random matrices

This leads to the tail probabilities shown in Table 3.2 that we want to compare with the experimental results.

We simulated the semigroup action for different parameters and noted the number of zeros, ones and fives in the generated public keys. We randomly uniform chose 2^{23} pairs

$\Pr[N(r;M) \leq 20]$	8.37393762858873e-13
$\Pr[N(r;M) \geq 112]$	1.0020732136780253e-08
$\Pr[N(r;M) \geq 200]$	4.38777185472177e-53
$\Pr[N(r;M) \geq 220]$	5.969396492834719e-68
$\Pr[N(r;M) \geq 250]$	1.5140206261192368e-93

Table 3.2: Probabilities for uniformly random matrices

(f, g) of polynomials in $C[x]^{(50)} \times C[x]^{(50)}$ and calculated $X = (f(L), g(R)) \,.\, M$.
$N(1; X)$ took on 168 different values between 107 and 292
$N(0; X)$ took on 74 different values between 19 and 121
$N(5; X)$ took on 11 different values between 0 and 20

For $N(5, X)$ and $N(1, X)$ Table 3.3 summarizes the results.

$N(5;M)$	# of occurrences
0	2098149
6	1
7	69
8	4410
9	204003
10	3984600
12	1
14	23
16	2216
18	102099
20	1993037

$N(1;M)$	# of occurrences	
≥ 200	7294864	87%
≥ 220	6462415	77%
≥ 250	4114861	49%

Table 3.3: Results for polynomials of degree 50

Comparing these results shows that the generated matrices behave quite differently from matrices chosen uniformly at random. This is of course undesirable since it makes the results predictable. Seeing as the secret key follows from another application of the group action, it follows that the secret key will behave similar.

This behavior is easily explained when considering the lattice structure of the additive semigroup $(R_6, +)$. Given two random variables x, y that uniformly random take on values in R_6 the following table shows the distribution of their sum.
These probabilities are of course even more amplified when more than two elements are

z	0	1	2	3	4	5
$\Pr[x = z]$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$	$\frac{1}{6}$
$\Pr[x + y = z]$	$\frac{1}{36}$	$\frac{11}{36}$	$\frac{5}{36}$	$\frac{5}{36}$	$\frac{3}{36}$	$\frac{11}{36}$

Table 3.4: Probabilities of sum of two elements

added. With this in mind we ran another set of experiments for polynomials of low degree. We tested all possible combinations of non-zero polynomials of degrees up to 11, giving us a total number of $(2^{12} - 1)^2$ pairs of polynomials.

$N(1; X)$ took on 280 different values between 9 and 290
$N(0; X)$ took on 307 different values between 20 and 363
$N(5; X)$ took on 16 different values between 0 and 20

The results for $N(5, X)$ and $N(1, X)$ are presented in Table 3.5.

$N(5; M)$	# of occurrences	$N(1; M)$	# of occurrences	
$0 \leq * < 5$	4638721	≥ 200	7294864	71%
$5 \leq * < 10$	7603200	≥ 220	6462415	55%
$10 \leq * < 20$	4527104	≥ 250	4114861	24%

Table 3.5: Results for polynomials of degree 11

The most severe flaw seems to be though the low number of different matrices generated by this procedure as is shown in Table 3.6.

deg	test runs	unique matrices	ratio
80	1.600.000	20839	76.78
50	$2^{23} = 8388608$	156995	53.43
11	$(2^{12} - 1)^2 = 16769025$	6952858	2.41

Table 3.6: unique public keys generated

While polynomials of small degree seem to lead to a semigroup action that produces more varied public keys, the low number of variables of course excludes them from cryptographic applications.

3.4.2 Steinwandt Suárez-Corona Attack

In [32] the authors identify several weaknesses of the proposed structures in [18] that have earlier been mentioned in [17][Assumption 5.19]. They describe a heuristic algorithm that solves the SAP in most cases, and we will present these ideas in this section.
Consider the problem of finding $(f, g) \in \operatorname{sap}_M P$. We start by reducing the problem to a special subset sum problem in the additive semigroup generated by the elements in

$$C[L]_{\leq d_L} MC[R]_{\leq d_R} := \{L^i M R^j \ : \ 0 \leq i \leq d_L, 0 \leq j \leq d_R\}$$

where the numbers d_L and d_R are the restrictions on the degrees of the polynomials. The SAP in Protocol 62 can be represented as a sum by using the linearity of the semigroup action, i.e. the distributivity of the matrix multiplication.

$$\begin{aligned} \text{Let} \qquad & f(L) = \sum_{i=0}^{d_L} f_i L^i \\ \text{and} \qquad & g(R) = \sum_{j=0}^{d_R} g_j R^j \\ \text{then} \qquad & f(L) M g(R) = \sum_{(i,j)} f_i g_j L^i M R^j. \end{aligned}$$

Now the problem of finding polynomials f and g is equivalent to finding a subset of $C[L]_{\leq d_L} MC[R]_{\leq d_R}$ that sums up to P such that the matrix $(f_i g_j)_{i,j}$ is the product of two vectors IJ^T. It seems counter intuitive that this approach helps since we transformed a problem in $d_L + d_R + 2$ independent variables to a problem in $(d_L + 1)(d_R + 1)$ variables with additional conditions on the possible choices. However it should be noted that all the matrices $L^i M R^j$ can be computed beforehand and only addition remains as an operation. The lattice structure of the semiring R_6 as described by Figure 1.2 can now be used to exclude summands $L^i M R^j$ since it holds that

$$A \leq A + B, \qquad \forall A, B \in \operatorname{Mat}_{n \times n} R_6$$

where the partial order is extended to matrices by componentwise comparison. Hence it follows

$$L^i M R^j \not\leq P \Rightarrow f_i g_j = 0.$$

Furthermore we see that for all join irreducible entries in P at least one summand must

carry the same entry at this position. In their work Corona Suarez and Steinwandt claim that after applying these observations a simple greedy algorithm that adds matrices that minimize the Hamming distance between the sum and the challenge element P and then adds the necessary matrices to keep the indices a cartesian product sufficed to solve the problem.

We note that two properties of the suggested semigroup action were exploited in this attack:

i) The semigroup action in Example 17 is linear, i.e. the maps $\varphi(L, R)$ on the set $\mathrm{Mat}_{n\times n}(R)$ are in fact morphisms on the additive semigroup $(\mathrm{Mat}_{n\times n}(R), +)$.

ii) The addition is compatible with a partial order on R_6 as described in Figure 1.2.

4 Attacks on Semigroup Actions

4.1 Brute-Force

For semigroup action problems we will now reconsider brute force attacks. Unless x is free with respect to the action the set of solutions $\operatorname{sap}_x y$ will contain more than one element. This will increase an adversary's chance of finding a solution.
Denote by $p := \frac{|\operatorname{sap}_x y|}{|S|}$ the ratio of solutions within the semigroup S. Consider a brute-force attack with independently uniformly at random chosen elements. The expected number of tests is computed as in the case for the DLP. The probability that the brute-force attack will succeed exactly at the t-th step is $(1-p)^{t-1} p$. Therefore the expected number of tries is

$$\mathrm{E}[t] = p \sum_{t=1}^{\infty} t\,(1-p)^{t-1} = p \frac{1}{(1-(1-p))^2} = \frac{1}{p}.$$

If we consider an attacker that avoids testing elements more than once, the analysis gets a little more complicated. To determine how big the set T of tested elements should be such that an attacker will have found a solution with probability at least $\frac{1}{2}$ we use the following notations, $s := |S|, t := |T|$ and $u := |\operatorname{sap}_x y|$. We approximate and solve for t

$$\begin{aligned} \Pr[T \cap \operatorname{sap}_x y \neq \emptyset] = & \\ 1 - \Pr[T \cap \operatorname{sap}_x y = \emptyset] = & \\ 1 - \frac{\binom{s-u}{t}}{\binom{s}{t}} \geq \frac{1}{2} & \end{aligned}$$

this is equivalent to

$$\begin{aligned} \Leftrightarrow \quad & 2\binom{s-u}{t} \leq \binom{s}{t} \\ \Leftrightarrow \quad & 2 \prod_{i=u}^{u+t-1} (s-i) \leq \prod_{i=0}^{t-1} (s-i) \end{aligned}$$

$$\Leftrightarrow \qquad 2\prod_{i=u}^{t-1}(s-i)\prod_{i=t}^{u+t-1}(s-i) \leq \prod_{i=0}^{u-1}(s-i)\prod_{i=u}^{t-1}(s-i)$$

$$\Leftrightarrow \qquad 2\prod_{i=t}^{u+t-1}(s-i) \leq \prod_{i=0}^{u-1}(s-i)$$

this is the case if

$$\Leftarrow \qquad 2\prod_{i=t}^{u+t-1}(s-i) \leq 2(s-t)^u \leq (s-u+1)^u \leq \prod_{i=0}^{u-1}(s-i)$$

$$\Leftrightarrow \qquad t \geq s - 2^{-\frac{1}{u}}(s-u+1).$$

We note that for $a > b$ the convention $\prod_a^b x = \prod_b^a x^{-1}$ was used. For $s = 2^{160}$ and different values of u Table 4.1 shows a lower bound for expected size of the set T. As

$u := \lvert \operatorname{sap}_x y \rvert$	$\log_2(s - 2^{-\frac{1}{u}}(s-u+1))$
1	159
2	158.23
64	153.46
1024	149.47

Table 4.1: Brute-Force estimates for a SAP in a semigroup of size $s = 2^{160}$

expected the effort of a brute-force attack is inversely proportional to the size of $\operatorname{sap}_x y$.

4.2 Time-Memory trade-offs for Semigroup Action Problems

Monico [20] and Maze [17] both mention variants of time-memory trade-offs for special cases of SAPs. Monico describes an attack based on Brent's cycle finding algorithm for the case where S is a group. Maze argues that semigroups with large subgroups could be attacked in the same manner by excluding all elements in $S \setminus S^*$ first and then employing Monico's idea on the remaining **subgroup** S^*.

Here, we will suggest an alternative to these lines of attack. We will adopt Shank's baby-step giant-step algorithm, since it lends itself to the situation at hand in a very

natural way, and develop a probabilistic algorithm.

Algorithm 67
Let X be an S-set and $\alpha \in X$, $\beta \in S\,.\,\alpha$.

i).. Precalculation: A list of giant-steps $\mathcal{A} := \{s_i\,.\,\alpha\}$ where $s_i \in S$ are chosen uniformly at random is calculated and stored in memory.

ii).. For random $u \in S^*$ the element $u\,.\,\beta$ is calculated and compared to the second entries of elements in the list $\mathcal{A}$.

iii).. If the element $u\,.\,\beta$ is equal to some element $s\,.\,\alpha$ in $\mathcal{A}$ then it follows that $u^{-1}s\,.\,\alpha = \beta$ and that $u^{-1}s$ is a solution.

We tacitly assume that the element α is free with respect to the action, therefore the SAP is of maximal hardness. Equivalently we assume that an adversary works in the set of equivalence classes induced by $\sim_\alpha$. Furthermore we mention that not only the elements $s_i\,.\,\alpha$ have to be stored but the according s_i as well.
To analyze the estimated running time of Algorithm 67 we use the following approximation.

Lemma 68 *Let X be a set of size n, and let $A, B \subseteq X$ be uniformly randomly chosen subsets of sizes a and b respectively, where $a + b \leq n$. The probability of A and B being disjoint is upper bounded by $e^{-\frac{ab}{n}}$.*

Proof The exact probability is seen to be given by

$$\Pr[A \cap B = \emptyset] = \left[\binom{n}{a}\binom{n-a}{b}\right]\left[\binom{n}{a}\binom{n}{b}\right]^{-1} = \binom{n-a}{b}\binom{n}{b}^{-1} = \frac{(n-a)!(n-b)!}{(n-a-b)!n!}.$$

We find a good upper bound by assuming that the elements in B are chosen independently of each other and might therefore form a set of size smaller than b, raising the probability that the sets do not intersect.
Each element of B then has a probability of $1 - \frac{a}{n}$ to not be in A. The probability that all elements of B are not in A can then be bounded by

$$\Pr\left[A \cap B = \emptyset\right] = \binom{n-a}{b}\binom{n}{b}^{-1} \leq \frac{(n-a)^b}{n^b} = \left(1 - \frac{a}{n}\right)^b \leq e^{-\frac{ab}{n}}. \tag{4.1}$$

The first inequality follows either by the considerations above or from multiplying by the

denominators and comparing the factors in the resulting products.

$$\begin{aligned} & \binom{n-a}{b}\binom{n}{b}^{-1} \leq \frac{(n-a)^b}{n^b} \\ \Leftrightarrow \quad & \binom{n-a}{b} n^b \leq \binom{n}{b}(n-a)^b \\ \Leftrightarrow \quad & \prod_{i=0}^{b-1}(n-a-i)n \leq \prod_{i=0}^{b-1}(n-i)(n-a) \end{aligned}$$

The last inequality in Equation (4.1) uses the well known inequality $1 - x \leq e^{-x}$ which holds for all x. ∎

Using Lemma 68 we can estimate the memory and computational effort needed for this attack. This lemma applies to our scenario, even though the set $\mathcal{B}$ in question is restricted to a right coset of the unit group, since $\mathcal{A}$ can be assumed to be sufficiently random over S.
The algorithm is successful iff the list $\mathcal{A}$ and the list $\mathcal{B}$ of elements $u_j.\beta$ have a non-empty intersection. We see that

$$\Pr[\mathcal{A} \cap \mathcal{B} \neq \emptyset] = 1 - \Pr[\mathcal{A} \cap \mathcal{B} = \emptyset] \geq 1 - e^{-\frac{ab}{n}}.$$

It follows that the attack succeeds with probability higher than $\frac{1}{2}$ if,

$$\begin{aligned} & \Pr[\mathcal{A} \cap \mathcal{B} \neq \emptyset] \geq \frac{1}{2} \\ \Leftarrow \quad & e^{-\frac{ab}{n}} \leq \frac{1}{2} \\ \Leftrightarrow \quad & \frac{ab}{n} \geq \ln(2) \approx 0.693. \end{aligned}$$

This shows that the complexity of this attack, given that a sufficient amount of units is available, can be estimated by $\mathcal{O}(\sqrt{n})$.
A possible bottleneck for this attack is the number of available units in S since $b \leq |S^*|$. In fact if we assume that S has no units, apart from an identity, this attack is no better than brute-force. If we consider the number b to be known or bounded, we can solve

Equation (4.1) for a prior to applying the last inequality.

$$\begin{aligned}\Pr\left[\mathcal{A}\cap\mathcal{B}=\emptyset\right] \leq \frac{(n-a)^b}{n^b} = \left(1-\frac{a}{n}\right)^b &\leq \frac{1}{2}\\ \ln(1-\frac{a}{n}) &\leq -\frac{\ln(2)}{b}\\ a &\geq \left(1-\sqrt[b]{\frac{1}{2}}\right)n \qquad (4.2)\end{aligned}$$

Example 69
We would like to estimate the effort for the SAP in Example 17 under the assumption that a free element exists. The number of invertible matrices in the semiring $\mathrm{Mat}_{n\times n}(R_6)$ has been calculated in [13, Corollary 4.13]. A generalized permutation matrix has exactly one non-zero entry in every row and column and that entry is a unit, these matrices are clearly units.The following theorem by Kendziorra [13] states that these are the only units in this and many other cases.

Proposition 70 *Let L be a finite irreducible lattice, I a finite index set, and $A \in \mathrm{Mat}_{I\times I}(\mathrm{Res}(L))$. Then A is invertible iff A is a generalized permutation matrix.*

We see that $|\mathrm{Mat}_{20\times 20}(R_6)^*| = 20!$ and in $\mathrm{Mat}_{20\times 20}\times\mathrm{Mat}_{20\times 20}^{op}$ we therefore have $b = 20!^2$ units and $n = 6^{800} \approx 2^{2068}$ elements. With the help of Equation (4.2) we can then find an estimate for a by

$$\log_2(a) \geq \log_2(n) + \log_2(1-2^{-\frac{1}{b}}) = \log_2(6^{800}) + \log_2(1-2^{-\frac{1}{20!^2}}) \approx 2068 - 123.$$

Solving Equation (4.1) for a leads to a similar estimate. It should be noted that this is an improvement over a brute force approach by a factor of 2^{122} since we would expect to test roughly 2^{2067}, i.e. half the elements of S, before finding the solution. Although this is an improvement it comes at the cost of some precalculation and memory and still is nowhere near feasible.

The situation for the SAP in Protocol 62 is of course very different. We work in the semigroup $P_{L,R}$ or depending on the choice of parameters a subset $P_{L,R}^{(d_L,d_R)}$ of limited degree polynomials. It is highly unlikely that the semiring $C[A]$ for some matrix A contains any units especially when we consider Proposition 70 with the low number and highly restricted form of units in $\mathrm{Mat}_{n\times n}(R_6)$. We will therefore modify this approach and arrive at a Pohlig-Hellman like reduction.

4.3 Pohlig-Hellman Reductions

Let $\varepsilon : S \to X$ be a function whose evaluation requires non-trivial computational effort. Then the problem of finding a/the preimage for a given $x \in \mathrm{im}(\varepsilon) \subseteq X$ is a common problem in cryptography. The function ε could be exponentiation of a generator in a finite field or calculating the multiple of a point on an elliptic curve. In these cases $S = \mathbb{Z}_n$ and the problem at hand is the DLP. When considering a hash function or a block cipher as the function ε the problem at hand relates to preimage resistance and resistance against cipher-text-only attacks. Both of which are actually very weak notions of security in their respective settings and we normally require much stronger security properties from hash functions and block ciphers.
We will consider a different viewpoint on the Pohlig-Hellman reductions that generalizes well to the semigroup action setting. Define the function

$$e := \chi_x \circ \varepsilon : S \longrightarrow \{0,1\}$$
$$s \longmapsto \begin{cases} 1 & \text{if } \varepsilon(s) = x \\ 0 & \text{otherwise} \end{cases}$$

that checks whether an element $s \in S$ is a solution. Evaluation of e requires one use of the function ε. We denote the set of solutions by $E_1 := e^{-1}(1) = \{s \in S : e(s) = 1\}$. A distinguishing attack works by finding another function $d : S \to \{0,1\}$ that is computationally less expensive than ε. The function d is called a **distinguisher** for e if it holds that $e(x) = 1 \Rightarrow d(x) = 1$ or in other words $D_1 \supseteq E_1$ where $D_1 := d^{-1}(1)$. Precision is traded for speed in this case and the function d is used to refine the search-space. This can be done in one of two ways, depending on how well the function d is understood:

1. The set D_1 is directly calculated and instead of searching through S one uses this smaller set.

2. Random elements from S are pretested with d, and ε is only applied to elements that are in D_1.

The advantage in the first case is obvious, since the search-space can be reduced. In the second case we can expect a gain if the function d is noticeable less computationally expensive than e and D_1 is not much bigger than E_1.
To find such a function d we revisit the Pohlig-Hellman reductions on DLPs, see Section 2.5. The Pohlig-Hellman attack uses the fact that the multiplication with a divisor $m|n$ leads

to a surjection onto a non-trivial ideal of $\mathbb{Z}_n$.

$$\lambda_m : \mathbb{Z}_n \to m\,\mathbb{Z}_n$$
$$z \mapsto mz$$

Furthermore this map is naturally compatible with the semigroup action by the flow property. For $x \,.\, \alpha = \beta$ we have

$$\lambda_m(x) \,.\, \alpha = (mx) \,.\, \alpha = m \,.\, (x \,.\, \alpha) = m \,.\, \beta.$$

Once a solution $y \in m\mathbb{Z}_n$ such that $y \,.\, \alpha = m \,.\, \beta$ is found, we know that $x \,.\, \alpha = \beta \;\Rightarrow mx \,.\, \alpha = m \,.\, \beta = y \,.\, \alpha$. If α is free with respect to the semigroup action this implies that $mx = y$. Hence, x can only be the solution if $mx = y$ or equivalently $mx \neq y \;\Rightarrow\; x \,.\, \alpha \neq \beta$. Therefore the function

$$d : z \longmapsto \begin{cases} 1 & \text{if } mz = y \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

is a distinguisher for the DLP.
We see that a necessary prerequisite is the existence of non-trivial divisors of n. Therefore it is common practice to use cyclic groups of prime order. One accepts the fact that Pollard's Rho method and similar collision attacks are easier to perform in these groups, since every collision will lead to a solution, to avoid these very powerful Pohlig-Hellman reductions. Furthermore a solution $y \in m\mathbb{Z}_n$ must be found. The hardness of this task depends amongst other things on the size of $m\mathbb{Z}_n$.

In [18] the authors state that they choose to work in congruence simple semirings to avoid a Pohlig-Hellman reduction for their semigroup action (see Example 17). As argued earlier this statement ignores the fact that the additive structure of the semiring is irrelevant for the semigroup action they consider. Assume there exists a non-trivial **semigroup** morphism $f : (\mathcal{M}, \cdot) \to (S, \circ)$ onto a smaller semigroup $(S, \circ)$. Then this morphism could be used for a reduction. Consider the action of $(A, B) \in \mathcal{M} \times \mathcal{M}^{op}$ on an element $X \in \mathcal{M}$ and apply the morphism f to the equation $AMB = (A, B) \,.\, M$

$$f(AMB) = f(A) \circ f(M) \circ f(B) = (f(A), f(B)) \,._S f(M)$$

where $.\,_S$ is the two sided action of $S \times S^{op}$ onto itself. It follows that every solution

$(A, B) \in \mathrm{sap}_M\, AMB$ can be mapped to a solution $(f(A), f(B)) \in \mathrm{sap}_{f(M)}\, f(AMB)$. This allows us to define a distinguisher

$$d : (A, B) \longmapsto \begin{cases} 1 & \text{if } f(A) \circ f(M) \circ f(B) = f(AMB) \\ 0 & \text{otherwise.} \end{cases}$$

We show that the Pohlig-Hellman attack is not exclusive to the semigroup $\mathbb{Z}_n$, but similar reductions can be applied to other semigroups and semigroup actions. Furthermore we will present a type of reduction that does not rely on the existence or knowledge of non-trivial morphisms. The considerations above motivate the following definition.

Definition 71 (Reduction)
Let X be an S-set then a **reduction** is a triple of maps (f, F, G), where $f : S \to T$ and $F, G : X \to Y$ with T a semigroup acting on Y, such that for all $s \in S$ and $x \in X$ it holds that $f(s) \,.\, G(x) = F(s \,.\, x)$.

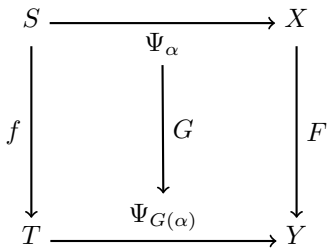

Figure 4.1: Reduction

Example 72

i). Let $(S, \cdot)$ be a semigroup and let $(a, b) \,.\, m := a \cdot m \cdot b$ be the action of $S \times S^{op}$ on S defined by two-sided multiplication. Then for any morphism $f : (S, \cdot) \to (T, \circ)$ the triple of maps $((f \times f), f, f)$ forms a reduction, since

$$f(a \cdot m \cdot b) = \big(f(a), f(b)\big) \,.\, f(m).$$

ii). Let $(\mathbb{Z}_n, \cdot)$ act on a cyclic group G of order n with generator g by exponentiation. Let $d|n$ be a non-trivial divisor of n then the reduction in the classical Pohlig-Hellman attack is given as $(\lambda_d, \Phi_d, \Phi_d)$, see Figure 4.2.

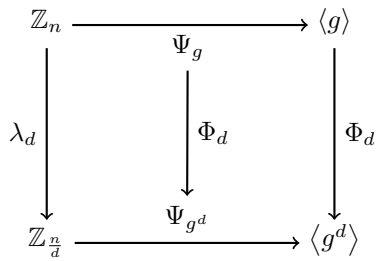

Figure 4.2: Pohlig-Hellman Reduction for the DLP

We call a reduction **effective** if the maps f, F and G can be efficiently computed and it holds that $|S| > |T| > 1$.
Given a reduction, an adversary can project a SAP in S onto a SAP in T. If she is able to solve the SAP in T she can restrict the search in S to preimages of the solutions in T under the map f. Note that this procedure does not rely on f being a morphism.
Given an effective reduction (f, F, G) we can construct a distinguisher

$$d(s) := \begin{cases} 1 & \text{if } f(s) \in \operatorname{sap}_{F(x)} G(y) \\ 0 & \text{otherwise.} \end{cases}$$

Comparing this definition of d with (4.3) we see that a further simplification could be made since in the case of a DLP the set of solutions is singleton. For more general SAPs we have to consider the possibility of $\operatorname{sap}_{F(x)} G(y)$ having several elements and it is not immediately clear whether it holds that for every $t \in \operatorname{sap}_{F(x)} G(y)$ there exists a solution to the original SAP in the preimage of t. The only counterexamples we have found are trivial in the sense that $t \notin \operatorname{im}(f)$ and therefore has no preimages. We formulate the following open problem.

Open Problem 73
Let X an S-set and (f, F, G) a reduction for the corresponding SAP, $t \in \operatorname{im}(f)$. Under what conditions does the following implication hold?

$$t \in \operatorname{sap}_{F(x)} G(y) \;\Rightarrow\; f^{-1}(t) \cap \operatorname{sap}_x y \neq \emptyset$$

We now want to describe a very general class of reductions on general semigroups.

Proposition 74 *Let S be a finite monoid, X an S-set and $m \in S$ then the triple of maps $(\lambda_m, \Phi_m, \mathrm{id})$ forms a reduction. This reduction is effective iff m is a non-unit and*

the monoid operation and action are efficiently computable.

PROOF By the definitions of λ_m and Φ_m it holds for all $\alpha \in X$ and $s \in S$ that

$$\lambda_m(s) \,.\, \alpha = (mx) \,.\, \alpha = m \,.\, (x \,.\, \alpha) = \Phi_m(s \,.\, \alpha).$$

The reduction maps the monoid S onto its right ideal mS, this ideal is a proper ideal iff m is not a unit. ∎

As in the case of a $\mathbb{Z}_n$ action the reductions can be applied recursively and in parallel. A SAP in S can be split into several smaller SAPs in right ideals $n_1S, \ldots, n_kS$. The problem in n_1S can further be reduced to problems in $n_{1,1}n_1S, \ldots, n_{\ell,1}n_1S$.

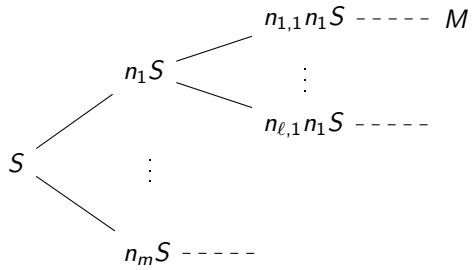

Figure 4.3: Pohlig-Hellman reductions for a semigroup S

The relations in the tree are not necessarily inclusions since mnS is not necessarily a subset of nS. But we assume that the elements n_* are chosen in such a way that $|n_{l,*}n_*S| < |n_*S|$, meaning the right ideals get successively smaller.
When creating such chains it is interesting to look at minimal non-trivial elements, e.g. the minimal right ideal M in Figure 4.3. By the finiteness of S it is obvious that every chain ends in the zero ideal. Consider a minimal non-trivial element M in such a chain, i.e. there exists no element $s \in S$ such that $|sM| < |M|$ unless sM is the zero ideal. Such an ideal M has the property that for all $s \in S$ it holds that either

$$sM = \{0\} \quad \text{or} \quad |sM| = |M|.$$

If we restrict this to the ideal M we see that for every element $x \in M$ it follows that λ_x is either zero or induces a permutation of the elements. The subset $M^* := \{x \in M : \lambda_x \neq 0\}$ of elements that act as permutations on M is in fact a group.

Example 75

1. For $(\mathbb{Z}_n, \cdot)$ the minimal ideals are the multiplicative semigroups of the finite fields $(\mathbb{Z}_p, \cdot)$ for all $p|n$.

2. In $\mathrm{Mat}_{n\times n}(k)$, where k is a field the minimal non-zero right ideals are generated by the matrices $E_{i,j}$ with a one in position i, j and zeros everywhere else. The matrices $E_{i,j}$ and $E_{i',j'}$ generate the same ideal iff $i = i'$ and we therefore have n different such ideals.
   The right ideal $I_i := E_{i,j}\,\mathrm{Mat}_{n\times n}(k)$ is the set of all matrices with zeros outside of the i-th row. Let $A \in I_i$ then λ_A is the zero function if A has a zero at position i, i, otherwise λ_A is a bijection of I_i.

3. We revisit Example 11, the semiring $\mathrm{Mat}_{n\times n}(R)$ of $n \times n$ matrices over a semiring R. For the whole matrix semiring the situation is similar to the previous example. Minimal right ideals are generated by the matrices $rE_{i,j}$ where $r \in R$ generates a non-trivial minimal right ideal in R.
   The semiring R_6 has two non-trivial minimal right ideals $\{0, 2, 5\}$ and $\{0, 3, 4\}$ we therefore have $2n$ non-trivial minimal right ideals in $\mathrm{Mat}_{n\times n}(R_6)$

4. For the commutative semiring $P_{L,R}$ as in Protocol 62 the only non-trivial minimal principal ideal is $\left(\sum_{n=p}^{p+r-1} L^n, \sum_{n=q}^{q+s-1} R^m\right) P_{L,R}$ which has only two elements.

We describe examples for reductions in different semigroups.

Example 76

1. Consider the semigroup $(\mathbb{Z}_n, \cdot)$, its non-units are the elements not coprime to n. Let $\prod_{i=1}^{m} p_i^{e_i}$ for $1 \le i \le m$ and define $d_i := \frac{n}{p_i^{e_i}}$. The reductions applied by the standard Pohlig-Hellman attack on $\mathbb{Z}_n$ actions are displayed in Figure 4.4. The

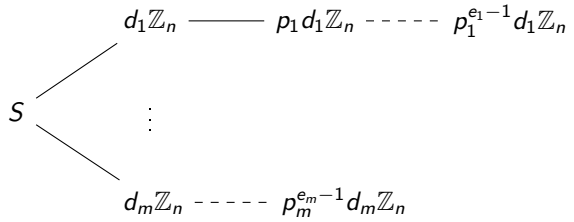

Figure 4.4: Pohlig Hellman reductions for $\mathbb{Z}_n$

   minimal elements $p_i^{e_i-1} d_i\mathbb{Z}_n$ are of size p_i and every $x \in p_i^e d_i\mathbb{Z}_n$ has exactly p_i preimages in $p_i^{e-1} d_i\mathbb{Z}_n$.

2. Let R be a congruence simple semiring and $C := \{0, 1\}$ its center. For fixed integers $p, r \in \mathbb{N}$ we will consider the commutative semiring $C_p^r[x] := {}^{C[x]}\!/\!_{\sim}$ with $\sim$ being the equivalence relation generated by $x^p \sim x^{p+r}$. We note that the semiring $C[L]$ used in Protocol 62 is a surjective image of $C_p^r[x]$ when p and r are the preperiod and period of L.
We describe the elements as sequences of their coefficients, e.g. for $f \in C_p^r[x]$ we write $f = [f_0, \ldots, f_{p-1}; f_p, \ldots, f_{p+r-1}]$. Consider the map λ_x in this semiring

$$\lambda_x([f_0, \ldots, f_{p-1}; f_p, \ldots, f_{p+r-1}]) = [0, f_0, \ldots, f_{p-2}; f_{p-1} + f_{p+r-1}, f_p, \ldots, f_{p+r-2}].$$

This map clearly is not injective if $p > 0$ as the element 1 has no preimage. The same reduction stays effective p times. For the semiring $x^p C_p^r[x]$ the map λ_x is a bijection, since for all $f \in x^p C_p^r[x]$ we have $f = [0, \ldots, 0; f_p, \ldots, f_{p+r-1}]$ and $\lambda_x(f) = [0, \ldots, 0; f_{p+r-1}, f_p, \ldots, f_{p+r-2}]$.
If r is not prime then the chain of ideals can be continued by taking any non-trivial product $r = a \cdot b$ and considering the ideal $x^p \sum_{n=0}^{b-1} x^{an} C_p^r[x]$. As a set this ideal consists of polynomials $f = \sum_{n=0}^{p+r-1} f_n x^n$ with $f_n = 0$ for $n < p$ and $f_n = f_{n+a}$ for $p \leq n < p + r$. The number of variables is reduced to a.
If r is prime we can repeatedly multiply by $1 + x$ to generate subideals. The ideal $(1 + x)^k x^p C_p^r[x]$ contains all polynomials such that $f_n = 0$ for $x < p$ and otherwise ones appear in blocks of length at least k when considering the highest r coefficients in a cyclic manner. That means this ideal is generated additively by the elements $x^{p+\ell}(1 + x)^k = \sum_{n=0}^{k} x^{n+\ell+p}$ for $0 \leq \ell < r$.
The number of such sequences is given by the following lemma, we would like to thank Richard Stanley for pointing out the necessary proposition in his book.

Lemma 77 *Consider the free monoid of sequences generated by* $\{0, 01^k, 01^{k+1}, \ldots\}$. *With the exception of the words* 1^r *for* $r \geq k$, *this monoid contains all binary sequences where ones appear in blocks of length at least* k. *This monoid is very pure (i.e. it has unique circular factorization) in the sense of [31] with generating function*

$$F(x) = x + \frac{x^{k+1}}{1 - x}.$$

By Proposition 4.7.13 in [31] it follows that the generating function for cyclic words

is given by

$$H(x) = \frac{xF'(x)}{1-F(x)} = \frac{x(1-x)^2 + (k+1)x^{k+1} - kx^{k+2}}{(1-x)[(1-x)^2 - x^{k+1}]}$$

We use this lemma to calculate the sizes of the ideals for the case of $C_{34}^{420}[x]$ that can be mapped onto $C[L]$ for L as in Section 3.4.1 for some values of k

k	$\log_2(\lvert I_k \rvert)$
1	420
2	340.78
3	291.58
4	257.27
5	231.61
10	160.58
20	105.72
50	57.21

Table 4.2: Sizes of ideals $I_k := (1+x)^k x^3 4\mathbb{C}_{34}^{420}$

The reductions on $C_p^r[x]$ mentioned in the last example have further consequences for the security of Protocol 62. The probability for the sum of two coefficients $f_i + f_j$ is skewed. Assume that $\Pr[f_i = 0] = q$ for all indices i, it then follows that $\Pr[f_i + f_j = 1] = 1 - q^2$ for $i \neq j$, since we work in the semiring C (isomorphic to $R_{2,b}$ see Example 11). This can be used to solve the SAP in the reduction faster, since not all elements are equally likely as solutions anymore.
Finding preimages under the map λ_x is done by the following considerations.

$$\lambda_x^{-1}(f) = \{[f_1, f_2, \ldots, 0; f_{p+1}, \ldots, f_{p+r-1}, 0]\} \qquad \text{if } f_p = 0$$

$$\begin{aligned}\lambda_x^{-1}(f) = \{&[f_1, f_2, \ldots, 1; f_{p+1}, \ldots, f_{p+r-1}, 0],\\ &[f_1, f_2, \ldots, 0; f_{p+1}, \ldots, f_{p+r-1}, 1],\\ &[f_1, f_2, \ldots, 1; f_{p+1}, \ldots, f_{p+r-1}, 1]\} \qquad \text{if } f_p = 1\end{aligned}$$

Similar observations hold for the reduction by the map λ_{1+x}. In this case the number of preimages in $\lambda_{1+x}^{-1}(f)$ depends on the element f.
Assume $f \in (1+x)^k x^p C_p^r[x]$, then $\lambda_{1+x}^{-1}(f)$ is unique if the sequence $[f_p, \ldots, f_{p+r-1}]$ seen as a cyclic sequence has no block of ones of length $2n$ or more. If a block of $2n$

ones is present there are 2 possibilities for the corresponding coefficients in the preimage; either a block of length $2n-1$ ones followed by a zero or two blocks of each $n-1$ ones and a zero.
In general it holds that if $[\dots, f_i, f_{i+1}, \dots] = [\dots, 0, 1, \dots]$ then $g = [\dots, g_i, g_{i+1}, \dots] = [\dots, 0, 1, \dots]$ and if $[\dots, f_i, f_{i+1}, \dots] = [\dots, 1, 0, \dots]$ then $g = [\dots, g_i, g_{i+1}, \dots] = [\dots, 0, 0, \dots]$ for all $g \in \lambda_{1+x}^{-1}(f)$.

Remark 78
Let M be a semigroup that is a minimal right ideal. Any M action on a set X is not susceptible to the reductions by non-units introduced in this chapter. The use of these semigroups for cryptographic applications might be very restricted though unless the subgroup M^* is very big, enabling an adversary to perform the earlier discussed collision attacks.
If the action is to be used in a Diffie-Hellman like protocol, see Protocol 54, then M is commutative. It follows that $M^* = M \setminus \{0\}$, which means that M consists of a subgroup of size $|M| - 1$ and a zero element.
In non-commutative protocols like Protocol 36 the choices of elements will have to be restricted to the subgroup M^* to avoid generating the zero element as a result of the multiplications involved in the protocol.
In both cases an adversary can therefore work in the subgroup M^*.

Example 79
A structure that seems to resist reductions without being minimal is given by the lattice $L(\ell)$ of subspaces of a line $\ell = \{1, \dots, n\}$ consisting of n points. Consider the commutative monoid $(L(\ell), \vee)$ where $\vee$ is the join operation on the lattice, i.e.

$$\begin{aligned} &\ell \vee x = \ell \\ &x \vee y = \begin{cases} x & \text{if } x = y \\ \ell & \text{otherwise} \end{cases} \\ &\emptyset \vee x = x \end{aligned}$$

for all $x, y \in \ell$. This monoid has $n+2$ elements and apart from a neutral element no other units. The non-trivial minimal ideals in this case are the submonoids $\{\{x\}, \ell\}$ for $x \in \ell$. We see that every reduction $(\lambda_x, \Phi_x, \mathrm{id})$ immediately leads to one of these minimal ideals and every chain has just length two. Furthermore the preimages under the map λ_x

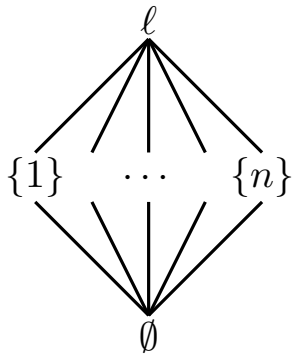

Figure 4.5: Lattice of subspaces of a line ℓ

are of sizes 2 and n respectively. Distinguishers based on these reductions perform very poor since the set D_1 will in almost all cases contain n of the $n+2$ elements and barely help reduce the search-space.
But similar considerations as in Remark 78 hold for this case. The join of two independently chosen elements in $L(\ell)$ will with probability $\frac{n^2+n+3}{(n+2)^2}$ result in the element ℓ.

Conclusion In this chapter we have shown that reductions are a serious threat to the security of a cryptosystem based on a semigroup action for many instances. Furthermore we reason that semigroups that resist these reductions are very close to groups and therefore susceptible to collision attacks.
For the action suggested in [18] we have not only presented a chain of reductions but also provided methods of calculating preimages. Together with the skewed probability for coefficients in the smaller semigroups these considerations form a viable means of breaking the proposed semigroup action.

5 Design of a Key-Exchange Protocol

5.1 A Non-Commutative Semigroup Action Key-Exchange Protocol

The cryptanalysis of Protocol 62 by Steinwandt and Suárez Corona motivated the development of our own protocol based on [1]. This protocol is based on semirings and has several advantages over the protocol by Monico, Maze and Rosenthal.

For our protocol we suggest using a semiring S as a base structure. The distributivity of the multiplication naturally induces parametrized morphisms on the additive semigroup.

$$\begin{aligned}\beta : S \times S &\to \mathrm{End}(S)_+ \\ (L, R) &\mapsto \beta_{L,R} := (M \mapsto LMR).\end{aligned}$$

We define $\gamma_1 : S \times S \to S,\ (A, B) \mapsto AB$ and $\gamma_2 : S \times S \to S,\ (A, B) \mapsto BA$. It then follows that

$$\gamma_1(A, \beta_{B,D}(C)) = ABCD = \gamma_2(D, \beta_{A,C}(B))$$

Furthermore let T be a finitely generated additive subsemigroup of $(S, +)$ with generating set $\mathcal{T}$. Then the restriction of $\beta_{L,R} : T \to S$ can be described as a list of its images of the elements in $\mathcal{T}$, i.e. $(\beta_{L,R}(t))_{t\in\mathcal{T}}$.

Protocol 80

0.) Setup:
The parties agree on a semiring S and a set of generators $\mathcal{T}$ for T.

1.) Generation of public/private keys:
Alice and Bob choose their secret keys $s_A = (\ell_A, r_A) \in S \times T$ and $s_B = (\ell_B, r_B) \in T \times S$ where $r_A = \sum_{i=1}^{m} t_i \in T$ and $\ell_B = \sum_{j=1}^{n} t_j \in T$ with $t_i, t_j \in \mathcal{T}$. Part of the secret key is the decomposition of the elements r_A and ℓ_B into the generators of T. The public keys are the morphisms $p_A = \beta_{\ell_A, r_A}$ and $p_B = \beta_{\ell_B, r_B}$.

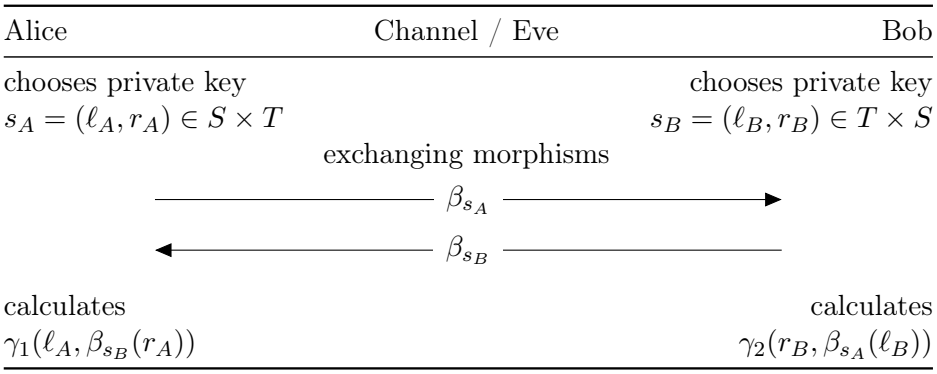

Figure 5.1: Protocol 80

2.) Exchange of public keys:
Alice and Bob transmit their public keys, as the images on the generators in $\mathcal{T}$, i.e. $p_A = (\beta_{\ell_A, r_A}(t))_{t \in \mathcal{T}}$ and $p_B = (\beta_{\ell_B, r_B}(t))_{t \in \mathcal{T}}$.

3.) Calculating the shared key:
Alice uses γ_1 and p_B to calculate

$$k_A = \gamma_1(\ell_A, \sum_{i=1}^{m} \beta_{\ell_B, r_B}(t_i)) = \gamma_1(\ell_A, \beta_{\ell_B, r_B}(\sum_{i=1}^{m} t_i)) = \gamma_1(\ell_A, \beta_{\ell_B, r_B}(r_A)),$$

Bob uses γ_2 and p_A

$$k_B = \gamma_2(r_B, \sum_{j=1}^{n} \beta_{\ell_A, r_A}(t_j)) = \gamma_2(r_B, \beta_{\ell_A, r_A}(\sum_{j=1}^{n} t_j)) = \gamma_2(r_B, \beta_{\ell_A, r_A}(\ell_B))$$

Correctness follows from the definitions, as $k_A = \ell_A \ell_B r_A r_B = k_B$.

Our protocol has several advantages over the previously described Protocol 36.
Since the secret keys contain an element that does not have to be constructed out of the given generators, the size of the key space is not entirely tied to the number of generators and the size of the public keys can be kept small. The public key consists of $|\mathcal{T}|$ elements of S while the key space is of size $|S \times T|$ where as in Protocol 36 using similar notation the key-space is of size $|S|$ for the same size of public keys.
In contrast to Protocol 62 our protocol does not restrict the choice of secret keys to a commuting subset instead the linearity of the action is used. This highly increases the size of the key-space.

The SAP, that an adversary faces in Protocol 80, comes from the semigroup action $\varphi : (S, T^{op}) \times \mathrm{End}_+(T) \to \mathrm{End}_+(T),\ ((s,t), f) \mapsto (x \mapsto s * f(x) * t)$. As shown in Section 3.3 it would not suffice to only find elements $s \in S$ and $t \in T$ that solve the SAP, but a decomposition of t into a sum of the generators in $\mathcal{T}$ would be needed as well.

5.2 Semigroup Semirings

In this section we discuss an extensive class of semirings that could be used for our protocol.

Definition 81 (Semigroup Semiring)
Let R be a semiring (with one and zero) and M a multiplicative semigroup. The **semigroup semiring** $R[M]$ is the set of all maps $f : M \to R$ (for infinite M we require f to be of finite support). It is common to denote the image of $x \in M$ under the map f by f_x. We define addition on these maps componentwise

$$[f + g]_x := f_x + g_x.$$

Multiplication of two maps is defined as the convolution

$$[f * g]_x := \sum_{yz=x} f_y \cdot g_z$$

where we sum over all pairs $(y, z) \in M \times M$ such that $yz = x$. With these operations $(R[M], +, *)$ is a semiring.
A different notation represents elements of $R[M]$ as $R-$linear combinations of the elements of M. An elements $f \in R[M]$ would then be

$$f = \sum_{x \in M} f_x x$$

i.e. the value of f at x is the coefficient for the element x. From this notation the two operations described above naturally follow.
There are embeddings of M and R into $R[M]$ if M is a monoid given by

$$x \longmapsto 1_R \cdot x \quad \text{for } x \in M$$
$$r \longmapsto r \cdot 1_M \quad \text{for } r \in R.$$

We will from now on identify elements in M and R with their images under these embeddings.

Example 82

i) If R is a field and M is a group then the structure of $R[M]$ is studied in representation theory of groups.

ii) The semiring $C_p^r[x]$ from Example 76 is a semigroup semiring with coefficient semiring $C(R_6) \cong R_{2,b}$ and semigroup ${}^{\langle x \rangle}\!/_{(x^p = x^{p+r})}$. Similarly the semiring $C[L]$ for $L \in \mathrm{Mat}_{n\times n}(R_6)$, with the same coefficient semiring C and semigroup $\langle L \rangle$ is a semigroup semiring.

iii) Another example is the group ring $\mathbb{Z}_7[S_5]$ from Section 3.1.

Remark 83

If we use a semigroup semiring $R[M]$ as a base structure in Protocol 80 we see that the corresponding semigroup action problem is equivalent to solving quadratic polynomial equations in R. Given $g, k \in R[M]$ find $f, h \in R[M]$ such that $f * g * h = k$ and $h = \sum_{t\in\mathcal{T}} r_t t$ is an R-linear combination of the generators in $\mathcal{T}$. For fixed $m \in M$ consider the equation $(f * g * h)_m = \sum_{abc=m} f_a g_b h_c = \sum_{abc=m} f_a g_b \sum_{t\in\mathcal{T}} r_t t_c = \sum_{abc=m} \sum_{t\in\mathcal{T}} f_a g_b r_t t_c = k_m$. For every $m \in M$ an adversary gains another quadratic equation in the variables f_a and r_t.

5.3 Reductions of Semigroup Semirings

The semigroup action of Protocol 80 is defined by two-sided multiplication and we will therefore have to consider reductions induced by morphism as in Example 72 as well as reductions by non-units.
In the following we will consider different reductions that could be applied. First we want to exclude certain classes of morphism that are common in semigroup semirings.

Lemma 84 *Let $a : M \to N$ be a morphism of semigroups. Then there exists a unique morphism $A : R[M] \to R[N]$ such that $A(m) = a(m)$ and $A(r) = r$.*

Proof Define $A : R[M] \to R[N]$ by

$$A(f)_n := \sum_{\substack{x\in M \\ a(x)=n}} f_x$$

for all $f \in R[M]$, then it follows that

$$A(f+g)_n = \sum_{\substack{x \in M \\ a(x)=n}} (f+g)_x = \sum_{\substack{x \in M \\ a(x)=n}} (f_x + g_x) = A(f)_n + A(g)_n.$$

And for the convolution we see that

$$\begin{aligned} A(f*g)_n &= \sum_{\substack{x \in M \\ a(x)=n}} \sum_{st=x} f_s g_t = \sum_{\substack{s,t \in M \\ a(st)=n}} f_s g_t = \sum_{\substack{u,v \in N \\ uv=n}} \left(\sum_{\substack{s \\ a(s)=u}} f_s \right) \left(\sum_{\substack{t \\ a(t)=v}} g_t \right) \\ &= (A(f) * A(g))_n \end{aligned}$$

Uniqueness follows from the morphism properties. ∎

It follows that the semigroup M should be congruence-simple to avoid having morphisms other than the trivial ones. While the identity morphism on M clearly induces the identity on $R[M]$ the zero morphism, that maps M to a single element, does not induce a trivial morphism A. In this case the map A is similar to the augmentation maps in group rings that map elements in $R[M]$ to elements in R by summing up all coefficients, $f \mapsto \sum_{m \in M} f_m$. While in group rings this reduction can be avoided by working in the kernel of this map, the augmentation ideal, we do not have the necessary structure in the general case.
The finite congruence-simple semigroups S with $|S| > 2$ are either finite simple groups, care of Theorem 65, or of the form described in Theorem 66. If M is not a group then it has non-units $m \in M$. These can be interpreted as a multiplicative non-unit in $R[M]$ by the embedding described in Definition 81. The map λ_m then induces a reduction as described in Proposition 74. It should be noted that even if m is an absorbing element in M, its embedding in $R[M]$ is not. In fact, if $|mM| = 1$ then the map λ_m is similar to the augmentation map introduced earlier, in the sense that for all images $\lambda_m(f)$ the only non-zero coefficient is the sum of all coefficients of f.
Summing up, we see that M should be a finite simple group to avoid most reductions mentioned above.

A similar extension of morphisms of the coefficient semiring exists.

Lemma 85 *Let $b : R \to S$ be a morphism of semirings. Then there exists a unique morphism $B : R[M] \to S[M]$ such that $B(m) = m$ and $B(r) = b(r)$.*

To avoid a reduction by a morphism B the semiring R must be congruence-simple. If

$R \setminus \{0\}$ has multiplicative non-units $r \in R$ then their embedding into $R[M]$ gives rise to multiplicative non-units in $R[M]$ by the embedding described in Definition 81. Even if M is not a monoid the element $r \cdot m$ for any $m \in M$ is a multiplicative non-unit in $R[M]$. By Proposition 74 this gives rise to a reduction. A semiring without multiplicative non-units apart from the zero element is called a semifield and it has been shown that in the finite case apart from finite fields the only such structure up to isomorphism is the boolean semifield $R_{2,b}$ presented in Example 11, see [8].
We conclude that R should be a finite semifield.

From now on, let G be a finite simple group and F be a finite semifield. A class of non-units in $F[G]$ that can be used for reductions is linked to subgroups of G.

Lemma 86 *Let R be a semiring with zero and one, G a group and $H < G$ a non-trivial subgroup of G then the element $\delta^{(H)}$, the indicator function for the set H, is a multiplicative non-unit in $R[G]$. Furthermore for all $f \in R[G]$ it holds that $\delta^{(H)} * f$ is invariant on right cosets of H.*

Proof For every $h \in H$ it holds that $(\delta^{(H)} * h)_g = \sum_{g' \in G} \delta^{(H)}_{gg'} h_{g'} = \delta^{(H)}_{gh} = \delta^{(H)}$. And for all $f \in R[G]$, $h \in H$ and $g \in G$ it holds that $(\delta^{(H)} * f)_{hg} = \sum_{g' \in G} \delta^{(H)}_{hgg'} f_{g'^{-1}} = \sum_{g' \in G} \delta^{(H)}_{gg'} f_{g'^{-1}} = (\delta^{(H)} * f)_g$. ∎

It follows that elements in the right ideal $\delta^{(H)}$ only have $\frac{|G|}{|H|}$ free variables. To avoid these reductions we would have to choose a group G without any non-trivial subgroups. This limits the possible choices to the cyclic groups of prime order $(\mathbb{Z}_p, +)$.

Remark 87
The semigroup semiring $R_{2,b}[(\mathbb{Z}_p, +)]$ is isomorphic to the semiring C_0^p from Example 76 and the same reductions of course apply.

This leaves the group rings $\mathbb{F}_q[(\mathbb{Z}_p, +)]$ as possible candidates. However, in this case we can simplify the semigroup action problem. Instead of looking for a pair $f, h \in \mathbb{F}_q[(\mathbb{Z}_p, +)]$ such that $f * g * h = (f * h) * g = k$ for given $g, k \in \mathbb{F}_q[(\mathbb{Z}_p, +)]$ we just solve the equation $j * g = k$ for a $j \in \mathbb{F}_q[(\mathbb{Z}_p, +)]$. This is equivalent to solving a set of simultaneous linear equations in $\mathbb{F}_q$ by similar considerations as in Remark 83.

Conclusion We developed a key-exchange protocol that uses non-commutative semirings as a base structure. We introduced an extensive class of such rings and analyzed different

parameter choices that would prevent an attacker from using the reductions introduced in the previous chapter. We finally could show that for no choice of parameters the introduced protocol could be secure under these conditions.

Bibliography

[1] Iris Anshel, Michael Anshel, and Dorian Goldfeld. "An Algebraic Method For Public-Key Cryptography". In: *Mathematical Research Letters* 6 (1999), pp. 287–291.

[2] Matan Banin and Boaz Tsaban. "A reduction of semigroup DLP to classic DLP." In: *IACR Cryptology ePrint Archive* (2013), p. 707. URL: `http://eprint.iacr.org/2013/707`.

[3] Daniel J. Bernstein and Tanja Lange. *SafeCurves: choosing safe curves for elliptic-curve cryptography*. URL: `http://safecurves.cr.yp.to/` (visited on 08/11/2014).

[4] Patrick Dehornoy. "Braid-based cryptography". In: *Group Theory, Statistics, and Cryptography*. Ed. by Alexei G. Myasnikov and Vladimir Shpilrain. Vol. 360. American Mathematical Society, 2004, pp. 5–33.

[5] W. Diffie and M.E. Hellman. "New directions in cryptography". In: *Information Theory, IEEE Transactions on* 22.6 (Nov. 1976), pp. 644–654. ISSN: 0018-9448.

[6] Taher ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". In: *IEEE Transactions on Information Theory*. Lecture Notes in Computer Science 31.4 (1985). Ed. by G R Blakley and David Chaum, pp. 469–472. ISSN: 0018-9448.

[7] Bernard Harris. "Probability Distributions Related to Random Mappings". In: *The Annals of Mathematical Statistics* 31.4 (Dec. 1960), pp. 1045–1062.

[8] U. Hebisch and H.J. Weinert. *Semirings: Algebraic Theory and Applications in Computer Science*. Series in algebra. World Scientific, 1998. ISBN: 9789810236014.

[9] J.M. Howie. *Fundamentals of Semigroup Theory*. LMS monographs. Clarendon Press, 1995. ISBN: 9780198511946.

[10] David Kahn. *The codebreakers : the story of secret writing*. New York: Scribner, 1996. ISBN: 0-684-83130-9.

[11] Delaram Kahrobaei, Charalambos Koupparis, and Vladimir Shpilrain. "Public key exchange using matrices over group rings". In: *Groups Complexity Cryptology* 5.1 (2013), pp. 97–115.

[12] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography - Principles and Protocols*. 1. Edition. CRC Press, 2007. ISBN: 978-1-584-88551-1.

[13] Andreas Kendziorra. "Computational Aspects of Finite Simple Semirings". PhD thesis. University College Dublin, May 2012.

[14] T. Kivinen and M. Kojo. *More modular exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)*. May 2003. URL: `http://tools.ietf.org/html/rfc3526`.

[15] KiHyoung Ko et al. "New Public-Key Cryptosystem Using Braid Groups". In: *Advances in Cryptology — CRYPTO 2000*. Ed. by Mihir Bellare. Vol. 1880. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2000, pp. 166–183. ISBN: 978-3-540-67907-3.

[16] Ueli Maurer and Stefan Wolf. "The Diffie–Hellman Protocol". In: *Designs, Codes and Cryptography* 19.2-3 (2000), pp. 147–171. ISSN: 0925-1022.

[17] G Maze. "Algebraic methods for constructing one-way trapdoor functions". PhD thesis. University of Notre Dame, 2003.

[18] Gérard Maze, Chris Monico, and Joachim Rosenthal. "Public Key Cryptography based on Semigroup Actions". In: *Advances in Mathematics of Communications* 1.4 (2007), pp. 489–507.

[19] Ralph C. Merkle. "Secure Communications over Insecure Channels". In: *Commun. ACM* 21.4 (Apr. 1978), pp. 294–299. ISSN: 0001-0782.

[20] Christopher J. Monico. *Semirings and Semigroup Actions in Public-Key Cryptography*. 2002.

[21] Cristopher Moore, Alexander Russell, and Leonard J. Schulman. "The Symmetric Group Defies Strong Fourier Sampling". In: *SIAM J. Comput.* 37.6 (Mar. 2008), pp. 1842–1864. ISSN: 0097-5397.

[22] Cristopher Moore, Alexander Russell, and Leonard J Schulman. "The symmetric group defies strong Fourier sampling". In: *SIAM Journal on Computing* 37.6 (2008), pp. 1842–1864.

[23] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information (Cambridge Series on Information and the Natural Sciences)*. 1st ed. Cambridge University Press, Jan. 1, 2004. ISBN: 0521635039.

[24] Fabien Peticolas. *Electronic version and English translation of "La cryptographie militaire" by Auguste Kerckhoffs*. Jan. 12, 2014. URL: `http://petitcolas.net/fabien/kerckhoffs/`.

[25] S.C. Pohlig and M.E. Hellman. "An improved algorithm for computing logarithms over GF(p) and its cryptographic significance (Corresp.)" In: *Information Theory, IEEE Transactions on* 24.1 (Jan. 1978), pp. 106–110. ISSN: 0018-9448.

[26] J.M. Pollard. "A monte carlo method for factorization". In: *BIT Numerical Mathematics* 15.3 (1975), pp. 331–334. ISSN: 0006-3835.

[27] J.M. Pollard. "Monte Carlo Methods for Index Computation (mod p)". In: *Mathematics of Computation* 32.143 (1978), pages. ISSN: 00255718. URL: `http://www.jstor.org/stable/2006496`.

[28] Daniel Shanks. "Class number, a theory of factorization, and genera". In: *1969 Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Brook, N.Y., 1969)*. Amer. Math. Soc., Providence, R.I., 1971, pp. 415–440.

[29] Peter W. Shor. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer". In: *SIAM journal on computing* (1997), pp. 124–134. arXiv: `quant-ph/9508027v2`.

[30] Vladimir Shpilrain and Alexander Ushakov. "The Conjugacy Search Problem in Public Key Cryptography: Unnecessary and Insufficient". In: *Applicable Algebra in Engineering, Communication and Computing* 17.3-4 (Mar. 2006), pp. 285–289. ISSN: 0938-1279.

[31] R.P. Stanley. *Enumerative Combinatorics*. 2nd. Cambridge studies in advanced mathematics v. 1. Cambridge University Press, 2002. ISBN: 9780521663519.

[32] Rainer Steinwandt and Adriana Suárez Corona. "Cryptanalysis of a 2-party key establishment based on a semigroup action problem". In: *Advances in Mathematics of Communications* 5.1 (2011), pp. 87–92.

[33] John M. Talbot and Dominic J. A. Welsh. *Complexity and cryptography - an introduction.* Cambridge University Press, 2006, pp. I–XII, 1–292. ISBN: 978-0-521-61771-0.

[34] Jens Zumbrägel. "Classification of Finite Congruence-Simple Semirings with Zero". In: *Journal of Algebra and Its Applications* 07.03 (2008), pp. 363–377.

[35] Jens Zumbrägel. "Public-Key Cryptography Based on Simple Semirings". PhD thesis. Universität Zürich, 2008.