Hidden Number Problems

Barak Shani

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy in Mathematics

The University of Auckland

2017

Abstract

The hidden number problem is the problem of recovering an unknown group element (the "hidden number") given evaluations of some function on products of the hidden number with known elements in the group. This problem enjoys a vast variety of applications, and provides cross-fertilisation among different areas of mathematics.

Bit security is a research field in mathematical cryptology that studies leakage of information in cryptographic systems. Of particular interest are public-key cryptosystems, where the study revolves around the information about the private keys that the public keys leak. Ideally no information is leaked, or more precisely extraction of partial information about the secret keys is (polynomially) equivalent to extraction of the entire keys. Accordingly, studies in this field focus on reducing the problem of recovering the private key to the problem of recovering some information about it. This is done by designing algorithms that use the partial information to extract the keys. The hidden number problem was originated to study reductions of this kind.

This thesis studies the hidden number problem in different groups, where the functions are taken to output partial information on the binary representation of the input. A special focus is directed towards the bit security of Diffie–Hellman key exchange. The study presented here provides new results on the hardness of extracting partial information about Diffie–Hellman keys.

Contents

Chapter 1

Introduction

40 years ago, in November 1976, Whitfield Diffie and Martin Hellman published their groundbreaking paper "New Directions in Cryptography", which proposes an approach that allows two people with no prior acquaintance to have private conversations over public communication channels. This approach was the first to show how two parties can share a secret key over insecure channels without long-term preparation, it is known today as "Diffie–Hellman key exchange". This work was the kickstart to public-key cryptography, a field which in its foundations are hard mathematical problems. In particular, the underlying problem in Diffie–Hellman key exchange, called the computational Diffie–Hellman problem, is to compute the shared secret key given the public keys.

A fundamental notion in public-key cryptography is one-way functions, which are operations that their result on a given input is easy to compute, while the opposite direction – given the result and computing the input – is computationally hard. This notion of hardness is at the center of the thesis. Is it possible to learn anything about the input (besides being in the domain of the function and its image under it)? Of particular interest is the binary representation of the input value: is it possible to compute, or predict with good probability, some of its bits? Extensive research on questions of this kind had been taken in the first two decades of public-key cryptography.

Left behind was the Diffie-Hellman key, the shared secret key in Diffie–Hellman key exchange, as no results on the hardness of computing its bits were given. However, 20 years ago, in August 1996, this long-lasting problem saw a first result in the seminal work of Boneh and Venkatesan. Using lattice basis reduction algorithms they show that a relatively small block of most significant bits of the Diffie–Hellman key is as hard to compute as the entire key. While improvements to this result have been given, this is in fact the only known result for the original Diffie–Hellman key exchange scheme, which takes place in the multiplicative group of a finite field. Subsequent results have been given for variants of the Diffie–Hellman key exchange scheme, where the Diffie–Hellman

key is a finite field element, however these results are based on the same lattice approach in the original work, providing the same result on these Diffie–Hellman keys. On the other hand, when the scheme takes place over a different group no results are known, with the exception of the group of elliptic curve points over a quadratic finite field.

Besides of the theoretical appeal of this problem it has practical sides. For example, when the parties perform a key share their shared key is a group element, from which they need to derive a bit string of certain length. It is most natural and practical that they use some string of bits of this element. In order to guarantee that this bit string gives the required security level, it is necessary to have a proof that computing the chosen bit string is not (drastically) easier than computing the entire key. Results of this kind are called bit security results.

The primary topic of the thesis is the hidden number problem, which abstracts details of secondary importance that arise in formalising questions on the bit security of Diffie–Hellman keys. This problem was introduced by Boneh and Venkatesan in the aforementioned work. Its generality makes this problem interesting for other purposes. It received attention from several different areas and enjoys many applications. In particular, it is related to the learning with errors problem, though we do not discuss this relation. The thesis studies variants of this problem, where for an application we focus on proving that bits of Diffie–Hellman keys as are hard to compute as the entire key.

The mathematical principle in the study of this problem is reduction. One reduces the problem of computing the entire value to computing partial information about it. This requires providing a proof that if an algorithm that computes some partial information about the "hidden" value is given, then it can be used to recover the entire value. This shows that computing the partial information is not easier than computing whole of the information. In order to provide such proofs, one can use a vast supply of mathematical tools, for example the lattice structure and lattice basis reductions as mentioned above. Finding the right structures and tools is many times the crux of the reduction. Unfortunately the ideas used in the literature are very limited, which is reflected by the lack of results.

1.1 Summary of Contributions

The main contribution of the study presented here is proofs of the bit security of Diffie–Hellman keys in different groups. Arguably the most interesting open problem in this research field is to provide bit security results for Diffie–Hellman key exchange in the group of elliptic curve points over a finite field of prime size. Elliptic curves are of greatest use in contemporary cryptography, and so studying their security is of main

interest. Proving a bit security result for elliptic curve Diffie–Hellman keys was stated as an open problem nearly 20 years ago. Such a result is presented here. In a similar fashion, the bit security of Diffie–Hellman key exchange in the algebraic torus is studied here, and a first result in this group is given. With a look to the future for the post-quantum era, where contemporary public-key cryptography is insecure, several Diffie–Hellman-like key exchanges have been proposed in the literature. Increasing attention is directed to the supersingular isogeny cryptosystem, proposed by Jao and De Feo, which is based on isogenies between supersingular elliptic curves. Its appeal for the post-quantum era comes from the fact that the ring structure that arises from these isogenies is non-commutative. The first bit security result for supersingular isogeny Diffie–Hellman key exchange is presented here. All these results are achieved by solving the corresponding hidden number problem.

We highlight the contributions of this thesis, followed by a detailed overview:

- Fourier analysis: introducing the multivariate scaling property and providing an alternative proof to the Fourier concentration of single-bit functions.
- Solution to the multivariate hidden number problem with chosen multipliers for all single-bit functions.
- Solution to the elliptic curve hidden number problem given a portion of 5/6-th most significant bits, and application to the bit security of elliptic curve Diffie–Hellman key exchange.
- Improving the bit security result for elliptic curve Diffie–Hellman key exchange for curves defined over extensions fields to all fields of a constant extension degree.
- Solution to the algebraic torus hidden number problem for the torus $\mathbb{T}_2$ given a portion of 2/3-rd most significant bits, and application to the bit security of algebraic torus Diffie–Hellman key exchange.
- Solution to the isogeny hidden number problem given one component of the quadratic field representation, and application to the bit security of supersingular isogeny Diffie–Hellman key exchange.
- Improving the bit security result relying on the decisional Diffie–Hellman assumption to single bits.

Thesis Structure The thesis structure is the following:

- Chapter 2 presents variants of Diffie–Hellman key exchange and provides background on the algebraic structures and the tools that are used throughout the thesis.

- Chapter 3 focuses on results in discrete Fourier analysis. We prove the multivariate scaling property of the Fourier transform, analyse the SFT algorithm and reprove that all single-bit functions are Fourier concentrated.

- Chapter 4 introduces the framework and basic notions in the study of bit security, for which the following chapters are dedicated to.

- Chapters 5–8 take the following structure: a concrete hidden number problem is presented, then a solution to it is given, and finally the solution is applied to obtain bit security results.

- Chapter 5 surveys the solutions to the original hidden number problem and applications to bit security of Diffie–Hellman keys in finite fields.

- Chapter 6 deals with a chosen-multiplier variant of the hidden number problem. We apply the tools from Chapter 3 to solve this variant of the hidden number problem where the operation in the latter is linear.

- Chapter 7 surveys the modular inversion hidden number problem, uses it to solve the elliptic curve and the algebraic torus hidden number problems, and provides bit security results for the corresponding Diffie–Hellman key exchange.

- Chapter 8 introduces and solves the isogeny hidden number problem.

- Chapter 9 studies the bit security of Diffie–Hellman key exchange with respect to the decisional Diffie–Hellman.

- Chapter 10 gives some concluding remarks and future directions.

Overview We now turn to a detailed overview of the main contributions.

Chapter 3 is dedicated to Fourier analysis on finite groups and may be of independent interest. In the past decade there has been a revival in the interest of studying the bit security of Diffie–Hellman keys. Underlying this renewed study is a chosen-multiplier variant of the hidden number problem, and some tools in Fourier analysis that allow to solve this variant. In fact, this variant already admits a solution, given by Håstad and Näslund almost 20 years ago, however this new proof is much simpler, which may be the cause that led to the renewed interest. Our study focuses on concentrated functions, which

are functions that can be approximated, up to any error term, by linear combinations using small number of Fourier coefficients. A result of Morillo and Ràfols is that every single-bit function is concentrated. Our main contribution in this field is a new approach that allows to prove that certain functions are concentrated. In particular, applying this approach on single-bit functions gives a simpler proof to the result of Morillo and Ràfols. A secondary contribution is a generalisation to the scaling property of the Fourier transform, which we call the multivariate scaling property. The original (univariate) scaling property is a special case of our result. The scaling property is of great use to the study of concentrated functions and their applications, as it allows to show that a concentrated function remains concentrated on a composition with affine functions. We give a criterion for functions that their composition on a concentrated function results in a concentrated function: roughly speaking, among all low degree rational functions only affine functions preserve concentration. Applications of these results to the hidden number problem and subsequently applications for Diffie–Hellman related schemes are presented in Chapter 6, along with a review of previous applications. The context of these applications to the bit security of Diffie–Hellman keys however is very limited.

Chapter 7 considers the hidden number problem for elliptic curves and the algebraic tori. Their relatively complex group operations make these structures unsuitable for approaches taken for finite fields. A very useful tool in studying the problem in these groups is a modular inverse variant of the hidden number problem. This variant was introduced by Boneh, Halevi and Howgrave-Graham, in part to study the bit security of elliptic curve Diffie–Hellman keys. They proposed a solution to this problem which uses known lattice basis algorithms that computes roots of small size for (modular) polynomials. However, they could not obtain a strong result for elliptic curve Diffie–Hellman, as these algorithms are highly sensitive to the structure of the polynomials. It is therefore necessary to generate polynomials with structure that better suits this kind of algorithms. We present an approach that produces polynomials of a desired structure, and follow the rigorous approach to this variant of the hidden number problem, that was given by Ling, Shparlinski, Steinfeld and Wang, to provide the first proof that certain bits of the elliptic curve Diffie–Hellman key are as hard to compute as the entire key. This approach carefully uses algorithms for computing short vectors in a lattice. This reduction also gives the first example where the full abstraction in the hidden number problem is too general for the bit-security study of Diffie–Hellman key exchange. We work with a less abstract problem, which still follows from the study of Diffie–Hellman key exchange, that enables us to generate better suited polynomials. We also show that this new approach is applicable for elliptic curves defined over extension fields, this enables us to extend and improve the result on Diffie–Hellman keys in these groups. For the torus, once the

problem is formalised, our result follows almost immediately from the previous work.

Chapter 8 studies the shared key in supersingular isogeny Diffie–Hellman key exchange. We formalise a new problem, which we call the isogeny hidden number problem, and show its relevance to this study. Our reduction uses the modular polynomials for elliptic curves, which are of great relevance to the study of isogenies. To study this problem it is very convenient to work on the isogeny graph of a fixed degree. We show how this cryptosystem allows us to work over elliptic curves of minimal distance on the graph, which is important for our use of the modular polynomial. The supersingular isogeny Diffie–Hellman key is an element of a quadratic finite field, which we represent by two components in the ground field. Our result shows that computing one component of this representation is sufficient for the full recovery of the key, that is we show how to compute the other component. This result can be thought of as computing half of the bits of the shared secret key is as hard as computing all of them. As information theoretically required, our reduction works with the minimum amount of two curves on the graph. Moreover, we generalise our reduction to the case in which the probability that we have obtained the correct component of the key is low. Reductions for Diffie–Hellman keys that hold in the case that some partial information is wrong are very limited, and usually impose some extra assumptions – these are not needed in our case. As an application, parties that use this key exchange scheme can safely use the value in one component in the key derivation, given that it admits the required bit length.

Chapter 9 takes a different approach. Instead of the computational Diffie–Hellman problem, we reduce the decisional Diffie–Hellman problem to the problem of computing bits of Diffie–Hellman keys. This approach is well known and for almost any partial information on the Diffie–Hellman key it allows one to solve the decisional Diffie–Hellman problem with better success probability than the guessing strategy. The study taken here is of making this success probability perfect. Not many studies of this kind have been performed, and the best result, given by Blake, Garefalakis and Shparlinski, shows how to solve the decisional Diffie–Hellman problem with overwhelming probability given a group element that approximates the second most significant bit and a fraction of the most significant bit of the Diffie–Hellman key. This is similar to the case where two most significant bits are given. With careful analysis we improve this result to the case where only one bit is given. Our reduction holds for almost every single bit of the Diffie–Hellman key (except for some inner bits), and holds for all bits if one is given four consecutive bits. We also improve the previous result to the case where one is given a group element that approximates only a fraction of the most significant bit of the Diffie–Hellman key. Finally, we show that the reduction holds also in the case that some of the given information is wrong.

Chapter 2

Background

This chapter reviews basic notions and their properties, and presents the notation. This review is not exhaustive, however it presents all the tools used in this thesis.

Basic Notation As is customary $\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}$ denote the sets of natural, integral, real and complex numbers, respectively. We use $\#S$, or $|S|$, to denote the cardinality of a set S. Vectors are given in row notation and are usually denoted by bold letters. For two vectors $\mathbf{x}, \mathbf{y}$ we write $\mathbf{x} \equiv \mathbf{y} \pmod{p}$ if the equivalence holds for each of their coordinates. The function log is the logarithmic function with base 2.

2.1 Finite Fields

The algebraic structures that appear throughout this work are built on finite fields. We define the notation and review some elementary facts. Finite fields are discussed extensively in the book by Lidl and Niederreiter [60].

Let $\mathbb{F}_q$ be the field with q elements, then q is a prime power $q = p^m$ for a prime p and a positive integer m. When $m = 1$ then $\mathbb{F}_p$ is a *prime field*, otherwise $\mathbb{F}_q$ is a *non-prime field*. We denote the multiplicative group of the field $\mathbb{F}_q$ by $\mathbb{F}_q^*$.

Prime Fields Let p be a prime number and $\mathbb{F}_p$ the field with p elements. We represent $\mathbb{F}_p$ by the set $\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$, with arithmetic modulo p. Its multiplicative group $\mathbb{F}_p^*$ is represented by the set $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$. The *modular norm* of an element $x \in \mathbb{F}_p$ is defined to be $|x| := \min\{x, p-x\}$; this notion can be generalised to any number z and any modulus N by $|z|_N := \min_{k \in \mathbb{Z}}\{|x - kN|\}$.

Extension Fields Let p be a prime number and a m be positive integer, and set $q := p^m$. The field $\mathbb{F}_q$ can be viewed as a vector space over $\mathbb{F}_p$ of dimension m. In this

case $\mathbb{F}_q$ is a finite field-extension of $\mathbb{F}_p$, which we call an *extension field*, m is the *extension degree* and p is the *characteristic* of $\mathbb{F}_q$. We remark that as a vector space $\mathbb{F}_q$ does not have to be taken over the prime field $\mathbb{F}_p$; it can also be viewed as vector space over any of its subfields. When $m > 1$ the non-prime field $\mathbb{F}_q$ has many different representations, and as the field $\mathbb{F}_q$ is unique, they are isomorphic to each other. A standard and common way to represent $\mathbb{F}_q$ is by the quotient ring $\mathbb{F}_p[x]/(f)$, where $\mathbb{F}_p[x]$ is the polynomial ring over $\mathbb{F}_p$, the polynomial f is a degree-m irreducible polynomial over $\mathbb{F}_p$ and (f) is the (principal) ideal generated by f. As a vector space, the *polynomial basis* of a polynomial representation is $\{1, \alpha, \alpha^2, \ldots, \alpha^{m-1}\}$, where α is a root of f. Another representation of $\mathbb{F}_q$ comes from the *normal basis* defined by the set of all conjugates $\{\theta, \theta^p, \theta^{p^2}, \ldots, \theta^{p^{m-1}}\}$ for a suitable element θ (such that this set is linearly independent). Every finite extension of $\mathbb{F}_q$ is algebraic over $\mathbb{F}_q$, so in general one can represent an extension field by adjoining algebraic elements to $\mathbb{F}_q$. Of our particular interest is the field $\mathbb{F}_{q^2}$, which we represent by $\mathbb{F}_{q^2} = \mathbb{F}_q(\theta)$, where

$$\theta^2 + A\theta + B = 0, \quad A, B \in \mathbb{F}_q \quad \text{and} \quad x^2 + Ax + B \text{ is irreducible over } \mathbb{F}_q\,. \tag{2.1}$$

Trace and Norm Let $\mathbb{F} = \mathbb{F}_q$ be a finite field and Let $\mathbb{K} = \mathbb{F}_{q^m}$ be a finite field-extension of $\mathbb{F}$ of degree m. Consider $\mathbb{K}$ as an m-dimensional vector space over $\mathbb{F}$. The Galois conjugates of elements in $\mathbb{K}$ give rise to two important notions. The *norm* $\mathrm{N}_{\mathbb{K}/\mathbb{F}} : \mathbb{K} \to \mathbb{F}$ is the function that multiplies the conjugates:

$$\mathrm{N}_{\mathbb{K}/\mathbb{F}}(\alpha) := \alpha \cdot \alpha^q \cdot \cdots \cdot \alpha^{q^{m-1}}.$$

The *trace* $\mathrm{Tr}_{\mathbb{K}/\mathbb{F}} : \mathbb{K} \to \mathbb{F}$ is the function that sums the conjugates:

$$\mathrm{Tr}_{\mathbb{K}/\mathbb{F}}(\alpha) := \alpha + \alpha^q + \cdots + \alpha^{q^{m-1}}.$$

Notice that in both cases the result is an element of the ground field $\mathbb{F}$. The trace is a linear transformation from $\mathbb{K}$ to $\mathbb{F}$, in fact all the linear transformations from $\mathbb{K}$ to $\mathbb{F}$ can be defined using the trace function. All $\alpha \in \mathbb{K}$ satisfies $\mathrm{Tr}(\alpha^q) = \mathrm{Tr}(\alpha)$. In addition, the trace is transitive in the following sense: if $\mathbb{F} \subseteq \mathbb{K} \subseteq \mathbb{L}$ is a tower of extensions, then $\mathrm{Tr}_{\mathbb{L}/\mathbb{F}}(\alpha) = \mathrm{Tr}_{\mathbb{K}/\mathbb{F}}(\mathrm{Tr}_{\mathbb{L}/\mathbb{K}}(\alpha))$ for every $\alpha \in \mathbb{L}$. When the fields are clear from the context we omit them and write $\mathrm{Tr}(\alpha)$.

Let $\{b_1, \ldots, b_m\}$ be a basis of $\mathbb{F}_{q^m}$, and let $\{\theta_1, \ldots, \theta_m\}$ be its *dual basis*, defined to satisfy

$$\mathrm{Tr}(b_i\theta_j) = \begin{cases} 0 & \text{if } i \neq j, \\ 1 & \text{if } i = j. \end{cases}$$

The linearity of trace allows to represent any $\alpha \in \mathbb{F}_{p^m}$ by $\alpha = \sum_{i=1}^{m} \mathrm{Tr}(\alpha\theta_i)b_i$. Indeed,

$$\sum_{i=1}^{m} \mathrm{Tr}(\alpha\theta_i)b_i = \sum_{i=1}^{m} \mathrm{Tr}\left(\sum_{j=1}^{m} \alpha_j b_j \theta_i\right) b_i = \sum_{j=1}^{m} \alpha_j \sum_{i=1}^{m} \mathrm{Tr}(b_j\theta_i)b_i = \sum_{j=1}^{m} \alpha_j b_j = \alpha\,.$$

2.1.1 Cyclotomic Subgroups and Algebraic Tori

We give a general description of the cyclotomic subgroup and the algebraic torus, and review some of their properties. Background and further details can be found in Galbraith [31, Chapter 6].

The *n-th cyclotomic polynomial* $\Phi_n(x)$ is the product of $(x-z)$ over all primitive n-th roots of unity z (over $\mathbb{C}$). It is well known that $x^n - 1 = \pi_{1\le d\le n}\Phi_d(x)$ for divisors $d \mid n$. The multiplicative group of the field $\mathbb{F}_q$ satisfies $|\mathbb{F}_{q^n}^*| = q^n - 1$. Therefore, $\Phi_n(q)$ divides $|\mathbb{F}_{q^n}^*|$, and so $\mathbb{F}_{q^n}^*$ has a subgroup of order $\Phi_n(q)$. We define the *cyclotomic subgroup* $G_{q,n}$ to be the subgroup of $\mathbb{F}_{q^n}^*$ of order $\Phi_n(q)$. The subgroup $G_{q,n}$ has the property that any of its elements of order greater than n does not lie in any proper subfield of $\mathbb{F}_{q^n}$.

Algebraic Tori Let $\mathbb{A}^n(\mathbb{F}_q)$ denote the n-dimensional affine space over $\mathbb{F}_q$. Consider an $\mathbb{F}_q$-linear bijection $f : \mathbb{A}^n(\mathbb{F}_q) \to \mathbb{F}_{q^n}$. We define the *algebraic torus* $\mathbb{T}_n(\mathbb{F}_q)$ to be all points in $a \in \mathbb{A}^n(\mathbb{F}_q)$ satisfying $\mathrm{N}_{\mathbb{F}_{q^n}/\mathbb{F}_{q^d}}(f(a)) = 1$; that is $\mathbb{T}_n(\mathbb{F}_q)$ is the affine algebraic set

$$V(\{\mathrm{N}_{\mathbb{F}_{q^n}/\mathbb{F}_{q^d}}(f(x_1,\ldots,x_n)) - 1 \mid 1 \le d \le n,\ d \mid n\}) \subseteq \mathbb{A}^n\,.$$

The torus $\mathbb{T}_n(\mathbb{F}_q)$ is irreducible as an algebraic set and of dimension $\varphi(n)$, where φ is Euler's phi function, and it is isomorphic as a group to the cyclotomic subgroup $G_{q,n}$. Therefore, there is a group operation on $\mathbb{T}_n(\mathbb{F}_q)$ inherited from the multiplication in $G_{q,n} \subseteq \mathbb{F}_{q^n}^*$. When the field $\mathbb{F}_q$ is clear in the context we omit it and write $\mathbb{A}^n, \mathbb{T}_n$.

We say that the torus $\mathbb{T}_n$ is *rational* if there is a birational map ρ (or ρ_n) from $\mathbb{T}_n$ to $\mathbb{A}^{\varphi(n)}$; denote its inverse by ψ (or ψ_n). This map is not necessarily one-to-one, as it may not be defined for all points. We say that a point $g \in \mathbb{T}_n$ is *regular* if it can be represented in $\mathbb{A}^{\varphi(n)}$ using the birational equivalence, that is the map is defined on g. If $\mathbb{T}_n(\mathbb{F}_q)$ is rational then $\mathbb{A}^{\varphi(n)}(\mathbb{F}_q)$ is an *explicit rational parametrization* of $\mathbb{T}_n$ which also gives a *compact representation* of $G_{q,n}$. In this case, the group operation on the torus $\mathbb{T}_n$ induces a "partial" group law on $\mathbb{A}^{\varphi(n)}$, denoted by $\star$ or $\star_n$. Sufficiently many points in $\mathbb{T}_n$ are regular, and so one can use the compact representation in $\mathbb{A}^{\varphi(n)}(\mathbb{F}_q)$ and the operation $\star$ to perform arithmetic in $\mathbb{T}_n(\mathbb{F}_q)$ or equivalently in $G_{q,n}$.

The Group $\mathbf{G_{q,2}}$ Let $\mathbb{F}_{q^2} = \mathbb{F}_q(\theta)$ as in (2.1). The conjugate of θ is $\bar{\theta} := \theta^q = -A - \theta$. The order of $G_{q,2} \cong \mathbb{T}_2(\mathbb{F}_q)$ is $q+1$. The subgroup $G_{q,2} \subseteq \mathbb{F}_{q^2}^*$ is the set of all elements $g \in \mathbb{F}_{q^2}^*$ such that $g\bar{g} = g^{q+1} = 1$. It is easy to check that the $q+1$ elements in the set

$$\left\{ \frac{a+\theta}{a+\bar{\theta}} \;\middle|\; a \in \mathbb{F}_q \right\} \cup \{1\}$$

are distinct and of norm 1, and so $G_{q,2}$ can be represented by this set.

Pick $g \in G_{q,2}$, then on one hand $g = \frac{a+\theta}{a+\bar{\theta}}$ for some $a \in \mathbb{F}_q$, and on the other $g = g_1 + g_2\theta$ as it is an element in $\mathbb{F}_{q^2}$. Equating the two, one can determine g_1, g_2 with respect to a. This gives that each element of the form $(a+\theta)/(a+\bar{\theta}) \in G_{q,2}$ corresponds to the element

$$\psi(a) := \left(\frac{a^2 - B}{a^2 - aA + B}, \frac{2a - A}{a^2 - aA + B} \right)$$

of $\mathbb{T}_2$. On the other hand, one can determine a with respect to g_1, g_2, which gives the mapping ρ (excluding the non-regular points ± 1). For $a, b \in \mathbb{A}^1(\mathbb{F}_q)$, the partial group law on $\mathbb{A}^1$, derived from $\frac{a+\theta}{a+\bar{\theta}} \cdot \frac{b+\theta}{b+\bar{\theta}}$, is given by the following operation in $\mathbb{F}_q$:

$$a \star b = \frac{ab - B}{a + b - A}\,.$$

The inverse map is given by $a' = A - a$.

The Group $\mathbf{G_{q,6}}$ Developing the theory for $G_{q,6}$ and $\mathbb{T}_6(\mathbb{F}_q)$ is more complex than the previous case. As the entire theory is not needed here, we highlight the basic facts. It is common to represent $\mathbb{F}_{q^6}$ as a degree 2 field extension of $\mathbb{F}_{q^3}$, since we can use the theory for $G_{q,2}$. That is, let $\mathbb{F}_{q^6} = \mathbb{F}_{q^3}(\theta)$, where $\theta^2 + A\theta + B = 0$ and $A, B \in \mathbb{F}_{q^3}$ such that $x^2 + Ax + B$ is irreducible over $\mathbb{F}_{q^3}$. The order of $G_{q,6} \cong \mathbb{T}_6(\mathbb{F}_q)$ is $q^2 - q + 1$. It follows that we can apply ρ_2 from above on $\mathbb{T}_6$ to have a representation in $\mathbb{A}^1(\mathbb{F}_{q^3}) \cong \mathbb{A}^3(\mathbb{F}_q)$. It is therefore left to find a birational map from the latter to $A^2(\mathbb{F}_q)$. This mapping involves a hypersurface in $\mathbb{A}^3(\mathbb{F}_q)$, which we call U. Then, for some point $P = (x_P, y_P, z_P)$ on U, we define the map $p_U : \mathbb{A}^3(\mathbb{F}_q) \to \mathbb{A}^2(\mathbb{F}_q)$ by

$$p_U(x_Q, y_Q, z_Q) = \left(\frac{y_Q - y_P}{x_Q - x_P}, \frac{z_Q - z_P}{x_Q - x_P} \right).$$

The inverse of this map is defined using some bivariate linear polynomial g and some bivariate quadratic polynomial h such that

$$p_U^{-1}(a_1, a_2) = P + \frac{g(a_1, a_2)}{h(a_1, a_2)}(1, a_1, a_2) = \left(\frac{g(a_1, a_2)}{h(a_1, a_2)} + x_p, \frac{g(a_1, a_2)}{h(a_1, a_2)} a_1 + y_p, \frac{g(a_1, a_2)}{h(a_1, a_2)} a_2 + z_p \right).$$

Finally, the map $\rho : \mathbb{T}_6(\mathbb{F}_q) \to \mathbb{A}^2(\mathbb{F}_q)$ is given by $\rho = p_U \circ \rho_2$, and its inverse $\psi : \mathbb{A}^2(\mathbb{F}_q) \to \mathbb{T}_6(\mathbb{F}_q)$ is given by $\psi = \psi_2 \circ p_U^{-1}$.

Let $a = (a_1, a_2), b = (b_1, b_2) \in \mathbb{A}^2(\mathbb{F}_q)$. To compute the partial group law in $\mathbb{A}^2(\mathbb{F}_q)$ one needs to compute $\psi(a)\psi(b)$ (as elements of $G_{q,6}$) and apply ρ on this product. Equivalently, we can compute $a \star b$ by applying p_U on

$$p_U^{-1}(a) \star_2 p_U^{-1}(b) = \frac{p_U^{-1}(a)p_U^{-1}(b) - B}{p_U^{-1}(a) + p_U^{-1}(b) - A},$$

where the operations in the right-hand side take place in $\mathbb{F}_{q^3}$.

2.2 Lattices

Lattices are structures that appear in different aspects of mathematics and are subject of extensive studies in different areas of mathematics such as geometry of numbers, Lie algebra and sphere packing. Their presence in mathematics is huge, but we only touch a tiny bit of the lattice world. We focus on *Euclidean lattices*, which are discrete subgroups of the Euclidean space. We restrict this review to basic structures and some hard problems.

Euclidean Lattices Consider the Euclidean space $\mathbb{R}^d$, and denote the (Euclidean) norm of a vector $\mathbf{v} \in \mathbb{R}^d$ by $\|\mathbf{v}\|$. We are interested in the structure arising from integral linear combinations of some set of vectors. Let $B = \{\mathbf{b}_1, \ldots, \mathbf{b}_r\}$ be a set of linearly independent vectors in $\mathbb{R}^d$. The set

$$L = L(B) = \left\{ \sum_{i=1}^{r} n_i \mathbf{b}_i \mid n_i \in \mathbb{Z} \right\}$$

is the *lattice generated by* B and B is a *basis for* L. The number r, which is the number of vectors in B, is the *dimension* or *rank* of $L(B)$. If $r = d$, the lattice $L(B)$ is a full-dimension (or full-rank) lattice. Full-rank lattices of $\mathbb{R}^d$ are isomorphic to $\mathbb{Z}^d$. An important notion is the *volume* or *determinant* of $L(B)$, denoted by $Vol(L(B))$. The volume $Vol(L)$ equals to the volume of the d-dimensional parallelepiped spanned by B. This is an invariant of the lattice and independent of the basis B. In particular, for a full-rank lattice the volume equals to the absolute value of the determinant of any lattice basis.

Hard Lattice Problems Since lattices are discrete they have a shortest non-zero vector, which its norm is known as the *first minima* and denoted by $\lambda_1(L)$. That is,

$\lambda_1(L) = \min\{\|u\| \mid 0 \neq u \in L\}$. The problem of finding a non-zero vector $v \in L$ with minimal norm, that is $\|v\| = \lambda_1(L)$ is the *shortest vector problem* (SVP). A more relaxed version is the γ-*shortest vector problem* (γ-SVP), where for a lattice L in $\mathbb{R}^d$ and a real number $\gamma \geq 1$ the problem is to find a non-zero lattice vector $v \in L$ with norm not larger than γ times the norm of the shortest non-zero vector in L. In other words, $\|v\| \leq \gamma \min\{\|u\| \mid 0 \neq u \in L\}$.

A similar problem is the *closest vector problem* (CVP), where for a lattice L and some vector $v \in \mathbb{R}^d$, not necessarily in L, the problem is to find a lattice vector $w \in L$ of minimum distance from v. In other words, $\|w\| \leq \min\{\|u - v\| \mid v \in L\}$. The γ-*closest vector problem* (γ-CVP) is defined similarly. Babai gave the following solution to γ-CVP.

Lemma 2.1 ([5, Theorem 3.1])**.** *Let L be a full rank lattice of dimension s. Given a point $v \in \mathbb{R}^d$, there exists a polynomial-time algorithm that finds a lattice point $w \in L$ such that*

$$\|u - w\| \leq 2^{d/2} \min\{\|u - b\| \mid b \in L\}\,.$$

Some improvements are given by Schnorr [76]. This result, like many others, uses the fundamental LLL algorithm of Lenstra, Lenstra and Lovász [56]. These problems are fundamental problems in lattice cryptography. References to surveys and state-of-the-art algorithms for these problems can be found in [61, Section 1.2].

2.3 Elliptic Curves

Elliptic curves play a main role in contemporary cryptography. The importance of elliptic curve cryptography to public-key cryptography is invaluable. We review basic properties and some of the notions regarding elliptic curve over finite fields. The book by Silverman [85] is a very good source for the theory of elliptic curves.

Consider the following equation

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6\,,$$

with coefficients in $\mathbb{F}_q$ and no singular points. The set of points $(x, y) \in \mathbb{F}_q^2$ that satisfy this equation is the set of elliptic curve points, denoted by $E(\mathbb{F}_q)$ or simply by E.

Basic Properties For fields $\mathbb{F}_q$ of characteristic not equal to 2 or 3, as will be in the following, one can represent an elliptic curve E in a short Weierstrass form

$$y^2 = x^3 + ax + b\,,$$

where $a, b \in \mathbb{F}_q$. An equation of this form is non-singular if and only if the *discriminant* $\Delta := -16(4a^3 + 27b^2)$ is not zero. Equivalently, there are no singular points if the right-hand side in the Weierstrass equation has no repeated roots. A point $P = (x, y) \in \mathbb{F}_p^2$ that satisfies this equation is a point on the curve E. We denote the x-coordinate (resp. y-coordinate) of a given point P by x_P or P_x (resp. y_P or P_y). The set of points on E, together with the *point at infinity* O, is known to be an abelian group. Hasse's theorem states that the number of points $\#E$ on the curve $E(\mathbb{F}_q)$ satisfies

$$|\#E - q - 1| \leq 2\sqrt{q}\,.$$

The (additive) inverse of a point $Q = (x_Q, y_Q)$ is $-Q = (x_Q, -y_Q)$. For an integer n we denote by $[n]P$ the successive n-time addition of a point P, and we let $[-n]P = [n](-P)$. The *m-torsion* subgroup of E, denoted by $E[m]$, is the group of points of order m, i.e. $E[m] := \{P \in E \mid [m]P = O\}$. Addition of points $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$, where $P \neq \pm Q$, is given by the following formula. Let $s = s_{P+Q} = \frac{y_P - y_Q}{x_P - x_Q}$, then

$$(P + Q)_x = s^2 - x_P - x_Q \quad \text{and} \quad (P + Q)_y = -(y_P + s((P + Q)_x - x_P))\,.$$

The *j-invariant* of E is equal to $-1728\frac{(4a)^3}{\Delta}$. It is an invariant of the curve in the sense that it is independent of the curve representation; all isomorphic curves give rise to the same j-invariant. An elliptic curve E is *supersingular* if and only if $E[p^r] = \{O\}$ for every $r \geq 1$; otherwise the curve is *ordinary* and then $E[p^r] \cong \mathbb{Z}/p^r\mathbb{Z}$. The number of isomorphism classes of supersingular elliptic curves over $\overline{\mathbb{F}}_q$ is approximately $p/12$; therefore there are approximately $p/12$ different j-invariants for supersingular curves of characteristic p. Every supersingular curve is isomorphic to a curve over $\mathbb{F}_{p^2}$.

Isogenies Let E_1, E_2 be two elliptic curves defined over $\mathbb{F}_q$. An *isogeny* is a morphism ϕ satisfying $\phi(O_1) = O_2$. We exclude the zero isogeny $[0]P \equiv O$ from our discussion. The degree of an isogeny ϕ is equal to the degree of ϕ as a morphism. If E_1, E_2 obtain a degree-r isogeny we say that E_1 and E_2 are *r-isogeneous*. By Tate's theorem E_1, E_2 are isogenous if and only if $\#E_1 = \#E_2$. If an isogeny ϕ is separable, then $\deg \phi = \#\ker \phi$. One can define an isogeny by its kernel, in the sense that for every subgroup $G \in \mathbb{E}_1$ there is a unique (up to isomorphism) curve E_2 and a separable isogeny $\phi : E_1 \to E_2$ such that $\ker \phi = G$. We write E_1/G for E_2. Thus, the number of distinct degree-r isogenies whose domain is E_1 is equal to the number of distinct subgroups of E_1 of order r. Specifically, for every prime r not equal to the characteristic of $\mathbb{F}_q$, there are $r + 1$ isogenies (over $\overline{\mathbb{F}}_q$) of degree r. In particular, for $r = 2$ there are three distinct subgroups of $E[2](\mathbb{F}_{p^2})$, and so there are three distinct isogenies over $\mathbb{F}_{p^2}$.

The *modular polynomial* is of special interest in the study of isogenies. For background and further details see [25, 9]. The modular polynomial of order r, denoted by $\Phi_r(X, Y) \in \mathbb{C}[X, Y]$, is symmetric, has integral coefficients and is absolutely irreducible ([44]). We would use the following result that relates r-isogenous curves.

Lemma 2.2. *there is an isogeny of degree r from E_1 to E_2 whose kernel is cyclic if and only if $\Phi_r(j(E_1), j(E_2)) = 0$.*

Computing isogenies between elliptic curves is considered to be a hard problem: given two r-isogenous elliptic curves over $\mathbb{F}_q$, there are no known algorithms that compute an isogeny of degree r between the curves in time polynomial in $\log(q)$.

Pairings Let E be an elliptic curve over the field $\mathbb{F}_q$, and let $\mu_m := \{z \in \overline{\mathbb{F}}_q \mid z^m = 1\}$ the set of all m-th roots of unity. The *Weil pairing* is a bilinear map $e_m : E[m] \times E[m] \to \mu_m$. That is

$$e_m(P_1 + P_2, Q) = e_m(P_1, Q)e_m(P_2, Q)\,,$$

$$e_m(P, Q_1 + Q_2) = e_m(P, Q_1)e_m(P, Q_2)\,.$$

The Weil pairing is alternating: $e_m(P, P) = 1$ for all $P \in E[m]$ (in particular $e_m(P, Q) = e_m(Q, P)^{-1}$); and is non-degenerate: if $e_m(P, Q) = 1$ for all $P \in E[m]$ then $Q = O$. Usually, one considers the pairing e_m into a finite extension field $\mathbb{F}_{q^k}$. An elliptic curve is *pairing-friendly* if the embedding degree induced from e_m is "small"; this ensures that one can efficiently compute the map e_m. For further theory and application see Galbraith [31, Section 26].

2.4 Elimination Techniques

Elimination theory addresses the techniques to eliminate variables in a system of multivariate polynomials. Fundamental concepts are the resultant for bivariate polynomials and the Gröbner basis, which gives a generating set for the ideal that generates the system, and this set is relatively convenient for computations. We refer to the book by Cox, Little and O'Shea [26] for further details.

Resultant Let $p, q \in k[x, y]$ be two polynomials over some field k (this is a special case of interest for the general theory of any number of variables). The *resultant* of p and q with respect to y, denoted $\mathrm{Res}(p, q, y)$, is given by the determinant of the Sylvester matrix of p and q as univariate polynomials in y, that is, we consider $p, q \in k(x)[y]$. The resultant $\mathrm{Res}(p, q, y)$ is a univariate polynomial in x, and so it belongs to $k[x]$. The resultant is

useful for intersecting the set of solutions of two polynomials, as a univariate polynomial can efficiently be factorised. Therefore, its degree is of interest. Denote by $\deg_z p$ the degree of a polynomial p in some variable z. If p satisfies $\deg_x p = n_x, \deg_y p = n_y$ and q satisfies $\deg_x q = m_x, \deg_y q = m_y$, then $\deg_x \operatorname{Res}(p, q, y) \leq m_y n_x + n_y m_x$. An important case is when the resultant is identically equal to zero, since then all points satisfy it. We have the following condition.

Lemma 2.3. $\operatorname{Res}(p, q, y) = 0$ *if and only if* f *and* g *have a common factor in* $k[x, y]$ *with positive degree in* x*.*

2.5 Fourier Analysis on Finite Groups

We review basic background on Fourier analysis on finite domains and specify some of the properties of the Fourier transform. For further background, more details and some applications see the book by Terras [90].

The Space $\mathbf{L^2(R)}$ Let $(R, +, \cdot)$ be a finite ring and denote by $G := (R, +)$ the corresponding additive abelian group. We are interested in the set of functions $L^2(R) := \{f : R \to \mathbb{C}\}$. The set $L^2(R)$ is a vector space over $\mathbb{C}$ of dimension $|R| = |G|$, with the usual pointwise addition and scalar multiplication of functions. To see this, consider the set of Kronecker delta functions $\left\{\delta_i(x) : R \to \{0, 1\}\right\}_{i \in R}$ given by

$$\delta_i(j) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

which forms a basis for $L^2(R)$ as any function $f : R \to \mathbb{C}$ can be written as $f(x) = \sum_{i \in R} f(i)\delta_i(x)$.

Let $f, g \in L^2(R)$. Inner product in $L^2(R)$ is given by $\langle f, g \rangle := \frac{1}{|R|} \sum_{x \in R} f(x)\overline{g(x)}$, where $\overline{z}$ denotes the complex conjugate of $z \in \mathbb{C}$. The inner product induces the L^2 norm $\|f\|_2 := \sqrt{\langle f, f \rangle}$. The infinity norm is $\|f\|_\infty := \max_{x \in R} |f(x)|$. Convolution is defined by $(f * g)(x) := \frac{1}{|R|} \sum_{y \in R} f(x - y)g(y)$.

Characters A *character* of G is a group homomorphism taking values in the non-zero complex numbers, namely $\chi : G \to \mathbb{C}^*$ such that $\chi(x+y) = \chi(x)\chi(y)$. Denote n successive additions of an element $x \in G$ by nx. By Lagrange's theorem $|G|\, x = 0_G$, where 0_G is the identity in G. We get $\chi(x)^{|G|} = \chi(|G|\, x) = \chi(0_G) = 1$. It follows that the characters take values in the complex $|G|$-th roots of unity.

Let $\widehat{G}$ be the set of characters of G. The cardinality of $\widehat{G}$, which equals to the number of $|G|$-th roots of unity in $\mathbb{C}^*$, is $|G|$. The set $\widehat{G}$ along with pointwise multiplication $(\chi_1 \cdot \chi_2)(x) = \chi_1(x)\chi_2(x)$ form a group; the inverse χ^{-1}, defined by $\chi^{-1}(x) = \chi(x)^{-1}$, is the complex conjugate $\overline{\chi}$. The groups $(G, +)$ and $(\widehat{G}, \cdot)$ are isomorphic. Therefore, we can index the characters by the elements of G. That is, once an isomorphism $G \to \widehat{G}$ is chosen, denote it by $\alpha \mapsto \chi_\alpha$.

The characters satisfy the following *orthogonality relations*

$$\sum_{x \in G} \chi(x) = \begin{cases} |G| & \text{if } \chi \text{ is the identity in } \widehat{G}, \\ 0 & \text{otherwise;} \end{cases} \qquad \sum_{\chi \in \widehat{G}} \chi(x) = \begin{cases} |G| & \text{if } x = 0, \\ 0 & \text{otherwise.} \end{cases}$$

For the group $G = \mathbb{Z}_{N_1} \times \cdots \times \mathbb{Z}_{N_m}$ and $\alpha, x \in G$, where $\alpha = (\alpha_1, \ldots, \alpha_m)$ and $x = (x_1, \ldots, x_m)$, the character χ_α is given by $\chi_\alpha(x) = \mathrm{e}^{\frac{2\pi i}{N_1}\alpha_1 x_1} \cdots \mathrm{e}^{\frac{2\pi i}{N_m} a_m x_m}$. In particular for $G = \mathbb{Z}_p$ we have $\chi_\alpha(x) = \mathrm{e}^{\frac{2\pi i}{p}\alpha x}$; for $R = \mathbb{Z}_p^m$ with dot product we have $\chi_{\boldsymbol{\alpha}}(\mathbf{x}) = \mathrm{e}^{\frac{2\pi i}{p}\boldsymbol{\alpha}\cdot\mathbf{x}}$. We denote the N-th root of unity $\mathrm{e}^{\frac{2\pi i}{N}}$ by ω_N.

Fourier Transform and Fourier Basis The (discrete) *Fourier transform* over $L^2(R)$ is the function $\mathcal{F} : L^2(R) \to L^2(R)$ such that $\mathcal{F}(f) = \widehat{f}$ where

$$\widehat{f}(\alpha) := \langle f, \chi_\alpha \rangle = \frac{1}{|G|} \sum_{x \in G} f(x)\overline{\chi}_\alpha(x)\,.$$

The inverse Fourier transform over $L^2(R)$ is the function $\mathcal{F}^{-1} : L^2(R) \to L^2(R)$ such that

$$\mathcal{F}^{-1}(g)(x) := \sum_{\alpha \in G} g(\alpha)\chi_\alpha(x)\,.$$

It follows that

$$\mathcal{F}^{-1}(\mathcal{F}(f))(x) = \sum_{\alpha \in G} \widehat{f}(\alpha)\chi_\alpha(x) = \sum_{\alpha \in G} \langle f, \overline{\chi}_\alpha \rangle \chi_\alpha(x) = \langle f, \sum_{\alpha \in G} \overline{\chi}_\alpha \chi_\alpha \rangle(x) = f(x)\,,$$

using $\overline{\chi}\chi \equiv 1$ and the orthogonality relations above. In particular we see that

$$f(x) = \sum_{\alpha \in G} \widehat{f}(\alpha)\chi_\alpha(x)\,,$$

and so the set of characters $\widehat{G}$ is a basis for $L^2(R)$, called the *Fourier basis*.

As opposed to the "local" Kronecker delta, the Fourier coefficient $\widehat{f}(\alpha)$ contains information about the function f at every point. In particular, the magnitude of $\widehat{f}(\alpha)$ measures the correlation of f with χ_α.

Parseval's Identity and Properties of the Fourier Transform Parseval's identity is the following:

$$\frac{1}{|G|}\sum_{x\in G}\left|f(x)\right|^2 = \|f\|_2^2 = \sum_{\alpha\in G}\left|\widehat{f}(\alpha)\right|^2.$$

Let $R = \mathbb{Z}_{N_1}\times\cdots\times\mathbb{Z}_{N_m}$ with componentwise addition and multiplication. Suppose $f, g \in L^2(R)$. The Fourier transform admits the following properties:

- scaling: if $g(x) := f(cx)$ for $c \in R^*$, then $\widehat{g}(\alpha) = \widehat{f}(c^{-1}\alpha)$;
- shifting: if $g(x) := f(c+x)$ for $c \in R$, then $\widehat{g}(\alpha) = \widehat{f}(\alpha)\chi_\alpha(c)$;
- convolution-multiplication duality: $\widehat{f * g}(\alpha) = \widehat{f}(\alpha)\widehat{g}(\alpha)$.

The scaling property is fundamental for some of the results that will be presented later. For completeness we present its proof:
Suppose $g(x) := f(cx)$ for some invertible element $c \in R$. Clearly c is not a zero divisor since if there exists $0 \neq d \in R$ such that $cd = 0$ then $d = c^{-1}cd = c^{-1}0 = 0$. By definition of the Fourier transform

$$\widehat{g}(\alpha) = \frac{1}{|G|}\sum_{x\in G} g(x)\overline{\chi}_\alpha(x) = \frac{1}{|G|}\sum_{x\in G} f(cx)\overline{\chi}_\alpha(x)\,.$$

Notice that the change of variable $x' := cx$ permutes R. Indeed, suppose for contradiction that there exists $y \neq z \in R$ such that $cy = cz$, then $c(y-z) = 0$ so c is a zero divisor. Therefore,

$$\begin{aligned}\widehat{g}(\alpha) &= \frac{1}{|G|}\sum_{x'\in G} f(x')\overline{\chi}_\alpha(c^{-1}x') = \frac{1}{|G|}\sum_{x'\in G} f(x')\mathrm{e}^{-\frac{2\pi i}{N_1}\alpha_1(c_1^{-1}x_1')}\cdots\mathrm{e}^{-\frac{2\pi i}{N_m}\alpha_m(c_m^{-1}x_m')}\\ &= \frac{1}{|G|}\sum_{x'\in G} f(x')\mathrm{e}^{-\frac{2\pi i}{N_1}(\alpha_1 c_1^{-1})x_1'}\cdots\mathrm{e}^{-\frac{2\pi i}{N_m}(\alpha_m c_m^{-1})x_m'} = \frac{1}{|G|}\sum_{x'\in G} f(x')\overline{\chi}_{\alpha c^{-1}}(x')\\ &= \widehat{f}(c^{-1}\alpha)\,.\end{aligned}$$

The shifting property is proved very similarly: suppose $g(x) := f(c+x)$ for some $c \in R$. Then,

$$\widehat{g}(\alpha) = \frac{1}{|G|}\sum_{x\in G} g(x)\overline{\chi}_\alpha(x) = \frac{1}{|G|}\sum_{x\in G} f(c+x)\overline{\chi}_\alpha(x)\,.$$

Notice that the change of variable $x' := c + x$ permutes R. Therefore,

$$\widehat{g}(\alpha) = \frac{1}{|G|}\sum_{x'\in G} f(x')\overline{\chi}_\alpha(x'-c) = \frac{1}{|G|}\sum_{x'\in G} f(x')\overline{\chi}_\alpha(x')\overline{\chi}_\alpha(-c) = \widehat{f}(\alpha)\chi_\alpha(c)\,.$$

2.6 Diffie–Hellman Key Exchange

Public key cryptography emerged from the seminal paper of Diffie and Hellman [28]. They developed a key exchange scheme that allows any two parties to share a secret key over a public communication channel. This section describes the *Diffie–Hellman key exchange* and several of its variants that were developed throughout the years.

The Discrete Logarithm Problem Underlies the security of Diffie–Hellman key exchange is the *discrete logarithm problem*. Let $(G, \cdot)$ be a group and let $g, h \in G$ such that $h = g^a$ for some integer a. The discrete logarithm problem (DLP) is to compute a. In our groups of interest exponentiation is easy to compute, so it is easy to generate instances of DLP.

We begin with a general description of Diffie–Hellman key exchange in an abstract abelian group $(G, \cdot)$. We let $g \in G$ of order n, and use exponentiation to denote successive multiplication of an element by itself, as usual. We let Alice and Bob be the two communicating parties. The values g, G are public information, i.e. known by any third party. The Diffie–Hellman key exchange protocol proceeds as follows:

1. Alice chooses a random integer $a \in [1, n]$, computes $A = g^a$ and sends A to Bob.
2. Bob chooses a random integer $b \in [1, n]$, computes $B = g^b$ and sends B to Alice.
3. Alice computes $B^a = (g^b)^a = g^{ab}$.
4. Bob computes $A^b = (g^a)^b = g^{ab}$.

Both parties obtain the element g^{ab}, called the *Diffie–Hellman key*. It is clear that the hardness of the discrete logarithm problem in G is necessary to make the computation of g^{ab} infeasible for third parties. However, it is not known whether it is a sufficient condition.

Diffie–Hellman Problems Underlying the security of Diffie–Hellman key exchange it the following problem, known as the *computational Diffie–Hellman* (CDH) problem: given a group $(G, \cdot)$, an element $g \in G$ and the values g^a, g^b, compute g^{ab}. DLP is stronger than CDH in the sense that a solution to DLP also gives a solution to CDH. Whether CDH is stronger than DLP is yet undecided. Certain cases require the weaker *decisional Diffie–Hellman* (DDH) problem: given a group $(G, \cdot)$, an element $g \in G$ and a triple (g^a, g^b, g^c), decide whether $g^c = g^{ab}$.

We describe variants of Diffie–Hellman key exchange in some specific groups.

2.6.1 Prime Fields

Diffie–Hellman in $\mathbb{Z}_{\mathbf{p}}^*$ The original proposal by Diffie and Hellman takes place in $\mathbb{Z}_p^*$ where $g \in \mathbb{Z}_p^*$ is taken to be a primitive element. In general there is no need to take g to be a primitive element, but any element for which DLP in $\langle g \rangle$ is supposed to be hard. Alice and Bob agree on a prime p and an element $g \in \mathbb{Z}_p^*$. The Diffie–Hellman key exchange protocol proceeds as follows:

1. Alice chooses a random integer $a \in [0, p-2]$, computes $A = g^a$ and sends A to Bob.
2. Bob chooses a random integer $b \in [0, p-2]$, computes $B = g^b$ and sends B to Alice.
3. Alice computes $B^a = (g^b)^a = g^{ab}$.
4. Bob computes $A^b = (g^a)^b = g^{ab}$.

Diffie–Hellman in $\mathbf{E}(\mathbb{F}_{\mathbf{p}})$ Using elliptic curves for cryptography, known as *elliptic curve cryptography*, was proposed by Miller [65] and Koblitz [52]. The group of elliptic curves points over a prime field can be taken as the underlying group in Diffie–Hellman key exchange. The parties agree on some prime p, an elliptic curve $E(\mathbb{F}_p)$ with some specific representation and a point $Q \in E$ of order n. The protocol proceeds as above.

2.6.2 Extension Fields

Both of the previous schemes naturally extend to the case where the ground field is non-prime. In the former, one takes the multiplicative group $\mathbb{F}_q^*$ of any finite field, and in the latter one defines the curve over an extension field $\mathbb{F}_q$. We describe schemes that make use of this greater generality. The brilliant idea of combining these groups leads to key exchange between three parties.

Tripartite Diffie–Hellman Joux [50] proposed the following scheme that allows three parties Alice, Bob and Charlie to have a *one-round* key share, similar to the case of two parties. The parties agree on some prime p, an integer m, a pairing-friendly elliptic curve $E(\mathbb{F}_{p^m})$, a paring $e_m : E[m] \times E[m] \to \mu_m$ and two independent points $P, Q \in E[m]$ (such that $e_m(P, Q) \neq 1 \in \mathbb{F}_{p^m}^*$). The protocol is similar to the one above where Alice, Bob and Charlie publish respectively

$$(A_P, A_Q) = ([a]P, [a]Q), \quad (B_P, B_Q) = ([b]P, [b]Q), \quad (C_P, C_Q) = ([c]P, [c]Q)$$

where a, b, c, are their secret keys, respectively. They proceed as follows to have the shared key $e_m(P, Q)^{abc} \in \mathbb{F}_{p^m}^*$:

1. Alice computes $e_m(B_P, C_Q)^a = e_m(C_P, B_Q)^a = (e_m(P,Q)^{bc})^a = e_m(P,Q)^{abc}$.

2. Bob computes $e_m(A_P, C_Q)^b = e_m(C_P, A_Q)^b = (e_m(P,Q)^{ac})^b = e_m(P,Q)^{abc}$.

3. Charlie computes $e_m(A_P, B_Q)^c = e_m(B_P, A_Q)^c = (e_m(P,Q)^{ab})^c = e_m(P,Q)^{abc}$.

The following schemes are known as part of *trace-based cryptography* and *torus-based cryptography*. In fact, the underlying group is $G_{q,n} \subseteq \mathbb{F}_{q^n}^*$ and the main idea is to represent elements in this group by some other algebraic notions that give a shorter representation than the $n\log(q)$ bits needed in general. The benefit of working specifically with $G_{p,n}$ is that any element of $G_{p,n}$ of order greater than n does not lie in any proper subfield of $\mathbb{F}_{p^n}$. Thus from the point of view of DLP the group $G_{p,n}$ is the "hardest part" of $\mathbb{F}_{p^n}^*$. This means that even though one works in a subgroup of $\mathbb{F}_{p^n}^*$, the security obtained is against attacks on the full group $\mathbb{F}_{p^n}^*$ (this is not true if one works in $G_{q,n} \subseteq \mathbb{F}_{q^n}^*$ for a non-prime q, as shown by Granger and Vercauteren [43] for $\mathbb{T}_n(\mathbb{F}_q)$). Additional benefits are that one can make arithmetic computations using the short representation directly and that the "compressed" representations allow to transmit elements of $\mathbb{F}_{p^n}^*$ using only $\varphi(n)\log(p)$ bits. Therefore, the factor $n/\varphi(n)$ is of interest, and the most useful n's to consider are elements of the sequence

$$1, \quad 2, \quad 2\cdot 3 = 6, \quad 2\cdot 3\cdot 5 = 30, \quad 2\cdot 3\cdot 5\cdot 7 = 210, \ \ldots$$

LUC and XTR Diffie–Hellman In trace-based cryptography we consider two finite fields $\mathbb{K} \subset \mathbb{L}$, where $\mathbb{L} = \mathbb{F}_{q^n}$ is a degree-n field-extension of $\mathbb{K} = \mathbb{F}_q$, the field trace function $\mathrm{Tr}_{\mathbb{L}/\mathbb{K}} : \mathbb{L} \to \mathbb{K}$ and the group $G_{q,n}$. The parties agree on a field $\mathbb{K} = \mathbb{F}_q$, an extension $\mathbb{L} = \mathbb{F}_{q^n}$ to the field $\mathbb{K}$ and an element $g \in G_{q,n}$. The Diffie–Hellman key exchange protocol proceeds as follows:

1. Alice chooses a random integer $a \in [1, |G_{q,n}|]$, computes $A = \mathrm{Tr}(g^a)$ and sends A to Bob.

2. Bob chooses a random integer $b \in [1, |G_{q,n}|]$, computes $B = \mathrm{Tr}(g^b)$ and sends B to Alice.

3. Alice computes $\mathrm{Tr}((g^b)^a) = \mathrm{Tr}(g^{ab})$.

4. Bob computes $\mathrm{Tr}((g^a)^b) = \mathrm{Tr}(g^{ab})$.

The computation of the function $\mathrm{Tr}(g) \mapsto \mathrm{Tr}(g^x)$, which uses ladder methods, is the crux of these cryptosystems.

Trace-based cryptography was initiated by Smith and Skinner [86], who proposed using Lucas sequences to perform arithmetic operations on groups of size $p+1$ (like $G_{p,2}$) in a cryptosystem called LUC. Efforts to generalise this approach took place in [13] (based on results from [91]) which eventually led to the XTR cryptosystem [57] (which takes place in $G_{p,6}$; see also [58]); another proposal is the cubic field system [38] (which takes place in $G_{p,3}$).

Since LUC and XTR admit optimal compression factors they received the most attention in the literature. We refer to [87, Capter 4] for more information on these cryptosystems and to [31, Chapter 6] for details on the ladder methods that allow to perform computations on the trace.

Algebraic Torus Diffie–Hellman Torus-based cryptography was initiated by Rubin and Silverberg [73] (see also [74, 75]), where they introduce the $\mathbb{T}_n$-cryptosystems. The group $G_{q,n}$ is taken as the underlying group in Diffie–Hellman key exchange, and since it is isomorphic to the algebraic torus $\mathbb{T}_n$ one can represent its (regular) elements in $\mathbb{A}^m$, where $m = \varphi(n)$, if $\mathbb{T}_n$ is rational. The parties agree on a field $\mathbb{F}_q$, an algebraic torus $\mathbb{T}_n(\mathbb{F}_q)$ that has an explicit rational parametrization, and a regular point $g \in \mathbb{A}^m$. The protocol proceeds as above.

We remark that in practice computing g^a, g^b, g^{ab} is not done directly in $\mathbb{A}^m$ using the partial group law, since the latter involves division in $\mathbb{F}_q$ (which is an "expensive" operation), and so for a uniformly random exponent x one needs about $\log(q)$ divisions. Instead, the operations are usually done in $G_{q,n}$, where one uses the birational maps to switch between the representations; see [73, Section 6] for a detailed overview of the protocol in practice.

2.6.3 Supersingular Isogeny Diffie–Hellman

Using isogenies between elliptic curves for public-key cryptography was first considered in the unpublished, though deep, work of Couveignes [24]. First key exchange using isogenies was proposed in the unpublished work of Rostovtsev and Stolbunov [72] and then in the published work of the latter [89]. These proposals considered ordinary elliptic curves, where composition of isogenies is commutative. However, they have been shown to be vulnerable to quantum computers, as this commutative operation can be exploited by a quantum algorithm to break the system in subexponential time [22]. As these schemes lost their cryptographical appeal, we do not elaborate on them; however, our result also holds for them. An alternative approach is to use supersingular elliptic curve, where composition of isogenies is not commutative. A cryptosystem of this kind was proposed by Jao and De Feo [47], and its key exchange, known as *supersingular isogeny*

Diffie–Hellman key exchange is a subject to our research.

The Supersingular Isogeny Problem Given two isogeneous supersingular elliptic curves E_1, E_2 defined over $\mathbb{F}_q$, compute an isogeny $\phi : E_1 \to E_2$.

This problem does not known to admit any solution in time less than exponential. On the other hand, given E and a subgroup $G \subseteq E$, Vélu's formulas can be used to efficiently compute a representation of the curve $E' := E/G$ and the isogeny $\phi : E \to E'$. We remark that a stronger problem is used in the cryptosystem: computing an isogeny of a given degree. However, more information is given as we explain.

Key Exchange Protocol Alice and Bob agree on a prime of the form $p = \ell_A^n \ell_B^m f \pm 1$, where ℓ_A, ℓ_B are small primes such that $\ell_A^n \approx \ell_B^m$ and f is small, a supersingular elliptic curve $E(\mathbb{F}_{p^2})$ with some specific representation and two pairs $P_A, Q_A \in E[\ell_A^n]$ and $P_B, Q_B \in E[\ell_B^m]$ of independent points (the group $\langle P_A, Q_A \rangle$ generated by P_A and Q_A has (full) order ℓ_A^{2n}, and similarly, $|\langle P_B, Q_B \rangle| = \ell_B^{2m}$).

The supersingular isogeny Diffie–Hellman key exchange protocol proceeds as follows:

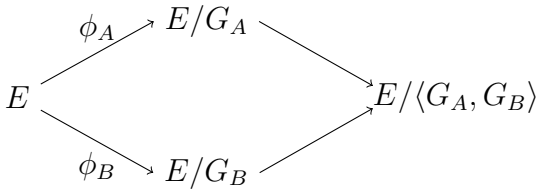
1. Alice chooses random integers $0 \leq a_1, a_2 < \ell_A^n$, not both divisible by ℓ_A, computes $G_A := \langle [a_1]P_A + [a_2]Q_A \rangle$ and an isogeny ϕ_A from E with kernel G_A. She then obtains the curve $E_A = \phi_A(E) = E/G_A$ and the points $\phi_A(P_B)$, $\phi_A(Q_B)$ on it, and sends this triple to Bob.

2. Bob chooses random integers $0 \leq b_1, b_2 < \ell_B^m$, not both divisible by ℓ_B, computes $G_B := \langle [b_1]P_B + [b_2]Q_B \rangle$ and an isogeny ϕ_B from E with kernel G_B. He then obtains the curve $E_B = \phi_B(E) = E/G_B$ and the points $\phi_B(P_A)$, $\phi_A(Q_A)$ on it, and sends this triple to Alice.

3. Alice computes $\phi_B(G_A) = \langle \phi_B([a_1]P_A + [a_2]Q_A) \rangle = \langle [a_1]\phi_B(P_A) + [a_2]\phi_B(Q_A) \rangle$ and an isogeny from E_B with kernel $\phi_B(G_A)$. She then obtains the curve $E_B/\langle \phi_B(G_A) \rangle = E/\langle G_A, G_B \rangle$ and computes its j-invariant.

4. Bob computes $\phi_A(G_B) = \langle \phi_A([b_1]P_B + [b_2]Q_B) \rangle = \langle [b_1]\phi_A(P_B) + [b_2]\phi_A(Q_B) \rangle$ and an isogeny from E_A with kernel $\phi_A(G_B)$. He then obtains the curve $E_A/\langle \phi_A(G_B) \rangle = E/\langle G_A, G_B \rangle$ and computes its j-invariant.

Both parties obtain the curve $E_{AB} := E/\langle G_A, G_B \rangle$. It is not guaranteed that they use the same representation of the curve, so using the j-invariant guarantees they share the same value. The protocol revolves around the following commutative diagram:



Notice that the underlying problem (for an adversary) is to compute a *specific* isogeny from E to E_A (or E_B). Indeed, given $\phi_B(P_A), \phi_B(Q_A)$, the adversary needs to know a_1, a_2 in order to compute $E_B/\langle \phi_B(G_A) \rangle$. These values cannot be obtained in general from any isogeny from E to E_A, see [35, Section 4.1]. In fact, ϕ_A itself is only being used to compute the auxiliary points $\phi_A(P_B), \phi_A(Q_B)$. On the other hand, the auxiliary points give more structure that may be exploited to break the scheme.

Chapter 3

Sets of Large Fourier Transform

The usefulness of representing a function by the Fourier basis comes from the coefficients $\widehat{f}(\alpha)$. As mentioned above the Fourier coefficient $\widehat{f}(\alpha)$ allows to measure how much a function f is correlated with a character χ_α. This chapter is dedicated to the analysis of functions that have a high correlation with one or a few characters. We start with basic definitions and some elementary results, which are an original contribution. Section 3.2 presents a very strong tool that allows to locate those Fourier coefficients with high correlation to some characters. Section 3.3 gives examples of functions with this property and presents a method to prove that a function has (or does not have) a high correlation with some characters; we use this method to reprove that every single-bit function admits this high-correlation property. In section 3.4 we describe a class of functions, of significant interest, with no high correlations to any character. Some of the notions that are presented in this chapter are used in Chapter 6.

3.1 Definitions and Elementary Results

In this section we present some basic notions of functions and their Fourier expansion, and generalise the scaling property of the Fourier transform to the multivariate case. The following definitions consider functions on G, but can be made for functions over rings R where G is their additive group.

Definition 3.1 (τ-heavy coefficient)**.** *Let $f : G \to \mathbb{C}$ be a function, $f = \sum_{\alpha \in G} \widehat{f}(\alpha)\chi_\alpha$, and let $\tau > 0$. The coefficient $\widehat{f}(\alpha)$ is* τ-heavy *if $\left|\widehat{f}(\alpha)\right|^2 \geq \tau$. The set of all τ-heavy Fourier coefficients of f is represented by $Heavy_\tau(f) := \{\alpha \in G \mid \left|\widehat{f}(\alpha)\right|^2 \geq \tau\}$.*

Parseval's identity ensures that the cardinality of $Heavy_\tau(f)$ is at most $\|f\|_2^2/\tau$. Indeed, $\sum_{\alpha \in G} \left|\widehat{f}(\alpha)\right|^2 = \|f\|_2^2 = (\|f\|_2^2/\tau)\tau$. The cases of interest are for sufficiently large τ, such that the amount of τ-heavy coefficients is "small", as we explain below.

Definition 3.2 (k-sparse function)**.** *A function $f : G \to \mathbb{C}$ is* k-sparse *if $\widehat{f}(\alpha)$ is non-zero for at most k elements $\alpha \in G$.*

Definition 3.3 (Restriction)**.** *Given a function $f : G \to \mathbb{C}$ and a set of characters $\Gamma \subseteq \widehat{G}$, the* restriction of f to Γ *is the projection of f onto the subspace* $\operatorname{span}\{\chi_\alpha \in \Gamma\}$*; that is $f|_\Gamma : G \to \mathbb{C}$ is defined by $f|_\Gamma := \sum_{\chi_\alpha \in \Gamma} \widehat{f}(\alpha)\chi_\alpha$.*

Definition 3.4 (ϵ-concentration)**.** *Let $\epsilon > 0$ be a real number. A family of functions $\{f_i : G_i \to \mathbb{C}\}_{i\in\mathbb{N}}$ is* Fourier ϵ-concentrated *if there exists a polynomial P and sets of characters $\Gamma_i \subseteq \widehat{G}_i$ such that $|\Gamma_i| \leq P(\log |G_i|)$ and $\|f_i - f_i|_{\Gamma_i}\|_2^2 \leq \epsilon$ for all $i \in \mathbb{N}$.*

Definition 3.5 (Concentration)**.** *A family of functions $\{f_i : G_i \to \mathbb{C}\}_{i\in\mathbb{N}}$ is* Fourier concentrated *if there exists a polynomial P and sets of characters $\Gamma_i \subseteq \widehat{G}_i$ such that $|\Gamma_i| \leq P(\log |G_i|/\epsilon)$ and $\|f_i - f_i|_{\Gamma_i}\|_2^2 \leq \epsilon$ for all $i \in \mathbb{N}$ and for all $\epsilon > 0$.*

Most applications are concerned with a single function that implicitly defines the entire family. In this case we say that the function, instead of the family, is concentrated or ϵ-concentrated. In this asymptotic case, we omit τ and say that a coefficient $\widehat{f}(\alpha)$ is heavy if $\|f\|_2^2/\tau$ is bounded by some polynomial (in $\log(|G|)$). This implies that a function can only have polynomially many heavy coefficients. The simplest example of a concentrated function is a single character.

The scaling and shifting properties show that if $f, g : R \to \mathbb{C}$ such that $g(x) = (f \circ \varphi)(x)$ for some affine function $\varphi : R \to R$, then we can easily represent the Fourier transform of g using the Fourier transform of f. We give a natural generalisation of this property for the multivariate case.

Lemma 3.6 (Multivariate scaling property)**.** *Let $f : \mathbb{Z}_p \to \mathbb{C}$, let $\boldsymbol{s} = (s_1, \ldots, s_m) \in \mathbb{Z}_p^m$ such that not all $s_i = 0$, and define $f_{\boldsymbol{s}} : \mathbb{Z}_p^m \to \mathbb{C}$ by $f_{\boldsymbol{s}}(\boldsymbol{x}) := f(\boldsymbol{s} \cdot \boldsymbol{x})$. For any $s_k \neq 0$, the Fourier transform of $f_{\boldsymbol{s}}$ satisfies*

$$\widehat{f_{\boldsymbol{s}}}(\mathbf{z}) = \widehat{f_{\boldsymbol{s}}}(z_1, \ldots, z_m) = \begin{cases} \widehat{f}(c) & \text{if } \ (z_1, \ldots, z_m) = (cs_1, \ldots, cs_m), \quad c \in \mathbb{Z}_p\,; \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

Proof. Recall that a character in $\mathbb{Z}_p$ is defined by $\chi_a(x) = \mathrm{e}^{\frac{2\pi i}{p}ax}$ and that for an element $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{Z}_p^m$ the character $\chi_{\mathbf{a}}(\mathbf{x})$ is given by $\chi_{\mathbf{a}}(\mathbf{x}) = \prod_{i=1}^m \chi_{a_i}(x_i)$. Therefore, for $1 \leq k \leq m$, we have

$$\begin{aligned} \chi_{(a_1,\ldots,a_m)}(x_1, \ldots, x_m) &= \prod_{i=1}^m \chi_{a_i}(x_i) = \prod_{i\neq k} \chi_{a_i}(x_i)\, \chi_{a_k}(x_k) \\ &= \chi_{(a_1,\ldots,a_{k-1},a_{k+1},\ldots,a_m)}(x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_m)\chi_{a_k}(x_k)\,. \end{aligned}$$

Assume without loss of generality that $s_m \neq 0$. Then,

$$\begin{aligned}
\widehat{f_{\mathbf{s}}}(\mathbf{z}) &= \frac{1}{p^m} \sum_{(x_1,\ldots,x_m)\in\mathbb{Z}_p^m} f_{\mathbf{s}}(x_1,\ldots,x_m)\,\overline{\chi}_{(z_1,\ldots,z_m)}(x_1,\ldots,x_m) \\
&= \frac{1}{p^m} \sum_{x_1,\ldots,x_m\in\mathbb{Z}_p} f(s_1x_1+\cdots+s_mx_m)\,\overline{\chi}_{(z_1,\ldots,z_m)}(x_1,\ldots,x_m) \\
&= \frac{1}{p^m} \sum_{x_1,\ldots,x_{m-1}} \sum_{x_m} f(s_1x_1+\cdots+s_mx_m)\,\overline{\chi}_{(z_1,\ldots,z_{m-1})}(x_1,\ldots,x_{m-1})\,\overline{\chi}_{z_m}(x_m) \\
&= \frac{1}{p^m} \sum_{x_1,\ldots,x_{m-1}} \overline{\chi}_{(z_1,\ldots,z_{m-1})}(x_1,\ldots,x_{m-1}) \sum_{x_m} f(s_1x_1+\cdots+s_mx_m)\,\overline{\chi}_{z_m}(x_m).
\end{aligned}$$

Since $x'_m := s_m x_m$ is a permutation of $\mathbb{Z}_p$, we change the order of summation and sum over x'_m, and express $\widehat{f_{\mathbf{s}}}(\mathbf{z})$ by

$$\frac{1}{p^m} \sum_{x_1,\ldots,x_{m-1}} \overline{\chi}_{(z_1,\ldots,z_{m-1})}(x_1,\ldots,x_{m-1}) \sum_{x'_m} f\left(s_1x_1+\cdots+s_{m-1}x_{m-1}+x'_m\right)\overline{\chi}_{z_m}(s_m^{-1}x'_m).$$

Let $y := s_1x_1+\cdots+s_{m-1}x_{m-1}+x'_m$, so that $f(s_1x_1+\cdots+s_{m_1}x_{m_1}+x'_m) = f(y)$. We get that

$$\begin{aligned}
\widehat{f_{\mathbf{s}}}(\mathbf{z}) &= \frac{1}{p^{m-1}} \sum_{x_1,\ldots,x_{m-1}} \overline{\chi}_{(z_1,\ldots,z_{m-1})}(x_1,\ldots,x_{m-1})\cdot \\
&\quad \frac{1}{p}\sum_y f(y)\,\overline{\chi}_{z_m}\left(s_m^{-1}(y-s_1x_1-\cdots-s_{m-1}x_{m-1})\right) \\
&= \frac{1}{p^{m-1}} \sum_{x_1,\ldots,x_{m-1}} \overline{\chi}_{(z_1,\ldots,z_{m-1})}(x_1,\ldots,x_{m-1})\cdot \\
&\quad \frac{1}{p}\sum_y f(y)\overline{\chi}_{z_ms_m^{-1}}(y)\,\overline{\chi}_{\left(-z_ms_m^{-1}s_1,\ldots,-z_ms_m^{-1}s_{m-1}\right)}(x_1,\ldots,x_{m-1}) \\
&= \frac{1}{p^{m-1}} \sum_{x_1,\ldots,x_{m-1}} \overline{\chi}_{\left(z_1-z_ms_m^{-1}s_1,\ldots,z_{m-1}-z_ms_m^{-1}s_{m-1}\right)}(x_1,\ldots,x_{m-1})\,\widehat{f}\left(z_ms_m^{-1}\right) \\
&= \widehat{f}\left(z_ms_m^{-1}\right)\frac{1}{p^{m-1}} \sum_{x_1,\ldots,x_{m-1}} \overline{\chi}_{\left(z_1-z_ms_m^{-1}s_1,\ldots,z_{m-1}-z_ms_m^{-1}s_{m-1}\right)}(x_1,\ldots,x_{m-1}).
\end{aligned}$$

The last sum equals 0 unless the character $\chi_{(z_1-z_ms_m^{-1}s_1,\ldots,z_{m-1}-z_ms_m^{-1}s_{m-1})}$ is the trivial character in $\mathbb{Z}_p^{m-1}$, in which case it equals p^{m-1}. Using the orthogonality relations we get that $\widehat{f_{\mathbf{s}}}(z_1,\ldots,z_m) = \widehat{f}(z_ms_m^{-1})$ when $z_j - z_ms_m^{-1}s_j = 0$ for all $1 \leq j \leq m-1$ and otherwise $\widehat{f_{\mathbf{s}}}(z_1,\ldots,z_m) = 0$. In the former case we get that if $z_ms_m^{-1} = c$ for any $c \in Z_p$, then $z_j = cs_j$ for all $1 \leq j \leq m$, as stated in (3.1). □

Note that among the p^m Fourier coefficients of $f_{\mathbf{s}}$, $p^m - p$ of them are zero. More

precisely, the Fourier coefficients of $f_{\mathbf{s}}$ are equal to zero outside the line $(x_1, \ldots, x_m) = (ts_1, \ldots, ts_m)$, where $t \in \mathbb{Z}_p$. Along this line the Fourier coefficients of $f_{\mathbf{s}}$ are (all of) those of f.

The multivariate scaling property shows that for functions $f : \mathbb{Z}_p \to \mathbb{C}$ and $g : \mathbb{Z}_p^m \to \mathbb{C}$ such that $g(\mathbf{x}) = (f \circ \varphi)(\mathbf{x})$ where $\varphi : \mathbb{Z}_p^m \to \mathbb{Z}_p$ is a linear function, one can easily represent the Fourier transform of g using the Fourier transform of f. The generalisation to affine φ is immediate and follows the proof of the shifting property in Chapter 2.

From the (univariate) scaling property it is easy to deduce that $Heavy_\tau(g)$, for $g(x) = f(cx)$, is a permutation of $Heavy_\tau(f)$. A similar result is derived from the multivariate scaling property, as we now show.

Proposition 3.7. *Let $f : \mathbb{Z}_p \to \{-1, 1\}$, let $\boldsymbol{s} = (s_1, \ldots, s_m) \in \mathbb{Z}_p^m$ be such that not all $s_i = 0$, and let $f_{\boldsymbol{s}} : \mathbb{Z}_p^m \to \{-1, 1\}$ be the function $f_{\boldsymbol{s}}(\boldsymbol{x}) := f(\boldsymbol{s} \cdot \boldsymbol{x})$. Then, $Heavy_\tau(f) = \{c_1, \ldots, c_t\}$ if and only if $Heavy_\tau(f_{\boldsymbol{s}}) = \{(c_i s_1, \ldots, c_i s_m) \mid 1 \le i \le t\}$. In other words, a coefficient $\widehat{f_s}(z_1, \ldots, z_m)$ of $f_{\boldsymbol{s}}$ is τ-heavy if and only if there exists $1 \le i \le t$ such that $z_j = c_i s_j$ for every $1 \le j \le m$ and $\widehat{f}(c_i)$ is τ-heavy.*

Proof. Let $1 \le k \le m$ such that $s_k \neq 0$. Assume $c \in Heavy_\tau(f)$ and consider the vector $(z_1, \ldots, z_m) = (cs_1, \ldots, cs_m)$. Specifically $z_k = cs_k$, so $c = z_k s_k^{-1}$ and therefore for every $1 \le j \le m$ one gets $z_j = cs_j = z_k s_k^{-1} s_j$ or $z_j - z_k s_k^{-1} s_j = 0$. From Lemma 3.6 we get that $\widehat{f_s}(cs_1, \ldots, cs_m) = \widehat{f_s}(z_1, \ldots, z_m) = \widehat{f}(z_k s_k^{-1}) = \widehat{f}(c)$. Therefore, we get that $(cs_1, \ldots, cs_m) \in Heavy_\tau(f_{\mathbf{s}})$. That is,

$$|\widehat{f}(c)|^2 > \tau \Longrightarrow |\widehat{f_{\mathbf{s}}}(cs_1, \ldots, cs_m)|^2 > \tau \,.$$

Conversely,

$$\begin{aligned} |\widehat{f_s}(z_1, \ldots, z_m)|^2 > \tau \;&\Longrightarrow \widehat{f_s}(z_1, \ldots, z_m) \neq 0 \\ &\Longrightarrow z_j = z_k s_k^{-1} s_j \text{ for every } 1 \le j \le m \\ &\Longrightarrow z_j = cs_j \text{ for } c = z_k s_k^{-1} \in \mathbb{Z}_p \\ &\Longrightarrow \widehat{f}(c) = \widehat{f}(z_k s_k^{-1}) = \widehat{f_s}(z_1, \ldots, z_m) \\ &\Longrightarrow |\widehat{f}(c)|^2 > \tau \,. \end{aligned}$$

That is, the coefficient $\widehat{f_{\mathbf{s}}}(z_1, \ldots, z_m)$ is τ-heavy if and only if there exists $1 \le i \le t$ such that $z_j = c_i s_j$ for every $1 \le j \le m$ and $\widehat{f}(c_i)$ is τ-heavy. □

Corollary 3.8. *Let f be a function defined over $\mathbb{Z}_p$, let $\boldsymbol{s} = (s_1, \ldots, s_m) \in \mathbb{Z}_p^m$ such that not all $s_i = 0$, and let $f_{\boldsymbol{s}}$ be a function over $\mathbb{Z}_p^m$ defined by $f_{\boldsymbol{s}}(\boldsymbol{x}) := f(\boldsymbol{s} \cdot \boldsymbol{x})$. The function f is concentrated if and only if the function $f_{\boldsymbol{s}}$ is concentrated.*

Proof. Let Γ be a set of characters of $\mathbb{Z}_p$, and define $\Gamma_{\mathbf{s}} := \{\chi_{\mathbf{a}} \mid \mathbf{a} = (as_1, \ldots, as_m), \chi_a \in \Gamma\} \subseteq \mathbb{Z}_p^m$. The proof is evident, since $\sum_{\mathbf{a}\in\Gamma_{\mathbf{s}}} |\widehat{f_{\mathbf{s}}}(\mathbf{a})|^2 = \sum_{a\in\Gamma} |\widehat{f}(a)|^2$. □

3.2 SFT Algorithms

A standard problem is to approximate a function, to a given error term ϵ, by a linear combination of a small number of characters. In the general case one can only expect that $\left|\widehat{f}(\alpha)\right|^2 \approx \|f\|^2/|G|$ so this is not possible. Roughly speaking, an ϵ-concentrated function has Fourier coefficients whose magnitude is large with respect to the function's norm, and a concentrated functions is a function that can be approximated, up to any error term, using linear combinations of a small number of characters (and the corresponding Fourier coefficients). In other words, these are functions that can approximated by k-sparse functions for some polynomially bounded k.

Sparse, or significant, Fourier transform (SFT) algorithms are algorithms that take a function as an input and output a sparse approximation of it; for example, they output a set of significant (heavy) Fourier coefficients of the function. The design of such algorithms is a big area in Learning Theory with applications in computer science and engineering, mostly in signal processing. To date, existing SFT algorithms only need to take a small set $\{f(x_1), \ldots, f(x_n)\}$ of function values. However, they require the ability to ask, and receive, any value $f(x)$. SFT algorithms that take random samples $f(x)$ are not known to exist, and are conjectured to be infeasible.

The Kushilevitz–Mansour algorithm [54] is a cornerstone in this research field, and serves as a basis for most SFT algorithms. This work was extended by Mansour [62]. The algorithm given by Akavia, Goldwasser and Safra [2] is the first to completely consider all finite abelian groups. See [33] for a group-theoretic analysis of these algorithms, which highlights its mathematical ideas, and [37] for a survey on the state-of-the-art algorithms. The following theorem summarises the results from these works.

Theorem 3.9. *Let G be an abelian group represented by a set of generators of known orders. There is a learning algorithm that, given query access to a function $f : G \to \mathbb{C}$, a threshold $\tau > 0$ and $\delta > 0$, outputs a list L of size at most $2\|f\|_2^2/\tau$ such that*

- *L contains all the τ-heavy Fourier coefficients of f with probability at least $1 - \delta$;*
- *L does not contain coefficients that are not $(\tau/2)$-heavy with probability at least $1 - \delta$.*

The algorithm runs in polynomial time in $\log\left(|G|\right)$, $\|f\|_\infty$, $\frac{1}{\tau}$ and $\log\left(\frac{1}{\delta}\right)$.

We give an overview of the algorithm's main procedure. Afterwards we shortly explain each step, specifically we explain the necessity of choosing the values $f(x)$. Starting with $D = G$, the algorithm proceed as follows:

- Partition $D = A \cup B$ (into some well-structured sets), and define the functions $f_A(x) := \sum_{\alpha \in A} \widehat{f}(\alpha)\chi_\alpha(x)$ and $f_B(x) := \sum_{\beta \in B} \widehat{f}(\beta)\chi_\beta(x)$.

- Approximate the values $f_A(x_i)$ and $f_B(y_j)$ for polynomially many samples x_i, y_j, chosen uniformly at random.

- Using the values from the previous step, approximate the norms $||f_A||_2^2$ and $||f_B||_2^2$.

- Using Parseval's identity $||f_A||_2^2 = \sum_{\alpha \in A} |\widehat{f}(\alpha)|^2$, if the approximation of the norm is smaller than[1] $\frac{\tau}{4/3}$ then with overwhelming probability f does not have a τ-heavy coefficient in A. Hence, dismiss A. Act similarly for f_B.

- Run the algorithm recursively on the remaining sets and stop when it reaches singletons.

Let us explain this procedure. For simplicity, we take $G = \mathbb{Z}_N$ where $N = 2^n$ is some power of 2. As the theorem states this is not necessary – the essence of the algorithm is the behaviour of certain exponential sums over G – however it gives the most simple presentation. Moreover, the work in [55] shows that considering groups whose domain is a power of 2 is sufficient to get a complete SFT algorithm for any finite abelian group. We elaborate on this work in Section 3.3.

The algorithm is recursive, and each iteration has several steps. First, the algorithm partitions $\mathbb{Z}_N = A \cup B$ where A contains all the even elements and B contains all the odd elements. We only focus on the set A, as everything is done similarly for B. Its aim is to decide whether potentially there are τ-heavy coefficients $\widehat{f}(\alpha)$ for $\alpha \in A$. More precisely, given the threshold τ, the algorithm determines is there are no $\tau/2$-heavy coefficients $\widehat{f}(\alpha)$ such that $\alpha \in A$.

Step 1. Recall that the function f_A is not given as we only have access to f. In order to get values from f_A we use the convolution-multiplication duality of the Fourier transform: $\widehat{f * h}(\alpha) = \widehat{f}(\alpha)\widehat{h}(\alpha)$. The function h, called a filter function, is taken to satisfy

$$\widehat{h_A}(\alpha) = \begin{cases} 1 & \alpha \in A, \\ 0 & \text{otherwise.} \end{cases}$$

[1] A lower threshold $\frac{3}{4}\tau$ is needed since the algorithm only approximates the norm. As a consequence, the final list may contain coefficients that are $\frac{\tau}{2}$-heavy but not τ-heavy.

We then have

$$\widehat{f * h_A}(\alpha) = \begin{cases} \widehat{f}(\alpha) & \alpha \in A, \\ 0 & \text{otherwise.} \end{cases}$$

In other words,

$$f * h_A = f_A \,.$$

The evaluation of $h(x) = \sum_{\alpha \in A} \chi_\alpha(x)$ can be done easily using the formula for geometric sum since A forms an arithmetic progression. Specifically, for A as above (A contains all the even integers in $\mathbb{Z}_N$ so the arithmetic progression is of difference 2) we get that

$$h_A(x) = \begin{cases} 2^{n-1} & \text{if } x = 0 \text{ or } x = 2^{n-1}, \\ 0 & \text{otherwise.} \end{cases}$$

One immediately sees that evaluating the characters of $\mathbb{Z}_N$ at $x = 2^{n-1}$ gives a distinction between even and odds, as they all collapse to the characters of order 2. Indeed, we get that

$$\widehat{h_A}(\alpha) = \frac{1}{2^n} \sum_x h_A(x) \omega_{2^n}^{\alpha x} = \frac{1}{2} \left(1 + (-1)^\alpha\right) = \begin{cases} 1 & \alpha \in A, \\ 0 & \text{otherwise.} \end{cases}$$

Step 2. Since $f_A = f * h_A$ we have

$$f_A(x) = f * h_A(x) = \frac{1}{|G|} \sum_{y \in G} f(x-y)h(y) = \frac{1}{2^n} \left(f(x)2^{n-1} + f(x - 2^{n-1})2^{n-1}\right).$$

Hence, to evaluate f_A at some point $x \in \mathbb{Z}_N$ we need the values $f(x)$ and $f(x - 2^{n-1})$. This explains why the ability to have chosen values of the original function f is necessary.

Step 3. One approximates the norm $\|f_A\|_2^2$ by taking sufficiently many values $f_A(x_i)$ for uniformly and independently chosen $x_i \in G$. Then $\frac{1}{m} \sum_{i=1}^m \left|f(x_i)\right|^2 \approx \|f_A\|_2^2$.

Step 4. By Parseval's identity $\|f_A\|_2^2 = \sum_{\alpha \in A} \left|\widehat{f}(\alpha)\right|^2$. The approximation from the previous step is used to determine if there may be heavy coefficients $\widehat{f}(\alpha)$ with $\alpha \in A$. More precisely, if the norm approximation is smaller than $3\tau/4$, then with overwhelming probability there are no τ-heavy coefficients $\widehat{f}(\alpha)$ such that $\alpha \in A$.

Step 5. As a result of the previous step, the algorithm decides whether to discard the set A (no τ-heavy coefficients) or to iterate where now A is partitioned into 2 sets that differ by their residue mod 4.

Using Parseval's identity, a simple analysis shows that the number of sets involved in this process is small, that is the algorithm will discard most sets. The analysis is similar to the one above which shows that the number of heavy coefficient is bounded.

3.3 Concentrated Functions

In this section we present some concentrated functions. The main result in this subject is that every single-bit function is concentrated. This was first proved by Morillo and Ràfols [67]. We conclude this section with a method to prove that a family of functions is concentrated. As a corollary, we get a new and simpler proof to the concentration of every single-bit function.

We already mentioned that the simplest example of a concentrated functions is a single character, as it consists of only one non-zero coefficient. We have not given yet an example of an ϵ-concentrated function, which is not concentrated. Adding some 'noise' to a character converts it from a concentrated function into an ϵ-concentrated function.

We define 'noisy characters' $f : \mathbb{Z}_N \to \mathbb{C}$ by $f(x) := \omega_N^{\alpha x + e(x)}$ for some random functions e. It is very easy to see that $\widehat{f}(\alpha)$ is a heavy coefficient of f (see for example [33, Section 6.1]). While in general the other coefficients are non-zero, the randomness of e guarantees that no other coefficient is heavy, so these functions are not concentrated (because, for example, they are not $\epsilon/2$-concentrated).

A concrete example of such noisy character is the function $\mathrm{LWE}_{\mathbf{s}} : \mathbb{Z}_p^n \to \mathbb{Z}_p$, given by $\mathrm{LWE}_{\mathbf{s}}(\mathbf{x}) = \omega_p^{\mathbf{x}\cdot\mathbf{s}+e(\mathbf{x})}$ for $e(\mathbf{x})$ drawn from a Gaussian distribution. Another example is the function $\mathrm{LPN}_{\mathbf{s}} : \{0,1\}^n \to \{0,1\}$, given by $\mathrm{LPN}_{\mathbf{s}}(\mathbf{x}) = (-1)^{\mathbf{x}\cdot\mathbf{s}+e(\mathbf{x})}$ for $e(x) \in \{0,1\}$ which is mostly 0. Let I be the set for which $e(\mathbf{x}) = 1$, then $\widehat{\mathrm{LPN}_{\mathbf{s}}}(\mathbf{s}) = \frac{1}{2^n}\sum_{\mathbf{x}\notin I} 1 + \frac{1}{2^n}\sum_{\mathbf{x}\in I}(-1) = 1 - \frac{2|I|}{2^n}$. The function's norm in 1 and $\frac{2|I|}{2^n}$ is expected to distribute evenly among the other coefficients. Therefore $\mathrm{LPN}_{\mathbf{s}}$ is ϵ-concentrated. These examples show that one does not expect a general learning algorithm where the function values cannot be chosen, as it will solve the LWE and LPN problems.

Definition 3.10 (Boolean predicate)**.** *A* Boolean predicate*, or just a* predicate*, on a set G is a Boolean-valued function $P : G \to \{x, y\}$ for some $x \neq y$.*

Definition 3.11 (Segment predicate [2, Definition 14])**.** *Let $\mathcal{P} = \{P_N : \mathbb{Z}_N \to \{\pm 1\}\}$ be a family of predicates.*

- *A predicate P_N is a* basic t-segment predicate *if $\left|\{x \in \mathbb{Z}_n \mid P_N(x+1) \neq P_N(x)\}\right| \leq t$.*
- *A predicate P_N is a* t-segment predicate *if there exists a basic t-segment predicate P' and $a \in \mathbb{Z}_N^*$ such that $P_N(x) = P'(x/a)$ for every $x \in \mathbb{Z}_N$.*
- *The set $\mathcal{P}$ is a* family of segment predicates *if for every N the predicate P_N is $t(N)$-segment predicate for $t(N) \leq poly(\log(N))$.*

We give a few examples of families of t-segment predicates. Let $2^{n-1} < N \leq 2^n$, we define the i-th bit function $\mathrm{bit}_i : \mathbb{Z}_N \to \{-1, 1\}$ by $\mathrm{bit}_i(x) = (-1)^{x_i}$ where $x = \sum_{j=0}^{n-1} x_j 2^j$ and $x_j \in \{0, 1\}$.

Example 3.12.

- *The most significant bit function* $bit_n : \mathbb{Z}_N \to \{-1, 1\}$*, where* $n = \lfloor \log(N) \rfloor$*, is a basic* 2*-segment predicate. Therefore, the family of most significant bit functions is a family of segment predicates.*

- *More generally, for every* $0 \leq k \leq n$ *the function* $bit_{n-k} : \mathbb{Z}_N \to \{-1, 1\}$*, where* $n = \lfloor \log(N) \rfloor$*, is a basic* 2^{k+1}*-segment predicate. For* $k \leq \log\log(N)$ *we get that the family of functions* bit_{n-k} *is a family of segment predicates, since there are at most* $2\log(N)$ *segments.*

- *The function half* $: \mathbb{Z}_N \to \{-1, 1\}$*, for which* $half(x) = 1$ *if* $0 \leq x < \frac{N}{2}$ *and* $half(x) = -1$ *otherwise, is a basic* 2*-segment predicate.*

- *Let* N *be odd, then the least significant bit function* $bit_0 : \mathbb{Z}_N \to \{-1, 1\}$*, is a* 2*-segment predicate, since* $bit_0(x) = half(x/2)$*.*

- *More generally, for* $0 \leq k \leq \log\log(N)$*, the function* $bit_k : \mathbb{Z}_N \to \{-1, 1\}$ *is a* 2^{k+1} *segment predicate, since the predicate* $P' : \mathbb{Z}_N \to \{-1, 1\}$*, defined by* $P'(x) = bit_k(x/2^{k+1})$*, partition* $\mathbb{Z}_N$ *into at most* 2^{k+1} *segments.*

A fundamental result is that a family of segment predicates is concentrated. In order to prove this result we need the following elementary claims. Recall that we define the modular norm by $|\alpha|_N := \min\{\alpha, N - \alpha\}$ for any $\alpha \in \mathbb{Z}_N$.

Claim 3.13. *Let* N *be a positive integer, let* $\omega_N := \mathrm{e}^{\frac{2\pi i}{N}}$ *and let* $k \in \mathbb{N}$*. Define*

$$S_{\alpha,K} = \sum_{x=0}^{K-1} \omega_N^{\alpha x}.$$

Then for any $\alpha \in \mathbb{Z}_N^*$

$$\left|S_{\alpha,K}\right|^2 \leq \frac{N^2}{|\alpha|_N^2 \left(\pi^2 - \frac{\pi^4}{12}\right)} < \frac{N^2}{|\alpha|_N^2}.$$

Proof. Since $S_{\alpha,K}$ is a sum of a geometric series, we have

$$S_{\alpha,K} = \frac{\omega_N^{\alpha K} - 1}{\omega_N^{\alpha} - 1} = \frac{(\cos(2\pi\alpha K/N) - 1) + i\sin(2\pi\alpha K/N)}{(\cos(2\pi\alpha/N) - 1) + i\sin(2\pi\alpha/N)}.$$

Therefore,

$$\left|S_{\alpha,K}\right|^2 = \frac{1-\cos(2\pi\alpha K/N)}{1-\cos(2\pi\alpha/N)} \leq \frac{2}{1-\cos(2\pi\alpha/N)}\,.$$

Using $\frac{x^2}{2}(1-\frac{x^2}{12}) \leq 1-\cos(x) \leq \frac{x^2}{2}$ and the fact that $\cos(x) = \cos(-x)$ (for the case $\alpha = N - |\alpha|_N$) we get that

$$\left|S_{\alpha,K}\right|^2 \leq \frac{2}{\frac{4\pi^2|\alpha|_N^2}{2N^2}\left(1-\frac{4\pi^2|\alpha|_N^2}{12N^2}\right)} \leq \frac{N^2}{|\alpha|_N^2\left(\pi^2-\frac{\pi^4}{12}\right)}\,.$$

□

Lemma 3.14 ([2, Claim 4.1])**.** *Let $\epsilon > 0$. For a basic t-segment predicate $P : \mathbb{Z}_N \to \{\pm 1\}$, P is concentrated within ϵ on $\Gamma = \{\chi_\alpha \mid |\alpha|_N \leq O(t^2/\epsilon)\}$, i.e.*

$$\|P|_{\{\chi_\alpha \mid |\alpha|_n > O(t^2/\epsilon)\}}\|_2^2 \leq \epsilon\,.$$

Proof. First, suppose that P is a 2-segment predicate. That is, there exists a segment I such that $P(x) = 1$ for $x \in I$, and $P(x) = -1$ for $x \notin I$. Therefore

$$\begin{aligned}\left|\widehat{P}(\alpha)\right| &= \left|\frac{1}{N}\sum_{x\in\mathbb{Z}_n} P(x)\chi_{-\alpha}(x)\right| = \frac{1}{N}\left|\sum_{x\in I}\chi_{-\alpha}(x) - \sum_{x\notin I}\chi_{-\alpha}(x)\right| \\ &\leq \frac{1}{N}\left|\sum_{x\in I}\chi_{-\alpha}(x)\right| + \frac{1}{N}\left|\sum_{x\notin I}\chi_{-\alpha}(x)\right|.\end{aligned}$$

Clearly, each of these sums can be expressed as the difference of two sums of the form $S_{\alpha,K}$ where K takes the values of the end points of I. Using Claim 3.13 we get that $\left|\widehat{P}(\alpha)\right| < O(1/\|\alpha|_N)$.

Now, let P be a basic t-segment predicate. The predict P partitions $\mathbb{Z}_N$ into t (distinct) segments I_j, such that P is constant on I_j. One can express P as $P = t-1+\sum_{j=1}^{t} P_j$ where each predicate $P_j : \mathbb{Z}_N \to \{\pm 1\}$ is given by

$$\begin{cases} P(x), & \text{if } x \in I_j, \\ -1, & \text{otherwise.}\end{cases}$$

Since each P_j is a basic 2-segment predicate we get that for any $\alpha \in \mathbb{Z}_N^*$

$$\left|\widehat{P}(\alpha)\right| = \left|\sum_{j=1}^{t}\widehat{P}_j(\alpha)\right| \leq O\left(\frac{t}{|\alpha|_n}\right).$$

Finally,

$$\sum_{|\alpha|_N>k} \left|\widehat{P}(\alpha)\right|^2 \leq O(t^2) \sum_{|\alpha|_N>k} \frac{1}{|\alpha|_N^2} < O\left(\frac{t^2}{k}\right).$$

Therefore, for every $\epsilon > 0$,

$$\left\|P|_{\{\chi_\alpha \mid |\alpha|_N > O(t^2/\epsilon)\}}\right\|_2^2 \leq \epsilon\,.$$

□

Corollary 3.15. *Let $P : \mathbb{Z}_N \to \{\pm 1\}$ be a t-segment predicate, then P is concentrated.*

Proof. Let $\epsilon > 0$. If P is a basic t-segment predicate we are done. Otherwise, there exist a basic t-segment predicate $P' : \mathbb{Z}_N \to \{\pm 1\}$ and $a \in \mathbb{Z}_N^*$ such that $P(x) = P'(x/a)$ for every $x \in \mathbb{Z}_N$. From the scaling property of the Fourier transform $\widehat{P}(\alpha) = \widehat{P'}(\alpha a)$. Since P' is a basic t-segment predicate, it is concentrated within ϵ on $\Gamma' = \{\chi_\alpha \mid |\alpha|_N \leq O(t^2/\epsilon)\}$. Therefore P is concentrated within ϵ on $\Gamma = \{\chi_{\alpha a^{-1}} \mid |\alpha|_N \leq O(t^2/\epsilon)\}$ □

Corollary 3.16. *Let $0 \leq k \leq \log\log(N)$, then the functions $bit_k : \mathbb{Z}_N \to \{-1, 1\}$ and $bit_{n-k} : \mathbb{Z}_N \to \{-1, 1\}$, where $n = \lfloor \log(N) \rfloor$, are concentrated.*

3.3.1 The Concentration of All Single-bit Functions

While segment predicates can be used to show the concentration of "outer bits" functions, their usefulness in proving that all single-bit functions are concentrated is not clear. The latter was proved by Morillo and Ràfols [67], where instead of giving a general argument, like the one that uses segment predicates, they analyse the Fourier coefficients of each function bit_i. This work is quite complicated and requires analysing several different cases.

The following presents a general technique to prove that a family of functions is concentrated by considering only a subfamily. The main result is Theorem 3.20. As a corollary we get a new and simpler proof for the concentration of all single-bit functions. Full details and proofs of the ideas that are presented in this section can be found in the work of Joel Laity with the author of this thesis [55].

The key observation of this new approach is that while the characters in the Fourier basis of $L^2(\mathbb{Z}_n)$ consists of n-th roots of unity, "embedding" functions of $L^2(\mathbb{Z}_n)$ in other vector spaces $L^2(\mathbb{Z}_m)$ almost does not change the behaviour of their Fourier coefficients.

Definition 3.17. *Let $m, n \in \mathbb{N}$. Let $f : \mathbb{Z}_n \to \mathbb{C}$. Define $\widetilde{f} : \mathbb{Z}_m \to \mathbb{C}$ by*

$$\widetilde{f}(x) = \begin{cases} f(x) & \text{when } 0 \leq x < \min(n, m), \\ 0 & \text{otherwise.} \end{cases}$$

The main result of [55] is that if a function $f : \mathbb{Z}_n \to \mathbb{C}$ has large Fourier coefficients, then also $\widetilde{f} : \mathbb{Z}_m \to \mathbb{C}$ has large Fourier coefficients. Moreover, there is a simple relation between the sets of large Fourier coefficients of both functions: roughly speaking, if $\widehat{f}(\alpha)$ is heavy then there is a heavy coefficient $\widehat{\widetilde{f}}(\beta)$ for β in a small neighbourhood around $\lfloor \frac{m}{n}\alpha \rceil$. We have the following theorems from [55].

Theorem 3.18. *Let $\{n_k\}_{k\in\mathbb{N}}, \{m_k\}_{k\in\mathbb{N}}$ be two sequences of positive integers with $m_k \geq n_k/2$ for every $k \in \mathbb{N}$. Let $Q \in \mathbb{R}[x]$ be a polynomial. Let $\{f_k : \mathbb{Z}_{n_k} \to \mathbb{C}\}_{k\in\mathbb{N}}$ be a concentrated family of functions such that $\|f_k\|_2^2 \leq Q(\log(n_k))$ for all $k \in \mathbb{N}$. Then $\{\widetilde{f_k} : \mathbb{Z}_{m_k} \to \mathbb{C}\}_{k\in\mathbb{N}}$ is a concentrated family of functions.*

Theorem 3.19. *Let $\{n_k\}_{k\in\mathbb{N}}, \{m_k\}_{k\in\mathbb{N}}$ be two sequences of positive integers with $m_k \geq n_k/2$ for every $k \in \mathbb{N}$. Let $t := \sup_{k\in\mathbb{N}}\{n_k/m_k\}$. Let $Q \in \mathbb{R}[x]$ be a polynomial. Let $\{f_k : \mathbb{Z}_{n_k} \to \mathbb{C}\}_{k\in\mathbb{N}}$ be a family of functions such that $\|f_k\|_2^2 \leq Q(\log(n_k))$ for all $k \in \mathbb{N}$. Then if $\{f_k : \mathbb{Z}_{n_k} \to \mathbb{C}\}_{k\in\mathbb{N}}$ is an ϵ-concentrated family of functions, the family $\{\widetilde{f_k} : \mathbb{Z}_{m_k} \to \mathbb{C}\}_{k\in\mathbb{N}}$ is a $(t\epsilon + \eta)$-concentrated family of functions for any $\eta > 0$.*

We see that a concentrated function f remains concentrated even if $\widetilde{f}$ is defined over a smaller domain. This is somewhat surprising, as some information about f is lost, however, roughly speaking, since a concentrated function "behaves" like a linear combination of a small set of characters, this behaviour also holds on the smaller domain. This property allows us to prove the following theorem.

Theorem 3.20. *Consider a family of functions $\mathscr{F} = \{f_{2^k} : \mathbb{Z}_{2^k} \to \mathbb{C}\}_{k\in\mathbb{N}}$ and define the family $\mathscr{F}' = \{f_n : \mathbb{Z}_n \to \mathbb{C}\}_{n\in\mathbb{N}}$, where for each $2^{k-1} < n \leq 2^k$ we let $f_n(x) := f_{2^k}(x)$ for every $x \in \mathbb{Z}_n$. If $\mathscr{F}$ is concentrated then $\mathscr{F}'$ is concentrated.*

Proof. Consider the family $\mathscr{G} = \{g_n : \mathbb{Z}_{m_n} \to \mathbb{C}\}_{n\in\mathbb{N}}$ where for each $2^{k-1} < n \leq 2^k$ we let $m_n = 2^k$ and define $g_n := f_{2^k}$. Suppose that $\mathscr{F}$ is concentrated, then since $\mathscr{G}$ and $\mathscr{F}$ contain the same functions, $\mathscr{G}$ is concentrated (with the same polynomial for which $\mathscr{F}$ is concentrated). Note that for each n, we have that $f_n = \widetilde{g}_n$ with $1 \leq \frac{n}{m_n} < 2$. From Theorem 3.18, since $\{g_n : \mathbb{Z}_{m_n} \to \mathbb{C}\}_{n\in\mathbb{N}}$ is concentrated it follows that $\{f_n : \mathbb{Z}_n \to \mathbb{C}\}_{n\in\mathbb{N}} = \{\widetilde{g}_n : \mathbb{Z}_n \to \mathbb{C}\}_{n\in\mathbb{N}}$ is concentrated. □

Applying this result, one can prove that the i-th bit function (over all domain $\mathbb{Z}_n$) is concentrated by showing that the family of the i-th bit function on domains of the form $\mathbb{Z}_{2^k}$ is concentrated, that is, that $\{\mathrm{bit}_i : \mathbb{Z}_{2^k} \to \{-1, 1\}\}_{i<k\in\mathbb{N}}$ is concentrated. The latter can be easily proven using the structure of these functions under these domains. This is summarised in the following lemma.

Lemma 3.21. *Let $k \in \mathbb{N}$, let $0 \le i < k$, and define $bit_i : \mathbb{Z}_{2^k} \to \{-1,1\}$ by $bit_i(x) = (-1)^{x_i}$ where $x = \sum_{j=0}^{k-1} x_j 2^j$ and $x_j \in \{0,1\}$. Let $\alpha \in \mathbb{Z}_{2^k}$, then $\widehat{bit_i}(\alpha) = 0$ unless α is an odd multiple of 2^{k-i-1}, in which case $|\widehat{bit_i}(\alpha)| = O(2^{k-i}/|\alpha|_{2^k})$.*

Proof. Writing $N = 2^k$ we have $\widehat{\mathrm{bit}_i}(\alpha) = \frac{1}{N}\sum_{x=0}^{2^k-1} \mathrm{bit}_i(x)\omega_N^{-\alpha x}$. We write $x = y + 2^i b + 2^{i+1}z$ where $0 \le y < 2^i$, $0 \le z < 2^{k-(i+1)}$ and b is the i-th bit. Then

$$\begin{aligned}\widehat{\mathrm{bit}_i}(\alpha) =& \frac{1}{N}\left(\sum_{y=0}^{2^i-1}\omega_N^{-\alpha y}\right)\left(\sum_{b=0}^{1}(-1)^b\omega_N^{-\alpha 2^i b}\right)\left(\sum_{z=0}^{2^{k-i-1}-1}\omega_N^{-2^{i+1}\alpha z}\right)\\ =& \frac{1}{N}\left(\sum_{y=0}^{2^i-1}\omega_N^{-\alpha y}\right)\left(1-\omega_N^{-2^i\alpha}\right)\left(\sum_{z=0}^{2^{k-i-1}-1}\omega_N^{-2^{i+1}\alpha z}\right).\end{aligned}$$

The third sum is just a sum over all 2^{k-i-1}-th roots of unity, so it is 2^{k-i-1} when α is a multiple of 2^{k-i-1} and otherwise is zero. The middle term $(1-\omega_N^{-2^i\alpha})$ is therefore 2 when α is an odd multiple of 2^{k-i-1} and is zero if it is an even multiple. For the first sum, we know from Claim 3.13 that

$$\left|\sum_{y=0}^{2^i-1}\omega_N^{-\alpha y}\right| < \frac{N}{|\alpha|_N}. \tag{3.2}$$

The result then follows: the Fourier coefficient $\widehat{\mathrm{bit}_i}(\alpha)$ is zero when α is not an odd multiple of 2^{k-i-1} and when it is non-zero it has magnitude bounded by $2^{k-i}/|\alpha|_{2^k}$. □

This shows that bit_i is concentrated on characters which are indexed by multiples of 2^{k-i-1} of small norm.

Corollary 3.22. *For every $i \in \mathbb{N}$, the i-th bit function over domains $\mathbb{Z}_{2^k}$, i.e. $\{bit_i : \mathbb{Z}_{2^k} \to \{-1,1\}\}_{k>i}$, is concentrated.*

Proof.

$$\begin{aligned}\sum_{|\alpha|_{2^k} > d2^{k-i-1}} \left|\widehat{\mathrm{bit}_i}(\alpha)\right|^2 \le& \sum_{|\alpha|_{2^k} > d2^{k-i-1}} O\left(\frac{2^{k-i}}{|\alpha|_{2^k}}\right)^2 = \sum_{|\alpha|_{2^k} > d2^{k-i-1}} O\left(\frac{2^{k-i}}{|\beta 2^{k-i-1}|_{2^k}}\right)^2\\ =& \sum_{2^{i+1} > |\beta|_{2^k} > d} O\left(\frac{4}{|\beta|_{2^k}^2}\right) < O\left(\frac{4}{d}\right).\end{aligned}$$

Therefore, for every $\epsilon < 0$,

$$\left\|\mathrm{bit}_i\right|_{\{\chi_\alpha \,|\, |\alpha|_{2^k} > O(2^{k-i+1}/\epsilon)\} \cup \{\chi_\alpha \,|\, \alpha \nmid 2^{k-i-1}\}}\right\|_2^2 \le \epsilon.$$

□

Applying Theorem 3.20 we get the following.

Corollary 3.23. *For every $i \in \mathbb{N}$, the i-th bit function $\{bit_i : \mathbb{Z}_n \to \{-1,1\}\}_{n>2^i}$ is concentrated.*

3.4 Non-concentrated Functions

The scaling property of the Fourier transform shows that if $f, g : R \to \mathbb{C}$ such that $g = f \circ \varphi$ and $\varphi : R \to R$ is an invertible linear function, then the Fourier coefficients of g are a permutation of the Fourier coefficients of f. Moreover, a similar result holds for $g : R^m \to \mathbb{C}$ and $\varphi : R^m \to R$ using the multivariate scaling property (Lemma 3.6). An immediate corollary is that f is concentrated if and only if g is concentrated. Indeed, both functions have the same Fourier coefficients in different order (in the generalised case, the additional coefficients are zero). The same results hold if we set φ to be an affine function.

A natural question is whether there are other operations φ for which both f, g are concentrated or ϵ-concentrated. It is clear that if f is a constant function then also g is constant (and they are both concentrated). Degenerate cases of functions which are constant almost everywhere are not of interest. We focus on functions which are far from constant, which we formalise in our proof by requiring that $\widehat{f}(0) = 0$ (in other words, f is "balanced"). In addition, we restrict to the case where φ is a rational function.

Our result uses bounds of exponential sums involving rational functions. We make use of following lemma, which is a special case of [66, Theorem 2]).

Lemma 3.24. *Let p be prime. For any polynomials $f, g \in \mathbb{F}_p[x]$ such that the rational function $h = \frac{f}{g}$ is not constant in $\mathbb{F}_p$, the following bound holds*

$$\left| \sum_{\lambda \in \mathbb{F}_p} {}^{*}\omega_p^{h(\lambda)} \right| \leq (\max\{\deg(f), \deg(g)\} + u - 2)\sqrt{p} + \delta\,,$$

where $\sum^*$ means that the summation is taken over all $\lambda \in \mathbb{F}_p$ which are not poles of h and

$$(u, \delta) = \begin{cases} (v, 1) & if \ \deg(f) \leq \deg(g), \\ (v+1, 0) & if \ \deg(f) > \deg(g), \end{cases}$$

and v is the number of distinct zeros of g in the algebraic closure of $\mathbb{F}_p$.

We formulate the following result for functions on $\mathbb{Z}_p$ for a prime p, but it can be generalised to finite fields $\mathbb{F}_{p^m}$ with $m > 1$. Let $g, h \in \mathbb{Z}_p[x]$ be polynomials where h is not the constant zero. Let Z_h be the set of zeroes in $\mathbb{Z}_p$ of h. We define $\varphi(x) = g(x)/h(x)$

for all $x \in \mathbb{Z}_p \setminus Z_h$ and $\varphi(x) = 0$ otherwise (since we will assume Z_h is small compared with p it does not matter how we define φ on Z_h).

Proposition 3.25. *Let p be a sufficiently large prime. Let f be a concentrated function on $\mathbb{Z}_p$ such that $\|f\|_2 = 1$ and $\widehat{f}(0) = 0$. Let $g, h \in \mathbb{Z}_p[x]$ be polynomials of degree bounded by poly$(\log(p))$ and let Z_h be the set of zeroes of h. Define $\varphi(x)$ as above and suppose this function is non-constant. Let $\tau = 1/poly(\log(p))$. If $f \circ \varphi$ has any τ-heavy Fourier coefficients then $\varphi(x) = ax + b$ for some $a, b \in \mathbb{Z}_p$.*

Proof. Let $G = \mathbb{Z}_p$ and write $f = \sum_{\alpha \in G} \widehat{f}(\alpha)\chi_\alpha$. Let $d = \max\{\deg(g(x)), \deg(h(x))\}$. Let $\epsilon = \frac{\tau}{32d^2}$. Since f is concentrated there is a set Γ of size $poly(\log(|G|))$ such that

$$\|f - f|_\Gamma\|_2^2 \leq \epsilon = \frac{\tau}{32d^2}\,.$$

Since $\widehat{f}(0) = 0$ it follows that Γ does not contain zero.

Now consider $f_\varphi(x) = f(\varphi(x)) = \sum_{\alpha \in G} \widehat{f}(\alpha)\chi_\alpha(\varphi(x))$. Assume it has a τ-heavy coefficients; for contradiction we suppose $\varphi(x) \neq ax + b$ for any a, b. For every $\beta \in G$ we have

$$\begin{aligned}\widehat{f_\varphi}(\beta) =& \frac{1}{|G|}\sum_{x\in G} f_\varphi(x)\overline{\chi_\beta(x)} = \frac{1}{|G|}\sum_{x\in G} f(\varphi(x))\overline{\chi_\beta(x)} = \\ & \frac{1}{|G|}\sum_{x\in G}\sum_{\alpha\in G}\widehat{f}(\alpha)\chi_\alpha(\varphi(x))\overline{\chi_\beta(x)} = \frac{1}{|G|}\sum_{\alpha\in G}\widehat{f}(\alpha)\sum_{x\in G}\chi_\alpha(\varphi(x))\overline{\chi_\beta(x)} = \\ & \frac{1}{|G|}\sum_{\alpha\in G}\widehat{f}(\alpha)\sum_{x\in G}\chi_1(\alpha\varphi(x) - \beta x) = \frac{1}{|G|}\sum_{\alpha\in G}\widehat{f}(\alpha)\sum_{x\in G}\chi_1(\psi_\alpha^\beta(x))\,,\end{aligned}$$

where we denote $\psi_\alpha^\beta(x) = \alpha\varphi(x) - \beta x$. Since $\widehat{f}(0) = 0$ we can ignore the case $\alpha = 0$ and by our supposition that $\varphi \neq ax + b$ we know that there are no α, β such that ψ_α^β is constant. Hence, the last sum is a character sum satisfying the conditions of Lemma 3.24. Furthermore, $\psi_\alpha^\beta = (\alpha g(x) - \beta x h(x))/h(x)$ and so the value u in Lemma 3.24 is bounded by $\max\{\deg(g), \deg(h)\} \leq d$. Applying Lemma 3.24, we get that for every $\alpha \neq 0$ and every β it holds that $|\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))| \leq C$ where $C = 2d\sqrt{p}$.

Now note that

$$\begin{aligned}\widehat{f_\varphi}(\beta) = & \frac{1}{|G|}\sum_{\alpha\in G}\widehat{f}(\alpha)\sum_{x\in Z_h}\chi_1(\psi_\alpha^\beta(x)) \\ & + \frac{1}{|G|}\sum_{\alpha\in\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi_1(\psi_\alpha^\beta(x)) + \frac{1}{|G|}\sum_{\alpha\notin\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi_1(\psi_\alpha^\beta(x))\,.\end{aligned}$$

For the first term we note that $|\sum_{x\in Z_h}\chi_1(\psi_\alpha^\beta(x))| \leq |Z_h| \leq d$ and that $\|f\|_2 = 1$ implies

$\sum_{\alpha\in G}|\widehat{f}(\alpha)| \le \sqrt{|G|} = \sqrt{p}$ and $|\widehat{f}(\alpha)| \le 1$ for all α. Therefore

$$\left|\widehat{f_\varphi}(\beta)\right| \le \frac{d}{\sqrt{p}} + \left|\frac{1}{|G|}\sum_{\alpha\in\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))\right| + \left|\frac{1}{|G|}\sum_{\alpha\notin\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))\right|.$$

We apply the triangle inequality on the first sum and the Cauchy–Schwarz inequality on the second. Let $k = |\Gamma|$ and write $\Gamma = \{\alpha_1, \ldots, \alpha_k\}$. Then using Lemma 3.24 we get

$$\begin{aligned}\left|\frac{1}{|G|}\sum_{\alpha\in\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))\right| &= \left|\frac{1}{|G|}\sum_{j=1}^{k}\widehat{f}(\alpha_j)\sum_{x\in G\setminus Z_h}\chi(\psi_{\alpha_j}^\beta(x))\right| \le \left|\frac{1}{p}\sum_{j=1}^{k}\widehat{f}(\alpha_j)\cdot C\right| \\ &\le \frac{1}{p}\sum_{j=1}^{k}\left|\widehat{f}(\alpha_j)\right| C = \frac{2kd}{\sqrt{p}}.\end{aligned}$$

Since $k = |\Gamma| = poly(\log(p))$ we have that this bound (and similarly for the earlier bound $d/\sqrt{p}$) is negligible, so we have for example

$$\frac{d}{\sqrt{p}} + \frac{2kd}{\sqrt{p}} < 2d\sqrt{\epsilon}.$$

From Parseval's identity $\sum_{\alpha\notin\Gamma}\left|\widehat{f}(\alpha)\right|^2 = \|f - f|_\Gamma\|_2^2 \le \epsilon$. Therefore, by the Cauchy–Schwarz inequality we have

$$\begin{aligned}\left|\frac{1}{|G|}\sum_{\alpha\notin\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))\right| &\le \frac{1}{|G|}\left(\sum_{\alpha\notin\Gamma}\left|\widehat{f}(\alpha)\right|^2\right)^{\frac{1}{2}}\left(\sum_{\alpha\notin\Gamma}\left|\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))\right|^2\right)^{\frac{1}{2}} \\ &\le \frac{1}{|G|}\sqrt{\epsilon}\left(\sum_{\alpha\notin\Gamma}C^2\right)^{\frac{1}{2}}.\end{aligned}$$

Then

$$\left|\frac{1}{|G|}\sum_{\alpha\notin\Gamma}\widehat{f}(\alpha)\sum_{x\in G\setminus Z_h}\chi(\psi_\alpha^\beta(x))\right| \le \frac{\sqrt{\epsilon}\sqrt{p-k}2d\sqrt{p}}{p} \le 2d\sqrt{\epsilon}.$$

Finally, combining the bounds we get

$$\left|\widehat{f_\varphi}(\beta)\right|^2 \le \left(\frac{d}{\sqrt{p}} + \frac{2kd}{\sqrt{p}} + 2d\sqrt{\epsilon}\right)^2 < \left(4d\sqrt{\epsilon}\right)^2 = \left(4d\frac{\sqrt{\tau}}{4d\sqrt{2}}\right)^2 = \frac{\tau}{2}.$$

Therefore, for every β the coefficient $\widehat{f_\varphi}(\beta)$ is not τ-heavy for any noticeable τ. This gives

the required contradiction and so we conclude that φ is affine. $\square$

ϵ-concentrated Functions Proposition 3.25 shows that if f is concentrated (and far from constant) and $f \circ \varphi$ has significant coefficients, then φ is affine. It is natural to wonder whether the condition that f is concentrated is necessary. In fact, the result cannot be weakened in general: if $\varphi(x) = g(x)/h(x)$ is non-affine and invertible almost everywhere (such as a Möbius function $\varphi(x) = (ax + b)/(cx + d)$ where $ad - bc = 1$) then $f(x) = \chi_\alpha(x) + \chi_\beta(\varphi^{-1}(x))$ is such that $f(x)$ has a significant coefficient at α and $f \circ \varphi$ has a significant coefficient at β. However, a version of Proposition 3.25 is true for some ϵ-concentrated functions of interest. Specifically, one can show that the result in Proposition 3.25 also holds for 'noisy characters'; see [33, Section 6.1]

Chapter 4

Bit Security

In the center of this study stands the following question: what can be learnt about X given *partial information* about X? The aim of this chapter is to cast shape to this amorphous question. We present the contexts and cases of interest in which we study this problem and the terminology that is used in the following chapters.

4.1 Motivation

A *one-way function* is a function that is easy to compute and hard to invert. In other words, given an input x to a one-way function f, computing $f(x)$ is easy; however given f and $f(x)$, computing x is hard. Specifically, a polynomial-time algorithm that takes $f, f(x)$ and outputs x does not exist. But what about algorithms that output *partial information* about x? Do algorithms that determine if x is prime or not, or predict the parity of x with success better than 50%, exist?

The notion of *hardcore functions* formalises these questions. Let f be a function, a function b is *computationally hardcore for* f if given $f(x)$ it is hard to compute $b(x)$. Note that we do not require f to be one way, though this notion is of interest if inverting f is "hard enough". Moreover, the cases of interest are when given x, computing $f(x)$, and especially $b(x)$, is easy. One can define hardcore functions by infeasibility to *distinguish* $b(x)$ from random. We do not use this notion, and therefore use "hardcore" for "computationally hardcore".

We can already present first examples, for the discrete logarithm problem. Exponentiation in $\mathbb{Z}_p^*$ is easy, but taking logarithms is not at all. For a prime $p > 2$ such that $2^n < p < 2^{n+1}$ for some integer n and an element $g \in \mathbb{Z}_p^*$ define $exp_{g,p}(x) : \mathbb{Z} \to \mathbb{Z}_p^*$ by $exp_{g,p}(x) := g^x \pmod p$. Hence, given the value $g^x \pmod p$, the question is what can be learnt about x. We now show that the most significant bit function $\mathrm{MSB} : \mathbb{Z}_p \to \{0,1\}$, given by $\mathrm{MSB}(x) = 0$ if and only if $0 \leq x < 2^n$, is hardcore

for $exp_{g,p}$. Write $x = \sum_{i=0}^{n} x_i 2^i$ in its binary representation. Notice that MSB(x) outputs the coefficient of 2^n. Suppose that an algorithm $\mathcal{A}$ takes $g^x \in \mathbb{Z}_p$ and outputs MSB(x). First, obtain $\mathcal{A}(x) = x_n$; given $x_n, \ldots, x_{n-j+1}$, let $z = \sum_{i=0}^{n-j} x_i 2^{i+j}$ and compute $\left(g^x exp_{g,p}(-(x_n 2^n + \cdots + x_{n-j+1}2^{n-j+1}))\right)^{2^j} = (g^x g^{-(x_n 2^n + \cdots + x_{n-j+1}2^{n-j+1})})^{2^j} = g^{2^j(x_{n-j}2^{n-j} + \ldots + x_0)} = g^z$. Invoking $\mathcal{A}$ on g^z gives x_{n-j}. This shows that given $\mathcal{A}$ we can recursively learn all bits of x, thus computing MSB will able us to invert $exp_{g,p}$.

The fact that a function is hard to invert does not mean that nothing can be learnt about its inverse image, as the following ultimate example shows. Consider again the function $exp_{g,p}$ where now $g \in \mathbb{Z}_p^*$ is a primitive element. Given the value $g^x \pmod p$ we show how its Legendre symbol allows us to determine the value LSB(x), for the function LSB : $\mathbb{Z}_p \to \{0,1\}$ which gives the parity of its integer input. The order of $\mathbb{Z}_p^*$ is $p-1 = 2q$, and so $g^{p-1} = 1 \pmod p$, while $g^q = -1 \pmod p$. Write $x = 2n + x_0$ for $x_0 \in \{0,1\}$ $(0 \leq n < q)$. Then $(g^x)^q = g^{(2n+x_0)(p-1)/2} = g^{n(p-1)+x_0(p-1)/2} = (g^{(p-1)})^n (g^q)^{x_0} \equiv (-1)^{x_0} \pmod p$. Therefore, if $(g^x)^q \equiv 1 \pmod p$ then x is even; if $(g^x)^q \equiv -1 \pmod p$ then x is odd. Notice that this is a group-theoretic result, and therefore holds for the exponentiation function, with a primitive element, in any (finite) group of even order. It can also be generalised if the order of the group is divisible by greater powers of 2.

The general context of one-way functions goes hand in hand with binary representation of integers. Therefore the greatest interest is to study *hardcore bits*, as the examples reflect. Moreover, bits can be thought of as atoms in the representation of integers, and therefore arise great interest. If a function is one way then clearly not all bits of the inverse image can be computed. We like to know how many bits cannot be computed, or whether there are fixed bits, independent of the input, that are always hardcore. The research field that studies question of this sort is called *bit security*.

To underline the source of the presumed hardness of f it is common to say that a function b is hardcore for some well-known problem; in the examples above, instead of talking about exponentiation, we say "hardcore for DLP". Another function of interest is $\text{RSA}_{N,e} : \mathbb{Z}_N \to \mathbb{Z}_N$, given by $\text{RSA}_{N,e}(x) := x^e \pmod N$ where N is a product of two large primes (for description of the RSA cryptosystem, as for other basic notions presented here, see the book by Galbraith [31] or Stinson [88]). Hardcore bits for RSA and DLP are well studied. Every single bit is hardcore for RSA. Furthermore, in groups of prime order for which discrete logarithm is presumed to be hard, every single bit is hardcore for DLP. These results are explained later. Many examples, exercises and further discussions can be found at [31, Chapter 21]. The survey [39] covers many results on hardcore functions which we do not present here; it also covers the basic notions we present in this Chapter.

4.2 Framework

Let us delve in the previous examples as they illustrate the study of hardcore functions. The second example demonstrates how to show that a function b is *not* hardcore for f; given $f(x)$ simply show how to compute $b(x)$. On the other hand, the first example already gives a taste of how to show that b *is* hardcore for f; present a process that inverts f if computing $b(x)$ is feasible. The mathematical method is proof by contradiction: given $f(x)$ and an algorithm that computes $b(x)$, show how to compute x; if f is one way, then this is a contradiction, thus no such algorithm exists, hence b is hardcore. The key point is to give a *reduction* from computing x to computing $b(x)$. Since the question about the existence of one-way functions is unanswered, in practice we consider candidates for one-way functions. Accordingly, a proper interpretation of this kind of reductions is to say that computing b is *not easier* than inverting f. Since the converse of this statement clearly holds, we say that computing b is *as hard as* inverting f.

Oracles To place the focus on the reduction, it is common to consider an *oracle* that provides the partial information about x, namely $b(x)$. The benefit of this further abstraction (of the algorithm that computes $b(x)$) is that it unifies different scenarios in which such reductions are of interest. Besides of the theoretical side of this research, it can also be applied to *side-channel attacks*. There, one is assumed to possess some partial information related to x, with the aim of combining it to compute x. A byproduct of every bit security result is that the reduction it gives can be used for practical purposes. We briefly discuss some of this later. In general, we use oracles to simulate different cases where partial information is obtained, without specifying a specific one.

Interaction We go back to the question in the beginning of this chapter. Clearly, the (given) partial information $b(x)$ can be learnt. This of course is not sufficient in general to learn x itself as normally there are many other values that possess the same partial information. For example, knowing that x is odd, while reduces the possibilities by half, still leaves us with a very long list of potential values for x. Thus, some kind of interaction with the oracle is needed. Furthermore, these interactions have to follow some algebraic relation, as having $b(y)$ for some random y, cannot be used to learn anything about x. The algebraic relations that underlies these interactions are of great value, and will be in focus in subsequent chapters.

Success Probability It is not assumed that the algorithm always succeeds to compute $b(x)$. More generally, it is not assumed that the oracle always provides the correct value for $b(x)$. An oracle that always gives the correct value is called *perfect*; an oracle

that sometimes fails in providing the correct value (or any value) is called *imperfect* or *unreliable*. When the function b is a few single bits then the strategy of guessing $b(x)$ has good success probability. This of course is not a useful method to recover x. *Advantage* is the success probability in computing $b(x)$ with respect to the guessing strategy. The oracle's advantage is of great interest, as it reflects how strong the oracle is assumed to be. From the bit security perspective, a reduction that handles oracles with lower success probability is considered a stronger result. The results presented in this thesis are of two extremes: either the oracles are perfect, or they only have non-negligible advantage over the guessing strategy.

4.3 Diffie–Hellman & the Hidden Number Problem

Earlier in this section we discussed the notion of bit security with examples from RSA and DLP, and mentioned that it is natural to associate hardcore functions to well-studied hard problems. Thus, it is also natural to extend our definition and consider the bit security of secret keys that arising from primitives that are based on such problems. The road to consider hardcore functions for CDH and the bit security of Diffie–Hellman keys is now paved.

We formalise this idea. Consider a group G and an element $g \in G$, recall that given g^a, g^b the Diffie–Hellman key is g^{ab}. The oracle, simulating an algorithm that computes $f(g^{ab})$ for some (partial information) function f, should take the public parameters and output $f(g^{ab})$. Formally, we define the oracle $\mathcal{O}_{g,G,f}$ such that $\mathcal{O}_{g,G,f}(g^a, g^b) = f(g^{ab})$. When g, G and f (or some of them) are clear, we omit them and let $\mathcal{O} = \mathcal{O}_{g,G,f}$. Notice that a choice on the oracle's type has been made here. One can also consider the *stronger* oracle $\mathcal{O}_{G,f}$ that takes as input the point g, as well as g^a, g^b. We demonstrate the strength of this oracle later. For the majority of results, the weaker oracle is sufficient.

As mentioned, it is crucial to be able to interact with the oracle in a manner that provides some algebraic relation to the Diffie–Hellman key g^{ab}. The key observation is that the pair g^{a+r}, g^b gives rise to the Diffie–Hellman key $g^{(a+r)b} = g^{ab}g^{br}$. Therefore for any integer r, $\mathcal{O}_{g,G,f}(g^{a+r}, g^b) = f(g^{ab}t)$ where $t = g^{br}$. One can restrict to $r \in [1,|G|]$, and so $g^{a+r} = g^a g^r$ and the *multiplier* $t = (g^b)^r$ are efficiently computable. We remark that one can interact with the oracle in other ways; for example, a similar exponent approach involves g^{ar}, g^b. These ideas were presented in the seminal work of Boneh and Venkatesan [17].

We now combine these ideas and define the *hidden number problem (Diffie–Hellman)*.

> HNP(DH): Fix a group $(G, \cdot)$ and elements $g \in G, h \in \langle g \rangle$. Let f be a function defined over G and let $s \in G$ be unknown. Recover s given oracle access to

the function $f_s(x) := f(s \cdot h^x)$.

This problem is an abstraction to the previous discussion, where $s = g^{ab}$ and $h = g^b$ (in the exponent version, one would receive $f(s^x)$). Unlike the DLP case where the "hidden" exponent is an integer for any group,[1] Diffie–Hellman keys are group elements and so a solution to HNP(DH) in one group does not necessarily imply a solution in a different group. HNP(DH) is a special case of the general *hidden number problem*.

> HNP: Let $(G, \cdot)$ be a group, let f be a function defined over G, let $t_1, \ldots, t_d \in G$ and let $s \in G$ be unknown. Recover s given the d pairs $(t_i, f(s \cdot t_i))$.

Thanks to its very general language, the hidden number problem is a tremendous source for applications. We name a few. First and foremost, it applies not only to the study of bit security of CDH, but also of RSA and DLP as we sketch below. It has been extensively used in the study of partial leakage of nonces in DSA and ECDSA signatures [45, 68, 69] (this is discussed in [70, Section 4.4]) and side-channel attacks in the context of signatures [27, 4]. The latter work also gives results on decomposition techniques in elliptic curves. The numerous variants and applications of the hidden number problem are presented in the comprehensive survey [82]. This problem is studied today in its own right and is of great theoretical interest.

We remark that besides of its theoretical interest the study of HNP(DH) has another motivation. At the end of the Diffie–Hellman key exchange protocol, both parties share the key g^{ab} of a certain bit length. In practice, this bit length is (at least twice) larger than the security level the parties seek, and so some key derivation function is applied on their shared key. A natural and efficient candidate for such function is to simply take some block of bits of g^{ab} with the required bit length. It is therefore desirable to have a rigorous proof that computing such block of bits is not much easier than computing the entire key.

Self-randomisation HNP can be self-randomised in the following way. Given $u \in G$ we define a new unknown $s' = s \cdot u$, and $t' = t \cdot u^{-1}$, then from any pair $(t, f(s \cdot t))$ we produce the pair $(t', f(s' \cdot t'))$ where $f(s' \cdot t') = f(s \cdot t)$. Further randomisation can be done in HNP(DH) in a similar manner to our discussion that led to the problem. See [17, Section 4.1] for exact details. This randomisation implies that we can assume without loss of generality that h generates $\langle g \rangle$.

[1] In the context of traces, as arises in LUC and XTR, the discrete logarithm of $\mathrm{Tr}(g^x)$ is not well defined. The work [59] has found a way to get around it and showed that all bits are hardcore for DLP in the context of LUC (the group $G_{q,2}$; it seems to hold for perfect oracles), but the question of XTR (the group $G_{q,6}$) is still open.

RSA and DLP The bit security of RSA and DLP is well-studied and well-understood; it is not the main topic of our research. For completeness, we show how it relates to the hidden number problem. Recall that $\mathrm{RSA}_{N,e}(x) := x^e \pmod N$, and suppose an oracle takes x^e and outputs $f(x)$. Then, invoke the oracle on $(xt)^e \equiv x^e t^e \pmod N$ to receive $f(xt)$. Similarly, recall $exp_{g,p}(x) := g^x \pmod p$, and suppose an oracle takes g^x and outputs $f(x)$. Then, on $g^{xt} = (g^x)^t$ the oracle replies with $f(xt)$. Notice that these problems reduce to a very relaxed variant of HNP, were one can choose the multipliers.

The rest of this thesis is dedicated to the study of variants of HNP(DH), where our primary motivation is to obtain bit security results for CDH. In most cases we give a solution to the more general HNP in the same group. An exception is the elliptic curve variant of HNP(DH), where access to the oracle is essential; it is the first solution to HNP(DH) that uses the (somewhat weak) control over the multipliers. We consider different groups such as multiplicative groups of finite fields, points on elliptic curves and algebraic tori (with the partial group law) and different functions; most of the functions are different blocks of most significant bits, but for the results for LUC and XTR the most significant bits function is composed with the trace function. An exception is the isogeny case, as supersingular isogeny Diffie–Hellman is not based on DLP; we define an equivalent problem.

4.4 Types of Partial Information

The partial information we consider is restricted to "bits". However, in the context of most significant bits some alternative definitions to the classical definition, which arguably better suit modular residues, are often given in the literature. This section introduces the different models of bits that will be used throughout the following sections.

4.4.1 Most Significant Bits

Among all sorts of partial information about Diffie–Hellman keys, the most significant bits have been studied the most. Three types of most significant bits are used in the literature.

Classical Bits. The binary representation is the most natural and desirable definition to consider. Let an integer $x \in [0, p-1]$ and denote $n := \lfloor \log(p) \rfloor$, so x is an $(n+1)$-bit number. Write $x = \sum_{i=0}^{n} \varepsilon_i 2^i$ where $\varepsilon_i \in \{0, 1\}$ for every i. Then the k most significant *classical bits* of x, denoted by $\mathrm{MSB}_k(x)$ is the sequence $(\varepsilon_n, \ldots, \varepsilon_{n-k+1})$. It is sometimes

convenient to represent this information as an element of $\mathbb{Z}_p$. Notice that given $\mathrm{MSB}_k(x)$ one can compute the value $\sum_{i=n-k+1}^{n} \varepsilon_i 2^i + 2^{n-k}$ which differs from x by at most 2^{n-k}.

This definition gives rise to the following issue. When $p = 2^n + r$ for "small" r the most significant bit gives almost no information, as we expect to have $\mathrm{MSB}_1(x) = 0$ for random choice of x with very high probability.

Modular Bits. This alternative definition gives an equal likelihood for each value to appear. We divide the interval $[0, p-1] \subset \mathbb{R}$ into 2^k subintervals, and associate the k most significant *modular bits* to the subinterval x belongs to. More precisely, define $\mathrm{MSMB}_k(x)$ as the unique integer in $\{0, \ldots, 2^k - 1\}$ such that

$$\mathrm{MSMB}_k(x)\frac{p}{2^k} \leq x < (\mathrm{MSMB}_k(x) + 1)\frac{p}{2^k}\,.$$

Notice that by computing $\lfloor \mathrm{MSMB}_k(x)\frac{p}{2^k} + \frac{p}{2^{k+1}} \rceil$ we get a value which differs from x by at most $p/2^{k+1}$.

Approximation. A looser definition is to give an indeterministic approximation of x. We define the k-bit *approximation* of x, denoted $\mathrm{APPR}_k(x)$, as any integer u such that $|x - u| \leq \frac{p}{2^{k+1}}$. As opposed to the deterministic representation given by the modular bits, the approximation can be thought of as an indeterministic one. For its greater generality, this definition is frequently being used in the literature.

We remark that in the last two definitions k does not have to be an integer, which gives the notion of "fractions" of bits. Both of these definitions are known to differ from classical bits by at most one bit, as already implied above. Therefore, when working with large k, a difference of one bit is insignificant as it does not play a role in our understanding of the actual hardness. In the following, for large k, we pay very little attention to these distinctions. On the other hand, the difference should not be ignored in cases of very small k; see Chapter 9 for a few examples that illustrate the differences.

4.4.2 Other Consecutive Bits

The definition for classical bits extends naturally for any sequence of bits. For $x = \sum_{i=0}^{n} \varepsilon_i 2^i$ we define $\mathrm{Bits}_{i,i+j}(x)$ as the sequence $(\varepsilon_{i+j}, \ldots, \varepsilon_i)$ where $0 \leq i \leq i + j \leq n$. Specifically, for a single bit $0 \leq i \leq n$ we define $\mathrm{Bit}_i(x) = \varepsilon_i$. To represent these values as elements of $\mathbb{Z}_p$ one can compute $a = \sum_{l=0}^{j} \varepsilon_{i+l} 2^{i+l}$ and consider $2^i a$. It holds that $x = 2^{i+j} b + 2^i a + c$, for some unknowns $0 \leq b \leq \frac{p}{2^{i+j}}$ and $0 \leq c \leq 2^j$. Specifically, we let $\mathrm{LSB}_k(x) := x \pmod{2^k}$. We would also like to consider $x \pmod{l}$ where l is not

necessarily a power of 2. Similar to the generalisations above, we allow k to take any (positive) real value, and so in this case we define LSB_k by $\mathrm{LSB}_k(x) := x \pmod{\lceil 2^k \rceil}$. In other words, $\mathrm{LSB}_k(x)$ gives $x \bmod l$ for $2 \leq l = \lceil 2^k \rceil \leq p$.

Unlike the case of most significant bit, for any form of p and any bit $i < n$ there is no one value that almost always occurs. That is, while the portion of values with $\varepsilon_i = 0$ may not be equal to the portion of values with $\varepsilon_i = 1$, the proportion of the two is always polynomial. Specifically, for the extreme case of $p = 2^n + 2^{n-1} + 1$, there are $2^n - 1$ elements $x \in \mathbb{Z}_p^*$ with $\mathrm{Bit}_{n-1} = 0$, while only $2^{n-1} + 1$ with $\mathrm{Bit}_{n-1}(x) = 1$. Going down to $i = 0$ this proportion completely balances with $(p-1)/2$ elements $x \in \mathbb{Z}_p^*$ for which $\mathrm{Bit}_0(x) = 0$ and $(p-1)/2$ elements for which $\mathrm{Bit}_0(x) = 1$.

Chapter 5

Hidden Number Problem in Finite Fields

This chapter surveys results on the hidden number problem in finite fields, and their applications for bit security of several variants of Diffie–Hellman key exchange.

Let us first present this specialisation of the hidden number problem.

> $\mathbb{F}_q$-HNP(DH): Fix a prime p, a positive integer m, an element $g \in \mathbb{F}_q^* = \mathbb{F}_{p^m}^*$ and $h \in \langle g \rangle$. Let f be a function over $\mathbb{F}_q$, let $s \in \mathbb{F}_q^*$ be unknown and let $\mathcal{O}_{s,h}$ be an oracle that on input x computes f on the product sh^x in $\mathbb{F}_q^*$. That is, $\mathcal{O}_{s,h}(x) = f(sh^x)$. Recover s given query access to the oracle $\mathcal{O}_{s,h}$.

For $m = 1$ there is only one representation to $\mathbb{F}_p$. For $m > 1$, the representation of $\mathbb{F}_q$ may affect the ability to have or have not a solution to the problem. The following gives a representation-free approach.

Let $\{b_1, \ldots, b_m\}$ be a basis of $\mathbb{F}_{p^m}$, and let $\{\theta_1, \ldots, \theta_m\}$ be its dual basis. For the secret $s \in \mathbb{F}_{p^m}$ in the hidden number problem write $s = \sum_{i=1}^m s_i b_i$, and write $t = h^x$. Notice that each component of st can be represented as a linear combination of the s_i's. Indeed, since $st = \sum_{i=1}^m \operatorname{Tr}(st\theta_i)b_i$ one gets $(st)_j = \operatorname{Tr}(st\theta_j) = \sum_{i=1}^m s_i \operatorname{Tr}(t\theta_j b_i) = \sum_{i=1}^m s_i \tilde{t}_i^j$, where we let $\tilde{t}_i^j = \operatorname{Tr}(t\theta_j b_i)$.

The most interesting solution to $\mathbb{F}_q$-HNP is for the function $f := \mathrm{APPR}_k$ applied on one coefficient of $st = \big((st)_1, \ldots, (st)_m\big) = \big(\sum_{i=1}^m s_i \tilde{t}_i^1, \ldots, \sum_{i=1}^m s_i \tilde{t}_i^m\big)$. Solutions for other functions reduce to this problem.

The trace function allows us to convert $\mathbb{F}_q$-HNP to a linear problem modulo p. This motivates us to define the following more abstract problem, called the *multivariate hidden number problem*.

$\mathbb{F}_p$-MVHNP: Fix a prime p and a positive integer m. Let f be a function over $\mathbb{F}_p$, let $\mathbf{t}^1, \ldots, \mathbf{t}^d \in \mathbb{Z}_p^m$, let $\mathbf{s} = (s_1, \ldots, s_m) \in \mathbb{Z}_p^m \setminus \{(0, \ldots, 0)\}$ be unknown and denote $\mathbf{s} \cdot \mathbf{x} = s_1 x_1 + \cdots + s_m x_m \mod p$. Recover $\mathbf{s}$ given the d pairs $(\mathbf{t}^i, f(\mathbf{s} \cdot \mathbf{t}^i))$.

When $m = 1$ this is a univariate problem, known as $\mathbb{F}_p$-HNP. This is a very important problem and it deserves being addressed separately. However, in order to keep this section relatively short, we treat $\mathbb{F}_p$-HNP and $\mathbb{F}_p$-MVHNP as one. The solutions we present to different variants of the latter are either trivial or hold as is when setting $m = 1$.

5.1 Solutions

This section summarises the known solutions to several instances of $\mathbb{F}_p$-MVHNP. The most enlightening solution is where one sets $f := \mathrm{APPR}_k$ for $k \geq \sqrt{\log(p)} + \log\log(p)$. We explain how to reduce $\mathbb{F}_p$-MVHNP with $f := \mathrm{LSB}_k$, as well as in the case that f provides some inner bits, to $\mathbb{F}_p$-MVHNP with $f := \mathrm{APPR}_k$. The special case $k = \log(p)$ has a simpler solution.

We start by considering $\mathbb{F}_p$-MVHNP with $f := \mathrm{APPR}_k$. Given d samples of the form $(\mathbf{t}^i, h^i)$, we can express $h^i = s_1 t_1^i + \cdots + s_m t_m^i - e^i - r^i p$, where e^i are some bounded unknowns and r^i are unknown integers that reduce the value to $[0, p-1]$. The $\mathbb{F}_p$-linear operation in $\mathbb{F}_p$-MVHNP motivates the use of the linear structure of lattices. Let us describe how to construct a lattice from the multipliers, and reduce $\mathbb{F}_p$-MVHNP to the problem of finding a close lattice vector.

Define $\mathbf{v_s} := (\mathbf{s}\cdot\mathbf{t}^1 - r^1 p, \ldots, \mathbf{s}\cdot\mathbf{t}^d - r^d p, s_1/p, \ldots, s_m/p)$ and $\mathbf{u} := (h^1, \ldots, h^d, 0, \ldots, 0)$. Consider the $(d+m)$-dimensional lattice spanned by the rows of the matrix

$$L = \begin{pmatrix} p & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & p & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & 0 & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & p & 0 & \vdots & \vdots & \vdots \\ t_1^1 & t_1^2 & \ldots & \ldots & t_1^d & 1/p & \vdots & \vdots & 0 \\ t_2^1 & t_2^2 & \ldots & \ldots & t_2^d & 0 & 1/p & \vdots & 0 \\ \vdots & \vdots & \ldots & \ldots & \vdots & \vdots & \vdots & \ddots & \vdots \\ t_m^1 & t_m^2 & \ldots & \ldots & t_m^d & 0 & \ldots & \ldots & 1/p \end{pmatrix}.$$

Notice that $(-r_1, \ldots, -r_d, s_1, \ldots, s_m)L = \mathbf{v_s}$ is a lattice vector close to $\mathbf{u}$. More

precisely, as $|h^i - \mathbf{s} \cdot \mathbf{t}^i + r^i p| = |e^i| \leq p/2^{k+1}$, the Euclidean distance of each of the first d coordinates of $\mathbf{v_s}$ and $\mathbf{u}$ is at most $p/2^{k+1}$. The distance of each of the other m coordinates is at most $1/2$. Therefore, the distance between $\mathbf{v_s}$ and $\mathbf{u}$ is at most $\left(d(p/2^{k+1})^2 + m/4\right)^{1/2} \leq \sqrt{d+m}\, p/2^{k+1}$.

The second step of the solution is to recover a close lattice point to the vector $\mathbf{u}$. Using the result of Babai on CVP (Lemma 2.1) one can find a lattice point within distance $\sqrt{d+m}\, p/2^{k+1}$ from $\mathbf{u}$, if $k \geq \sqrt{\log p} + \log\log p$.

The final step is to show that for sufficiently large d, any lattice point $\mathbf{w}$ within this distance from $\mathbf{u}$ is of the special form $\mathbf{w} = \mathbf{v}_{\boldsymbol{\alpha}}$ for $\boldsymbol{\alpha} \equiv \mathbf{s} \pmod{p}$. In particular, the number of samples should satisfy $d \geq 2\sqrt{m \log(p)}$. This is known as the Uniqueness Theorem [17, Theorem 5] (see [59, Lemma 8] or [80, Lemma 6] for the multivariate case).

Proving this claim can by obtained by a simple counting argument over solutions to equations of the form

$$a_1 w_1 + \cdots + a_m w_m \equiv c \pmod{p}\,.$$

In particular, let $\mathbf{w}$ be a vector within distance $\sqrt{d+m}\, p/2^{k+1}$ from $\mathbf{u}$. Using the triangle inequality, for any $1 \leq j \leq d$

$$\left| \sum_{i=1}^{m} (w_i - s_i) t_i^j \pmod{p} \right| \leq p/2^k\,. \tag{5.1}$$

If $\mathbf{w} \equiv \mathbf{s} \pmod{p}$ this inequality holds. Otherwise, assuming for now that for every j the coefficients t_i^j distribute independently and uniformly at random in $\mathbb{Z}_p$, it is straightforward to show that the left hand side of (5.1) distributes independently and uniformly at random in $\mathbb{Z}_p$, and so this inequality is unlikely to hold for sufficiently large d.

This completes the solution to $\mathbb{F}_p$-MVHNP with $f = \mathrm{APPR}_k$, for $k \geq \sqrt{\log p} + \log\log p$ and $d \geq 2\sqrt{m \log(p)}$: recover a close lattice point $\mathbf{w}$ to the vector $\mathbf{u}$, and construct $\mathbf{s} = (s_1, \ldots, s_m)$ from the last m coordinates of $\mathbf{w}$.

Restricting the Set of Multipliers

The solution, as presented above, assumes that the multipliers can take any value in the group. This assumption is used in order to show that the left-hand side of (5.1) can take any value in $\mathbb{Z}_p$, which in its turn gives a non-trivial bound on the probability that (5.1) holds. Intuitively, a similar result should hold even if the multipliers have not necessarily perfectly uniform distribution, and as long as the left hand side of (5.1) has similar probability to lie in different equal-sized intervals of $\mathbb{Z}_p$.

This intuition can be made formal using the notions of *discrepancy* and *homogeneous distribution* which we do not formally define here. Roughly speaking, discrepancy of a

sequence of points measures how close the sequence is to be equidistributed; homogeneous distribution of a sequence of points is a stronger notion that simultaneously measures the discrepancy of all sequences formed by multiplying the original sequence by a scalar. The latter important notion is what allows us to achieve the result above.

The theory of exponential sums has been extensively applied to achieve, and improve, results as above when the multipliers are taken from proper subgroups of the original group in $\mathbb{F}_p$-MVHNP. This covers all (size-wise) cryptographically interesting subgroups. Moreover, based on the work of Bourgain and Konyagin [19], in some cases the subgroup's order can be as small as p^ϵ for fixed ϵ and sufficiently large p.

This extended solution to $\mathbb{F}_p$-MVHNP gives a solution to $\mathbb{F}_q$-HNP with $f = \mathrm{MSB}_k$, for $k \geq \sqrt{\log p} + \log\log p$, where the element h may lie in proper subgroup of $\mathbb{F}_q^*$.

Literature Review The results presented above are the outcome of numerous works. The description of the algorithm, along with a proof for the case of uniformly distributed multipliers, was given in the seminal work of Boneh and Venkatesan [17] for $\mathbb{F}_p$-HNP, i.e. the univariate case. The use of results from exponential sums to apply the solution to subgroups was first considered by González Vasco and Shparlinski [41], which was further extended in [68, 40, 83]. Explicit constructions for multivariate cases were first given by Shparlinski in [80, 79] for the *polynomial hidden number problem* (poly-HNP) and the *trace hidden number problem* (trace-HNP) respectively. The latter was improved in [59] to consider small subgroups. Finally, Verheul [91] suggested representing a single coefficient of a finite field element by the (linear) trace function. His theory is broader and applies to *summing* functions.

5.1.1 Other Partial Knowledge

Let $x \in \mathbb{Z}_p$ (as usual, represented in $[0, p-1]$), and suppose some lower bits of x are known. That is, the value $h \equiv x \pmod{l}$ is known. Write this as $h = x - el$ for some unknown $-\frac{p}{2l} < e \leq \frac{p}{2l}$. Let α be the multiplicative inverse of l as an element of $\mathbb{Z}_p$, that is $\alpha := l^{-1} \mod p$. Multiplying h by α and reducing mod p one gets

$$\alpha h \equiv \alpha x - e(l\alpha) \equiv \alpha x - e \pmod{p}.$$

Recall that $|e| \leq \frac{p}{2l}$, and so $\alpha h \mod p$ gives some most significant bits of αx. Writing $l = \lceil 2^k \rceil$, we get $|e| < \frac{p}{2l} \leq \frac{p}{2^{k+1}}$.

Thus, given k least significant bits of x, we can obtain the k-bit approximation of αx. This gives a reduction from $\mathbb{F}_p$-MVHNP with $f = \mathrm{LSB}_k$ to $\mathbb{F}_p$-MVHNP with $f = \mathrm{APPR}_k$.

The solution to the latter gives a solution to the former for $k \geq \sqrt{\log(p)} + \log\log(p)$. We get a similar result for $\mathbb{F}_q$-HNP.

This nice trick can be generalised for other bits of x, as shown in [68, Section 5.1]. Suppose we know l consecutive inner bits of x starting at position j. Writing $x = 2^{l+j}b + 2^j a + c$, we know a while $0 \leq b \leq \frac{p}{2^{l+j}}$ and $0 \leq c \leq 2^j$ are unknown. Let $h := 2^j a = x - 2^{l+j}b - c$.

By a result of Vinogradov [92, Lemma 1], for an integer w coprime to p and any positive integer λ there exist relatively prime integers α, β such that $\beta w \equiv \alpha \pmod{p}$ with $0 < \beta \leq \lambda$ and $0 < |\alpha| < p/\lambda$. The values α, β can be computed efficiently using convergents from the continued fraction expansion of w/p (see [68, Lemma 16] for exact details).

Applying this result with $w = \left((2^{l+j})^{-1} \mod p\right), \lambda = 2^{j+\frac{l}{2}}$ we have $\alpha 2^{l+j} \equiv \beta \leq \lambda = 2^{j+\frac{l}{2}}$ and $|\alpha| < \frac{p}{2^{j+\frac{l}{2}}}$. Notice that $0 \leq |\alpha|c < 2^j \frac{p}{2^{j+\frac{l}{2}}} = \frac{p}{2^{\frac{l}{2}}}$ and that

$$\left|\beta b - \frac{p}{2^{\frac{l}{2}+1}}\right| \leq \left|2^{j+\frac{l}{2}}\frac{p}{2^{l+j}} - \frac{p}{2^{\frac{l}{2}+1}}\right| = \frac{p}{2^{\frac{l}{2}}} - \frac{p}{2^{\frac{l}{2}+1}} = \frac{p}{2^{\frac{l}{2}+1}}\,.$$

Then, $h\alpha + p/2^{\frac{l}{2}+1} \equiv x\alpha - \alpha 2^{l+j}b - \alpha c + p/2^{\frac{l}{2}+1} \equiv x\alpha - \left(\beta b + \alpha c - p/2^{\frac{l}{2}+1}\right) \pmod{p}$, and $|\beta b + \alpha c - p/2^{\frac{l}{2}+1}| \leq |\beta b - p/2^{\frac{l}{2}+1}| + |\alpha|c < \frac{p}{2^{\frac{l}{2}+1}} + \frac{p}{2^{\frac{l}{2}}} = \frac{p}{2^{\frac{l}{2}+\log(\frac{2}{3})}}$.

Thus, rounding the left hand side if needed, given l consecutive inner bits of x, we can obtain roughly $\frac{l}{2}$ most significant bits of αx. This gives a reduction from $\mathbb{F}_p$-MVHNP with $2(k+1)$ consecutive inner bits to $\mathbb{F}_p$-MVHNP with $f = \mathrm{APPR}_k$. The solution to the latter gives a solution to the former for $k \geq \sqrt{\log(p)} + \log\log(p)$. We get a similar result for $\mathbb{F}_q$-HNP.

Knowledge of a Complete Component Consider the case where the oracle in $\mathbb{F}_q$-HNP provides a complete component $(st)_j$ of the product st. This is equivalent to getting the entire value $h = \sum_{i=1}^{m} s_i t_i$ in $\mathbb{F}_p$-MVHNP. Given m linearly independent equations of this form, the solution is trivial. Standard arguments show that one is expected to obtain the required linear independence.

Unreliable Oracles

The solutions above assume the oracle always responds with the correct answer, equivalently the samples in $\mathbb{F}_p$-MVHNP are not erroneous. Getting a similar result with an unreliable oracle seems much harder, as either in the linear algebra solution or in the lattice approach having a wrong sample completely changes the space of solutions. Several works have been trying to deal with the case of unreliable oracles, though none really gives an enlightening approach, as we now explain. In all of these works, one tries to

obtain a set of non-erroneous samples.

First observation is that imperfect oracles with *very high success probability* are likely to produce the required non-erroneous samples. Some details are given in [17, Section 4.1] along with the randomisation of samples.

When a complete component is given we solve using linear algebra (in $\mathbb{Z}_p$) with m (linearly independent) equations. In the case that an explicit set of wrong samples (indexed by the multipliers) is *pre-known*, Shparlinski [81] shows how to randomise the queries to the oracle to obtain equations outside of this set. We remark that the assumption to know the explicit set in advance is strong. However, there are cases where it is realistic, as we show in Section 6.2 below (in a different context and a solution that does not use lattice approach).

Moreover, if we fix the oracle's success probability ϵ, then the probability that m independent equations are error-free is ϵ^m. Now, suppose one *fixes* m, then asymptotically this probability becomes non-negligible with respect to $\log(p)^{-1}$. More precisely, the success probability ϵ^m is constant. This was already remarked and formalised in the original work of Verheul [91, Theorem 25]. Subsequent works, as [94, Theorem 4] and [93], take essentially this exact same method. The latter work combines Shparlinski's randomisation approach with the approach taken in [40] (see the following paragraph) to produce an even better probability (i.e. the constant is smaller). It is not clear that the approaches in these works have any novelty.

The lattice approach for $\mathbb{F}_p$-HNP ($m = 1$) was considered in [40] with several error models. Sufficient randomisation of oracle queries, along with a well-known hybrid argument that follows from Markov inequality, show how to produce a set of error-free samples. However, to get a polynomial-time solution there is a tradeoff with the number of given bits, which can be as low as $O(\log p/ \log\log(p))$. We remark that in the case that many bits are known one can implement the ideas of Bleichenbacher [10]: by adding sufficiently many samples, one can generate new samples that lie in bounded intervals of $\mathbb{Z}_p$. Bleichenbacher proposes an approach to solve $\mathbb{F}_p$-HNP in this case, which applies even for approximations of 1 bit, or below, and the samples can be erroneous (there is no need to generate error-free equations). This approach is used indirectly in the following Chapter (we do not explain this connection; see [33, Section 3.1] for more details).

5.2 Applications

This section lists the known bit security results that follow from the solutions to $\mathbb{F}_p$-MVHNP and $\mathbb{F}_q$-HNP. We use the fact that obtaining bit security results for the following variants of Diffie–Hellman key exchange can be reduced to solving variants of the hidden

number problem, as explained in the previous chapter.

Diffie–Hellman in Prime Fields Obtaining bit security results for the original Diffie–Hellman key exchange protocol, i.e. in the multiplicative group of a prime field, was the motivation of introducing the hidden number problem by Boneh and Venkatesan [17]. They gave the first solution to $\mathbb{F}_p$-HNP with $f := \mathrm{MSMB}_k$ for $k \geq \sqrt{\log(p)} + \log\log(p)$, which is the only known result to date. As this solution also holds for other bits (see Section 5.1.1), we get the following bit security result for Diffie–Hellman key exchange: computing the $\sqrt{\log(p)}+\log\log(p)$ most or least significant bits of the Diffie–Hellman key is as hard as computing the entire key; moreover computing any $2(\sqrt{\log(p)} + \log\log(p))$ consecutive inner bits of the Diffie–Hellman key is as hard as computing the entire key.

Diffie–Hellman in Non-prime Fields Consider the field $\mathbb{F}_{p^m}$ as an m-dimensional vector space over $\mathbb{F}_p$, and represent elements of $\mathbb{F}_{p^m}$ as vectors. The result of Verheul [91] shows that computing one component of the Diffie–Hellman key is as hard as computing the entire key. In particular, for $p = 2$, we get that computing any bit of the Diffie–Hellman key is as hard as computing the entire key. The solution to $\mathbb{F}_{p^m}$-HNP shows that computing the $\sqrt{\log(p)} + \log\log(p)$ most or least significant bits (or twice as many inner bits) of any component of the Diffie–Hellman key is as hard as computing the entire key.

Tripartite Diffie–Hellman This specialisation of the hidden number problem is almost equivalent to $\mathbb{F}_q$-HNP(DH): the oracle takes $([a]P, [a]Q)$, $([b]P, [b]Q)$ and $([c + r]P, [c+r]Q)$, and outputs partial information about $e(P, Q)^{ab(c+r)} = e(P, Q)^{abc}e(P, Q)^{abr}$. Letting $s := e(P, Q)^{abc}$ and $t := e(P, Q)^{abr}$ we get $\mathbb{F}_q$-HNP. Subsequently, computing the $\sqrt{\log(p)} + \log\log(p)$ most or least significant bits (or twice as many inner bits) of any component of the tripartite Diffie–Hellman key is as hard as computing the entire key. See [32] for complete details.

LUC & XTR Diffie–Hellman The bit security of the Diffie–Hellman variants of LUC and XTR follows from trace-HNP(DH), where one gets oracle access to $f(x) := \mathrm{MSB}_k\left(\mathrm{Tr}(g^{ab}h^x)\right)$, or simply receives samples of the form $(t^i, \mathrm{MSB}_k\left(\mathrm{Tr}(st)\right)$, and the goal is to recover $\mathrm{Tr}(s) = \mathrm{Tr}(g^{ab})$. Trace-HNP can be thought of as a specialisation of $\mathbb{F}_p$-MVHNP as we already explained: from the linearity of trace $\mathrm{Tr}(st) = \mathrm{Tr}(\sum_{i=1}^m s_i b_i \cdot t) = \sum_{i=1}^m s_i \,\mathrm{Tr}(tb_i) = \sum_{i=1}^m s_i \tilde{t}_i$.

Recall that in LUC and in XTR the Diffie–Hellman key $\mathrm{Tr}(g^{ab})$ is an element of $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$, respectively. The solution to trace-HNP shows that computing the $\sqrt{\log(p)} +$

$\log\log(p)$ most or least significant bits (or twice as many inner bits) of any component of LUC and XTR Diffie–Hellman key is as hard as computing the entire key. Moreover, with elementary facts about the trace function one can also consider the trace function over $\mathbb{F}_p$ (for XTR), or the hardness of computing g^{ab} given partial information about $\mathrm{Tr}(g^{ab})$. In both cases, the same results hold. See [79, 59] for complete details.

Chapter 6

Hidden Number Problem with Chosen Multipliers

This chapter presents the *chosen multiplier* variants of HNP with focus on $\mathbb{F}_p$-HNP and $\mathbb{F}_p$-MVHNP. We present the solutions to the former and give a solution to the latter. The tools and results presented in Chapter 3 are used to solve these hidden number problems, and we show that they cannot be used for other classes of hidden number problems, with non-linear operations. We show how these solutions are used to provide bit security results in schemes related to Diffie–Hellman key exchange. The solutions and results presented in this chapter do not apply to the original Diffie–Hellman key exchange or its elliptic curve counterpart.

6.1 Linear Operations

The chosen-multiplier hidden number problem, as its name suggests, allows one to choose the multipliers in the hidden number problem.

> $\mathbb{F}_p$-HNP(CM): Fix a prime p, let f be a function defined over $\mathbb{F}_p$ and let $s \in \mathbb{F}_p^*$ be unknown. Recover s given oracle access to the function $f_s(x) := f(sx)$.

One can think of different models of oracle access: in the *adaptive* model access to the oracle is given throughout the entire recovery procedure, so it is possible to adapt the queries throughout the process; in the *non-adaptive* model one is allowed to query only once on a set of chosen points before the recovery process starts. Clearly the latter model is stricter than the former. This distinction has no applications to bit security result, but can be used to model side-channel attacks, for example.

This problem has a long history, although the term "hidden number problem" was coined by Boneh and Venkatesan in a later stage. It was originally studied in the context

of bit security of RSA, and with the abstraction of the hidden number problem it was realised that it also applies to DLP, as to other computational problems.

If the function f in $\mathbb{F}_p$-HNP(CM) is the least significant bit function, then it can be easily solved by swapping the bits, as shown above. This idea generalises naturally to all outer $\log\log(p)$ bits. Therefore, most of the focus was directed to cases of unreliable oracles; inner bits functions, especially with unreliable oracles, is much more complex. A result for the least significant bit function for any oracle with non-negligible advantage over a guess was first given by Alexi, Chor, Goldreich and Schnorr [3], based on the work in [6]. This solution is adaptive and uses a *list-decoding* approach, which outputs the unknown s within a small-sized list (potentially with some other elements). A complete solution to $\mathbb{F}_p$-HNP(CM) with any single-bit function and for any oracle with non-negligible advantage over a guess was presented by Håstad and Näslund [46], based on the latter's thesis. This solution is adaptive and also uses a list-decoding approach. More on the history and development throughout the years can be found in the very nice survey [39].

Many of these works are very complex, especially for inner bits, and require complicated algebraic manipulations such as tweaking and untweaking bits. A new approach to $\mathbb{F}_p$-HNP(CM) was suggested by Akavia, Goldwasser and Safra [2]. This approach enjoys many advantages: first, it is very simple and clear; it enjoys a greater generality and more of a mathematical appeal than bit manipulations; it applies to a larger class of functions; the solution is non-adaptive. We rephrase the result from the original work [2, Theorem 2] and sketch its proof.

Theorem 6.1. *For any function f (over $\mathbb{F}_p$) with a non-zero τ-heavy Fourier coefficient, $\mathbb{F}_p$-HNP(CM) has a (non-adaptive) solution in time polynomial in* $\log(p), \|f\|_\infty, \tau^{-1}$.

Proof sketch. Run the (non-adaptive) SFT algorithm from Theorem 3.9 in the additive group $\mathbb{Z}_p$ with threshold τ on f and f_s to get lists L, L_s of τ-heavy coefficients for each function, respectively. If τ is not known, one can experiment with the learning algorithm (in polynomial time) to choose a suitable threshold. By the scaling property $\widehat{f_s}(\alpha) = \widehat{f}(\alpha s^{-1})$ for every $\alpha \in \mathbb{F}_p^*$. Therefore, for every $\alpha \in L_s$ (for which $\widehat{f_s}(\alpha)$ is τ-heavy) there exists $\beta \in L$ such that $\beta = \alpha s^{-1}$. The value $\alpha\beta^{-1}$ is a candidate for s. Each list is of size at most $2\|f\|_2^2/\tau$, and so the number of candidates for s is at most $\left(2\|f\|_2^2/\tau\right)^2$. One can try all possible candidates in order to match the pairs. □

Extending the 1-dimensional $\mathbb{F}_p$-HNP(CM) to higher dimensions, we get the following.

> $\mathbb{F}_p$-MVHNP(CM): Fix a prime p and a positive integer m, let f be a function defined over $\mathbb{F}_p^m$ and let $\mathbf{s} \in \mathbb{F}_p^m \setminus \{(0, \ldots, 0)\}$ be unknown. Recover $\mathbf{s}$ given oracle access to the function $f_{\mathbf{s}}(\mathbf{x}) := f(\mathbf{s} \cdot \mathbf{x}) = f(s_1 x_1 + \cdots + s_m x_m)$.

With the multivariate scaling property (Lemma 3.6) $\mathbb{F}_p$-MVHNP(CM) enjoys a similar solution to $\mathbb{F}_p$-HNP(CM).

Theorem 6.2. *For any function f (over $\mathbb{F}_p$) with a non-zero τ-heavy Fourier coefficient, $\mathbb{F}_p$-MVHNP(CM) has a (non-adaptive) solution in time polynomial in $m, \log(p), \|f\|_\infty, \tau^{-1}$.*

Proof. The proof follows from Proposition 3.7 and the proof of Theorem 6.1. We repeat the main arguments. Run the (non-adaptive) SFT algorithm from Theorem 3.9 in the additive group $\mathbb{Z}_p$ (resp. $\mathbb{Z}_p^m$) with threshold τ on f (resp. $f_{\mathbf{s}}$) to get a list L (resp. $L_{\mathbf{s}}$) of τ-heavy coefficients.

From Proposition 3.7, $(\alpha_1, \ldots, \alpha_m) \in L_{\mathbf{s}}$ if and only if there exists $\beta \in L$ such that $\alpha_j = \beta s_j$ for every $1 \leq j \leq m$. The value $(\alpha_1\beta^{-1}, \ldots, \alpha_m\beta^{-1})$ is a candidate for $\mathbf{s}$. □

Theorem 6.1 and Theorem 6.2 hold for any oracle with non-negligible advantage. In this case one needs to lower the threshold for the SFT algorithm as in general the erroneous values make the function less correlated with a small set of characters; see [33, Section 3.4] for more details. As opposed to the random-multiplier case (presented in the previous chapter), this solution does not require an explicit set of error-free samples. The theorems specifically holds for concentrated functions (see Definition 3.5) and since single-bit functions are concentrated (see Section 3.3.1) and admit $\|f\|_\infty = 1$, we get the following (the multivariate case follows also from Corollary 3.8).

Corollary 6.3. *There exist (non-adaptive) polynomial-time solutions to $\mathbb{F}_p$-HNP(CM) and $\mathbb{F}_p$-MVHNP(CM) for any single-bit function and any oracle with non-negligible advantage over guessing.*

Remark 6.4. *One can try to solve $\mathbb{F}_p$-MVHNP(CM) using the oracle queries of the form $(0, \ldots, 0, x_i, 0, \ldots, 0)$ and the solution to $\mathbb{F}_p$-HNP(CM). As mentioned above, the special interest in these problems is mostly for different cases of unreliable oracles. An oracle may be inherently wrong on such deterministic queries. The direct solution to $\mathbb{F}_p$-MVHNP(CM) allows to generate sufficient randomisation to overcome such cases.*

6.2 Applications

6.2.1 Bit Security of *non-uniform* Diffie–Hellman Related Schemes

The fundamental difference between $\mathbb{F}_p$-HNP, $\mathbb{F}_p$-MVHNP and $\mathbb{F}_p$-HNP(CM), $\mathbb{F}_p$-MVHNP(CM) is what allows us to achieve much stronger results for the latter. In non-uniform models, along with a problem's setting, one receives some *advice* bits that otherwise would be out of his reach. Using non-uniform models to study problems related to Diffie–Hellman key

exchange goes back to the work of Maurer [63]. The idea of using advice to solve different variants of the hidden number problem was first considered by Boneh and Venkatesan in their subsequent work [18]. Using advice bits, independent of the secret s, they were able to solve $\mathbb{F}_p$-HNP with uniform and independent samples for a function that outputs $2\log\log p$ most significant bits. Shparlinski and Winterhof [84] modified this work to extend the result to certain subgroups of $\mathbb{F}_p$, also under the provided advice.

Recall that our interest in the random-multiplier variants comes from $\mathbb{F}_q$-HNP(DH), where one has some (very weak) control over the multipliers, namely h^x on chosen x. Achieving a strong control over the multipliers in this case seems hopeless, as this means one needs to solve the discrete logarithm problem in $\mathbb{F}_q^*$ – which will make the Diffie–Hellman problem vacuous.

The advice needed in order to reduce $\mathbb{F}_q$-HNP(DH) to its chosen-multiplier counterparts is discrete logarithms to the base h. Recall that $h = g^b$ in Diffie–Hellman key exchange, so obtaining specific discrete logarithm values for h is not helpful to prove bit security results for Diffie–Hellman key exchange, as the value g^b changes in every key share, unless one considers static Diffie–Hellman. Consequently, Boneh and Venkatesan consider Diffie–Hellman related schemes where the value g^b is fixed. In particular, they addressed ElGamal's public key system and Okamoto's conference key sharing scheme. See [18, Section 3] for details on these schemes and Theorem 3.2 there for their exact results.

This approach can be used to obtain stronger results. Based on Corollary 6.3, and assuming advice bits – discrete logarithms to base g^b – that depend only on p and g (and not on the secret s), one gets the following.

Corollary 6.5. *Given advice bits depending only on p and h, there exist (non-adaptive) polynomial-time solutions to $\mathbb{F}_q$-HNP(DH) with any single-bit function and for any oracle with non-negligible advantage over a guess.*

Akavia [1] used this result for $\mathbb{F}_p$-HNP(DH) to give single-bit results for ElGamal's public key system and Okamoto's conference key sharing scheme taking place in $\mathbb{Z}_p^*$. The equivalence between $\mathbb{F}_{p^m}$-HNP to $\mathbb{F}_p$-MVHNP is explained in Chapter 5, and so one obtains the same results for ElGamal's public key system and Okamoto's conference key sharing scheme, taking place in $\mathbb{F}_{p^m}^*$.

We remark that while the number of discrete logarithms given as advice in the lattice approach of Boneh and Venkatesan is bounded by $2\log(p)$ (in fact one can take $\log(p) + \log\log(p) + 4$), the number of discrete logarithms given as advice needed for the SFT algorithm (and subsequently the solution in Corollary 6.5) is larger, but still linear in $\log(p)$. It should also be noted that when g is restricted to a small subgroup, it is not

clear that the required chosen multipliers lie in the group generated by g (see [33, Section 6.2] for more details).

6.2.2 Hardness of Computing Bits of Diffie–Hellman Keys in Different Group Representations Simultaneously

The results for $\mathbb{F}_q$-HNP(DH), presented in the previous chapter, are quite weak compared to the result presented in this chapter for $\mathbb{F}_p$-HNP(CM). This fact led researchers to consider a broader study of the bit security of Diffie–Hellman key exchange that exploits the solution of $\mathbb{F}_p$-HNP(CM). One study of this kind exploits different representations of the group, as we now explain.

In Chapter 4 we mentioned that instead of the oracle $\mathcal{O}_{G,g}$, for which the point g is "embedded" to, one can consider a stronger oracle $\mathcal{O}_G$, that also takes as input the base point g, that is $\mathcal{O}_G(g, g^a, g^b) = f(g^{ab})$. In this section we present an even stronger oracle $\mathcal{O}$ that also takes as input the group representation $\Phi(G)$, that is $\mathcal{O}(\Phi(G), g, g^a, g^b) = f\left(\Phi(g^{ab})\right)$.

Framework

Let Φ be a homomorphism on G. Given $g, g^a, g^b \in G$ one can compute $\Phi(g)$, $\Phi(g)^a = \Phi(g^a)$ and $\Phi(g)^b = \Phi(g^b)$. The former triple gives rise to the Diffie–Hellman key g^{ab} and the latter triple to $\Phi(g)^{ab}$. Fix some representation for G. Consider an oracle $\mathcal{O}$ that takes as input a Diffie–Hellman triple (g, g^a, g^b) and *also* a group isomorphism Φ and outputs some partial information about the Diffie–Hellman key $\Phi(g)^{ab}$ follows from the triple $\left(\Phi(g), \Phi(g)^a, \Phi(g)^b\right)$ *in the representation induced from* Φ. Notice that this is different from querying the oracle $\mathcal{O}_G$ for $\mathcal{O}_G\left(\Phi(g), \Phi(g)^a, \Phi(g)^b\right)$, as the latter considers $\Phi(g)^{ab}$ in G, while the oracle $\mathcal{O}$ considers $\Phi(g)^{ab}$ in a different representation of G, where a different multiplication table is used.

The trick is to find representations of G where $\Phi(x) = rx$ for some $r \in \mathbb{Z}_p^*$ and every $x \in G$. If a family $\{\Phi^r\}_{r \in \mathbb{Z}_p^*}$ of such isomorphisms is at our disposal, we can reduce to the chosen-multiplier hidden number problem: for any chosen r apply the oracle query $\mathcal{O}(\Phi^r, g, g^a, g^b) = f\left(\Phi(g^{ab})\right) = f\left(rg^{ab}\right)$. To use the single-bit result in Corollary 6.3, it is left to find group isomorphisms of this form.

Elliptic Curves This framework was developed by Boneh and Shparlinski for elliptic curves over prime fields $\mathbb{F}_q$ where the oracle takes different Weierstrass equations of the curve [16]. Given an elliptic curve in a short Weierstrass form $W : y^2 = x^3 + Ax + B$, the curve given by $W_\lambda : Y^2 = X^3 + A\lambda^4 X + B\lambda^6$ is isomorphic to W by the mapping ϕ_λ :

$W \to W_\lambda$ that takes $P = (x, y)$ on W to $P_\lambda = (\lambda^2 x, \lambda^3 y)$ on W_λ, for any non-zero $\lambda \in \mathbb{F}_q$. Specifically, the image of the point $S = (s_x, s_y) \in W$ under ϕ_λ is $\phi_\lambda(S) = (\lambda^2 s_x, \lambda^3 s_y)$. The problem of recovering an unknown $s \in \mathbb{F}_p$ where on chosen λ one gets bits of $\lambda^d s$ is known as $\mathbb{F}_q$-HNP(CM)^d (see [16]).

To solve $\mathbb{F}_q$-HNP(CM)^2 with secret s_x, if t is a quadratic residue in $\mathbb{F}_p$, that is $t = \lambda^2$ for some $\lambda \in \mathbb{F}_q$, then by applying the isomorphism Φ^{λ^2} one can choose the multiplier t for the unknown s_x. This shows how the problem of recovering s_x can be reduced to $\mathbb{F}_p$-HNP(CM) where half of the multipliers can be chosen (assuming the characteristic of the field is not 2, which is implicit from the Weierstrass equation). For the other half, i.e. when t is not a quadratic residue in $\mathbb{F}_q$, we guess the value (the specific bit of $s_x t$). We expect to guess correctly half of the time. These guesses therefore simulates an unreliable oracle. Since the solution for $\mathbb{F}_p$-HNP(CM) holds for such oracles, i.e. when some of the values are incorrect, we can use the solution to $\mathbb{F}_p$-HNP(CM) to recover s_x. Similar arguments for the y-coordinate show that we can recover s_y in this model given only one bit.

Having one coordinate of the secret point is sufficient to recover the complete secret point as there are at most three possibilities for the other coordinate. Furthermore, a result of Shoup [78, Theorem 7] can be used to determine which of the candidates for the Diffie–Hellman key is the correct one.

This result was given by Boneh and Shparlinski [16] for the least significant bit function (using the result for $\mathbb{F}_p$-HNP(CM) in [3]; a different approach is taken in [49]), and was first noticed by Kiltz [51] to hold for every single bit (using the result for $\mathbb{F}_p$-HNP(CM) in [46]). The solution to the more general $\mathbb{F}_p$-MVHNP(CM), through an equivalent $\mathbb{F}_q$-HNP(CM), immediately shows that these results also hold when one sets $\mathbb{F}_q$ to be a finite extension field, that is for elliptic curves over extension fields. The latter case holds in a greater generality as one can also change the representation of the field, given the results below (see [34, Section 5.2.2] for details).

The result in this model has the following interpretation: given an instance of Diffie–Hellman problem $(P, [a]P, [b]P)$ in an elliptic curve over a finite field under some representation, simultaneously computing single bits of $[ab]P$ for (a non-negligible fraction of) short Weierstrass equations of the curve is as hard as computing $[ab]P$ in the original representation of the curve.

Notice that the context of Diffie–Hellman key exchange almost does not come into play in this model, the interaction with the oracle – which gives the multipliers – only uses the different isomorphisms. This method therefore holds in greater generality for other secret values in the group. This observation was used in [29] to show hardness of individual bits of elliptic curve and pairing-based functions (for elliptic curves over prime

fields).

Extension Fields The same approach can be taken with representations of non-prime fields $\mathbb{F}_{p^m}$. It is of interest to consider individual bits of the i-th component in $\mathbb{F}_p$ for every $1 \leq i \leq m$. In this case it is convenient to rephrase the problem as follows. Write $\mathbf{r} = (r_1, \ldots, r_m) \in \mathbb{F}_p^m$ and fix $1 \leq i \leq m$; we look for isomorphisms $\Phi^{\mathbf{r}} : \mathbb{F}_{p^m} \to \mathbb{F}_{p^m}$ such that $(\Phi^{\mathbf{r}}(x))_i = \sum_{j=1}^m r_j x_j$. That is, the i-th component of the image under $\Phi^{\mathbf{r}}$ is the dot product of $\mathbf{x} = (x_1, \ldots, x_m)$ and $\mathbf{r}$. This will reduce the problem to $\mathbb{F}_p$-MVHNP(CM).

A special case is for $\mathbf{r} = (0, \ldots, 0, r_i, 0, \ldots, 0)$, which gives $\mathbb{F}_p$-HNP(CM), and so one is able to recover s_i, the i-th component of the secret $\mathbf{s}$. As shown in the previous chapter, recovering one component in these problems is as hard as recovering the entire secret. This approach is taken in [30, 94] for polynomial representations of the field. Explicit isomorphisms are given for every i, except of $i = 1$ (the case of $i = 1$ is explicitly treated for $\mathbb{F}_{p^2}$). We refer to [34, Section 5.2] for more details on these solutions.

We use this approach with general $\mathbf{r} = (r_1, \ldots, r_m)$ to extend the results in this model. We give explicit isomorphisms for any $\mathbf{r}$ for normal basis representation of the field, and as an abstract vector space, for any $1 \leq i \leq m$. We also show that for polynomial representations, the property $\Phi^{\mathbf{r}}$ cannot hold for $i = 1$ with all $\mathbf{r}$. As mentioned above, these results hold for a larger class of secret values in the field.

The result in this model has the following interpretation: given an instance of the Diffie–Hellman problem (g, g^a, g^b) under some representation of a finite extension field $\mathbb{F}_{p^m}$, simultaneously computing single bits of g^{ab} in (a non-negligible fraction of) different representations (of specific form) of the field is as hard as computing g^{ab} in the original representation of $\mathbb{F}_{p^m}$.

Proposition 6.6. *Let $s = g^{ab}$ be a Diffie–Hellman key in $\mathbb{F}_{p^m}$. Given $g, g^a, g^b \in \mathbb{F}_{p^m}$, computing a single bit of s in a vector space or normal bases representation of $\mathbb{F}_{p^m}$, for representations induced from $\Phi^{\mathbf{r}}$, is as hard as computing s.*

Proof. Write $s = (s_1, \ldots, s_m)$ where $s_i \in \mathbb{F}_p$. Assume one has oracle access to a bit of component j of $\Phi^{\mathbf{r}}(s)$. Given $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m) \in \mathbb{F}_p^m$, one needs to construct an isomorphism $\Phi^{\boldsymbol{\lambda}} : \mathbb{F}_{p^m} \to \mathbb{F}_{p^m}$ between representations of the finite field such that component j of $\Phi^{\boldsymbol{\lambda}}(s)$ is of the form $\left(\Phi^{\boldsymbol{\lambda}}(s)\right)_j = \lambda_1 s_1 + \cdots + \lambda_m s_m$. The result then follows from the solution to $\mathbb{F}_p$-MVHNP(CM). We briefly discuss the construction of a suitable isomorphism in the cases of interest.

General Vector Space $\mathbb{F}_{\boldsymbol{p}}^{\boldsymbol{m}}$ Let $B_1 = \{v_1, \ldots, v_m\}, B_2 = \{u_1, \ldots, u_m\}$ be two bases of $\mathbb{F}_p^m$. The mapping $\Phi^{\boldsymbol{\lambda}}$ of an element $s = s_1 v_1 + \cdots + s_m v_m$ should satisfy

$$\Phi^{\boldsymbol{\lambda}}(s) = (*)u_1 + \cdots + (\lambda_1 s_1 + \lambda_2 s_2 + \cdots + \lambda_m s_m)u_j + \cdots + (\star)u_m \,.$$

Consider this linear map as a matrix. One can easily see that the j-th row of this matrix should be $(\lambda_1 \ , \ \lambda_2 \ , \ \ldots \ , \ \lambda_m)$. In order for the matrix to be a full rank map – therefore an isomorphism – it should be nonsingular. One can easily construct such a linear map.

Normal Basis Let $B_1 = \{\alpha, \alpha^p, \ldots, \alpha^{p^{m-1}}\}, B_2 = \{\beta, \beta^p, \ldots, \beta^{p^{m-1}}\}$ be two normal bases of $\mathbb{F}_{p^m}$. The mapping $\Phi^{\boldsymbol{\lambda}}$ of an element $s = s_1\alpha + \cdots + s_m\alpha^{p^{m-1}}$ should satisfy

$$\Phi^{\boldsymbol{\lambda}}(s) = (*)\beta + \cdots + (\lambda_1 s_1 + \lambda_2 s_2 + \cdots + \lambda_m s_m)\beta^{p^{j-1}} + \cdots + (\star)\beta^{p^{m-1}} \,. \tag{6.1}$$

Consider the linear map satisfying $\Phi^{\boldsymbol{\lambda}}(\alpha) = \lambda_j\beta + \lambda_{j-1}\beta^p + \cdots + \lambda_{j+1}\beta^{p^{m-1}}$ (indices for λ_k are taken modulo m such that $1 \le k \le m$, i.e., $\lambda_0 = \lambda_m$ and $\lambda_{m+1} = \lambda_1$). Then

$$\begin{aligned}
\Phi^{\boldsymbol{\lambda}}(s) &= \Phi^{\boldsymbol{\lambda}}(s_1\alpha + \cdots + s_m\alpha^{p^{m-1}}) = s_1\Phi^{\boldsymbol{\lambda}}(\alpha) + s_2\Phi^{\boldsymbol{\lambda}}(\alpha)^p + \cdots + s_m\Phi^{\boldsymbol{\lambda}}(\alpha)^{p^{m-1}} \\
&= s_1(\lambda_j\beta + \lambda_{j-1}\beta^p + \cdots + \lambda_{j+1}\beta^{p^{m-1}}) \\
&\quad + s_2(\lambda_j\beta + \lambda_{j-1}\beta^p + \cdots + \lambda_{j+1}\beta^{p^{m-1}})^p + \cdots \\
&\qquad + s_m(\lambda_j\beta + \lambda_{j-1}\beta^p + \cdots + \lambda_{j+1}\beta^{p^{m-1}})^{p^{m-1}} \\
&= s_1(\lambda_j\beta + \lambda_{j-1}\beta^p + \cdots + \lambda_{j+1}\beta^{p^{m-1}}) \\
&\quad + s_2(\lambda_j\beta^p + \lambda_{j-1}\beta^{p^2} + \cdots + \lambda_{j+1}\beta) + \cdots \\
&\qquad + s_m(\lambda_j\beta^{p^{m-1}} + \lambda_{j-1}\beta + \cdots + \lambda_{j+1}\beta^{p^{m-2}}) \,,
\end{aligned}$$

where the last equality follows from $\beta^{p^m} = \beta$ for normal bases. After collecting the terms for each β^{p^k} (with $0 \le k \le m-1$) one gets (6.1). In order for $\Phi^{\boldsymbol{\lambda}}$ to be an isomorphism, one needs to check that $\Phi^{\boldsymbol{\lambda}}(\alpha)^{p^m} = \Phi^{\boldsymbol{\lambda}}(\alpha)$ and that the set $\{\Phi^{\boldsymbol{\lambda}}(\alpha), \Phi^{\boldsymbol{\lambda}}(\alpha)^p, \ldots, \Phi^{\boldsymbol{\lambda}}(\alpha)^{p^{m-1}}\}$ is linearly independent. This can be easily shown: the former property follows from $\beta^{p^m} = \beta$, while the latter from the linear independence of the basis B_2. □

We address the case of polynomial representations and show it is more restrictive.

Polynomial Basis Given a polynomial $a = a_m x^{m-1} + \cdots + a_2 x + a_1$, one looks for an isomorphism $\Phi^{\boldsymbol{\lambda}}$ such that

$$\Phi^{\boldsymbol{\lambda}}(a) = (*)x^{m-1} + \cdots + (\lambda_1 a_1 + \lambda_2 a_2 + \cdots + \lambda_m a_m)x^{j-1} + \cdots + (\star)x^0 \,.$$

For the constant polynomial $1 = 0 \cdot x^{m-1} + \cdots + 0 \cdot x + 1$ one gets that the coefficient of x^{j-1} of the polynomial $\Phi^{\boldsymbol{\lambda}}(1)$ is λ_1, i.e., $\Phi^{\boldsymbol{\lambda}}(1) = \lambda_1 x^{j-1} + \ldots$. Since an isomorphism maps the identity element to the identity element, it follows that if $j \neq 1$, then λ_1 has to be 0, and if $j = 1$, then λ_1 has to be 1. Therefore, when using polynomial representations, one cannot choose multipliers for s_1 and therefore cannot recover the secret s_1 using the solution to $\mathbb{F}_p$-MVHNP(CM). One can still try to recover some, or all, of the other coefficients using the method to solve $\mathbb{F}_p$-MVHNP(CM).

Remark 6.7. *An approach of this kind was taken for hyperelliptic curves (where Diffie–Hellman key exchange takes place in the divisor class group of the curve) in [96], where one considers different Mumford representations. It is shown that computing a single bit of any component of the secret key in this model is as hard as computing the entire component. A complete proof that one can use this to recover the entire key is not given.*

This strong model is unsuitable to prove results for Diffie–Hellman key exchange in a prime field, as it has a unique representation. In Chapter 9 we consider a weaker model – we assume the decisional Diffie–Hellman assumption holds – that allows to prove the bit security of individual bits of the Diffie–Hellman key.

6.3 Non-linear Operations

The solutions to the $\mathbb{F}_p$-HNP(CM) and $\mathbb{F}_p$-MVHNP(CM) are based on Fourier analysis in additive groups and exploit the scaling property of the Fourier transform for the functions $f_s(x) := f(sx)$ and $f_{\mathbf{s}}(\mathbf{x}) := f(s_1 x_1 + \ldots + s_m x_m)$. In other words, these functions are compositions of f with a linear map.

It is natural to consider whether this approach can be used for other algebraic groups (such as elliptic curves and algebraic tori). The hidden number problem in these cases involves operations which are rational, and not linear, over $\mathbb{Z}_p$. The natural approach is to still use Fourier analysis in the additive group $(\mathbb{Z}_p, +)$ but instead of composing with a linear map, to compose with a rational function.

If such tools could be developed we might have an approach to the bit security of Diffie–Hellman key exchange in the group of elliptic curve points in certain models. There are also other interesting problems that could be approached with Fourier analysis on general groups. For example, the authors of [15] raise the question whether it is possible to apply these results to the modular inversion hidden number problem. We present all of these problems in the following chapter.

Unfortunately, there is a major obstacle to applying the SFT algorithm to these sorts of problems. Let $f : G \to \mathbb{C}$ be a function and let $f_s(x) = f \circ \varphi_s(x)$, where $\varphi_s : G \to G$

is an efficiently computable function (that depends on some value s). To generalise the proof of Theorem 6.1 one needs the following three conditions:

1. the function f has a (non-zero) τ-heavy coefficient for a non-negligible τ;
2. the function f_s has a (non-zero) τ-heavy coefficient for a non-negligible τ;
3. there exists a relation between the (τ-heavy) coefficients of f and f_s that allows to determine s (or at least a small set of candidates for s).

Since single-bit functions are concentrated, as shown in Section 3.3.1, under the conditions of Proposition 3.25, the function $f_s = f \circ \varphi_s$ has no non-negligible τ-heavy coefficients. The elliptic curve addition law and the partial group law for the algebraic tori, as well as the operation in the modular inversion hidden number problem, satisfy the conditions on φ in Proposition 3.25. Therefore, the function f_s in the corresponding hidden number problems has no heavy coefficients, and so the solution involving the SFT algorithm cannot be used.

Chapter 7

The Modular Inversion Hidden Number Problem

This chapter addresses hidden number problems with non-linear operations. We review previous work and give applications to the bit security of prime-field elliptic curve Diffie–Hellman and the torus $\mathbb{T}_2$ Diffie–Hellman.

The operation in the hidden number problem is not necessarily linear over the ground field, as happens for example for elliptic curve addition. A basic building block in addressing such problems is the *modular inversion hidden number problem* (MIHNP), which was introduced by Boneh, Halevi and Howgrave-Graham [15]. This problem is formulated as follows.

> MIHNP: Fix a prime p and positive numbers k, d. Let $s \in \mathbb{Z}_p$ be unknown and let $t_1, \ldots, t_d \in \mathbb{Z}_p \setminus \{-s\}$ be chosen independently and uniformly at random. Recover s given the d pairs $\left(t_i, \mathrm{APPR}_k\left(\frac{1}{s+t_i}\right)\right)$.

As with the original hidden number problem, also the language of this problem is broad enough to give a large number of applications, as already appears in the original paper [15]. It is also closely related to the study of the inversive congruential generator (see for example [11, 12]; this relation is made explicit in [95]).

For this problem it is not clear how to construct a lattice on which one can use Babai's result for a close lattice point, as done for the linear problems, as explained in Chapter 5. Yet, other lattice algorithms have been proven to be useful in solving the problem. We present the main basic steps of these solutions.

Recall that $h := \mathrm{APPR}_k\left(\frac{1}{s+t}\right) = \frac{1}{s+t} - e \mod p$ where $|e| \leq \frac{p}{2^{k+1}}$. First, as we want to construct a lattice, a linearization of the problem is needed. The first step is to

eliminate the denominator:

$$(h+e)(s+t) \equiv 1 \pmod{p}$$

or

$$es + hs + te \equiv 1 - ht \pmod{p}.$$

This is not satisfactory as the product es of the two unknowns s, e imposes strong difficulties, as on the one hand the size of es is not small enough in general to use algorithms for short lattice vectors, and on the other hand it prevents us from looking for a linear combination of the lattice basis with coefficients that only depend on s, as in the lattice-based approach for $\mathbb{F}_p$-MVHNP.

To overcome this issue, one can isolate s in h_d by

$$s \equiv \frac{1-(h_d+e_d)t_d}{h_d+e_d} \equiv \frac{1}{h_d+e_d} - t_d \pmod{p},$$

and substitute this value into h_j for all $1 \leq j < d$. This gives

$$1 \equiv (h_j+e_j)(s+t_j) \equiv (h_j+e_j)\left(\frac{1}{h_d+e_d} - t_d + t_j\right) \pmod{p}.$$

Multiplying by h_d+e_d and rearranging gives

$$(h_d+e_d)(h_j+e_j)(t_d-t_j) + e_d - e_j + h_d - h_j \equiv 0 \pmod{p}.$$

Finally, to get a linear combination of the unknowns we write the latter as

$$(t_d-t_j)e_de_j + (h_j(t_d-t_j)+1)e_d + (h_d(t_d-t_j)-1)e_j + h_dh_j(t_d-t_j) + h_d - h_j \equiv 0 \pmod{p}.$$

The next step consists of expressing this relation as a polynomial, in order to apply algorithms for computing small roots of polynomials. That is, the bivariate polynomial

$$\begin{aligned} F_j(X,Y) &:= A_jXY + B_jX + C_jY + D_j \\ &= (t_d-t_j)XY + (h_j(t_d-t_j)+1)X + (h_d(t_d-t_j)-1)Y + h_dh_j(t_d-t_j) + h_d - h_j \end{aligned} \tag{7.1}$$

satisfies $F_j(e_1, e_j) \equiv 0 \pmod{p}$.

Algorithms for computing small roots of modular polynomials are well known, and they are based on the ideas of Coppersmith [23]. However, in order to achieve a rigorous result, a more careful analysis is needed.

We begin with a relatively simple non-rigorous explanation of how to recover small

roots of the polynomials. This method is fully described in [15, Section 3.1]. From this method, one can derive the bounds on the roots' size.

Let $n := d - 1$. The solutions to the system of the n polynomials in (7.1) can be represented by a lattice of dimension $3n + 2$, as follows. The lattice is spanned by the rows of a matrix M of the following structure

$$M = \begin{pmatrix} E & R \\ 0 & P \end{pmatrix}$$

where E and P are diagonal square matrices of dimensions $2n + 2$ and n, respectively, and R is a $(2n + 2) \times n$ matrix. Each of the first $2n + 2$ rows of M is associated with one of the terms in (7.1), and each of the last n columns is associated with one of these equations. For example, for $n = 2$ we get the matrix (m is the bit size of p and k the number of bits we get)

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & D_1 & D_2 \\ 0 & 2^{k-m} & 0 & 0 & 0 & 0 & C_1 & 0 \\ 0 & 0 & 2^{k-m} & 0 & 0 & 0 & 0 & C_2 \\ 0 & 0 & 0 & 2^{k-m} & 0 & 0 & B_1 & B_2 \\ 0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & A_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & A_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & p & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & p \end{pmatrix}.$$

For e_d, e_j, the last n columns give us equations over the integers:

$$A_j e_d e_j + B_j e_d + C_j e_j + D_j + k_j p = 0\,.$$

For the corresponding solution vector

$$\mathbf{v} := \langle 1, e_1, \ldots, e_{d-1}, e_d, e_d e_1, \ldots, e_d e_{d-1}, k_1, \ldots, k_{d-1} \rangle\,,$$

we get that $\mathbf{v}M =$

$$\langle 1, \frac{e_1}{2^{m-k}}, \ldots, \frac{e_{d-1}}{2^{m-k}}, \frac{e_d}{2^{m-k}}, \frac{e_d e_1}{2^{2(m-k)}}, \ldots, \frac{e_d e_{d-1}}{2^{2(m-k)}}, 0, \ldots, 0 \rangle\,.$$

Therefore, $\mathbf{v}M$ is a lattice point with $2n + 2$ non-zero entries, all of which are smaller than 1 (since $|e_i| \leq p/2^{k+1} < 2^{m-k}$), so its Euclidean norm is smaller than $\sqrt{2n + 2}$.

The determinant of the lattice is $\frac{p^n}{2^{(m-k)(3n+1)}}$. We apply the heuristic assumption for

short lattice vectors and expect that $\mathbf{v}M$ is the shortest vector if

$$\sqrt{2n+2} \ll \sqrt{3n+2}\left(2^{(k-m)(3n+1)}p^n\right)^{1/(3n+2)}.$$

Substituting $p = 2^{m+O(1)}$ and ignoring lower terms we get $2^k \gg 2^{2m/3}$, and so we expect that $\mathbf{v}M$ is the shortest lattice vector when we get more than $\frac{2}{3}m$ bits. Therefore, this becomes a problem of recovering the shortest lattice vector.

Notice that even though we end with a polynomial, this lattice approach treats each monomial individually, so one can think of these polynomials as multilinear functions, with the number of variables as the number of monomials (minus the constant coefficient), hence the use of term "linearization".

This method has turned into a rigorous method in [61] (in fact, earlier implementation appears in [11, 12]). Section 7.1 below uses this rigorous approach to solve the elliptic curve hidden number problem. For now, we highlight the key steps in this approach.

Step 0**:** We assume that the secret value s does not belong to a set of small (constant) cardinality of exceptional values. This set can be described by its elements and is independent of s. Hence, one can range over each element to check if it is the secret, i.e. test if it is consistent with all samples, or just fail with a negligible probability (first randomising the secret).

Step 1**:** Using the given samples, one constructs polynomials for which a root (mod p) of bounded size corresponds to the missing information on $s \cdot t_i$.

Step 2**:** From these polynomials one constructs a certain lattice that contains a certain vector $\mathbf{e}$, generated by these roots. Since the roots are bounded in size, this vector is a very short vector in the lattice.

Step 3**:** A short lattice vector $\mathbf{f}$ is found using existing algorithms. As $\mathbf{e}$ and $\mathbf{f}$ are short lattice vectors, we expect them to be parallel, i.e. a scalar multiple of one another. In practice it is sufficient that certain coordinates differ by a scalar multiple (so $\mathbf{f}$ is a linear combination of $\mathbf{e}$ and some other small vector consists of several zero coordinates). We show that this is the case unless the multipliers belong to another exceptional set (which is defined as a set of zeros of a certain family of polynomials), that its cardinality depends on the size bound of the roots.

Step 4**:** We compute this scalar by observing the first coordinate of $\mathbf{e}$ and $\mathbf{f}$. We then compute e_i by observing specific coordinates of $\mathbf{e}$ and $\mathbf{f}$.

Step 5: Once a candidate for e_i is obtained, one can derive a candidate the secret s and test if it is consistent with all samples. The probability of the existence of $s' \neq s$ which is consistent with all samples can be shown to be negligibly small, given sufficiently many samples.

The method presented above assumes that e_j, $1 \leq j < d$, is used to satisfy only a single relation of the form (7.1). We finish this section with an analysis of cases where the e_j's satisfy more than one relation. This analysis is used later on.

From the heuristic arguments above we see that the bound on the errors e_i is derived from some proportion between the determinant of the lattice and the Euclidean norm of the vector $\mathbf{v}$. In cases where we have more relations that are satisfied by the the same root, we get a lattice of larger determinant (as its dimension is larger), but the norm of $\mathbf{v}$ is unchanged (it consists of more zeros). It is therefore important that the generating set for the lattice is linearly independent, as otherwise one would construct a lattice with determinant zero. More formally, suppose that for $1 \leq \ell \leq r$ the polynomials

$$F_j^\ell(X, Y) = A_j^\ell XY + B_j^\ell X + C_j^\ell Y + D_j^\ell \tag{7.2}$$

satisfy $F_j^\ell(e_1, e_j) \equiv 0 \pmod{p}$ for some $A_j^\ell, B_j^\ell, C_j^\ell, D_j^\ell$.

It is not necessary that all polynomials are of the same form or satisfied by the same root, only that the r relations F_j^ℓ do not introduce too many values e_j that have not been previously used. Moreover, we note that it only makes sense to discuss about polynomials of total degree greater than 1 (where we have a monomial that contains some product $e_i e_j$), as otherwise there are simpler techniques to solve linear systems, like those presented in Chapter 5. We restrict this discussion to polynomials which are satisfied by the same root. In this case, if the system $\{F_j^\ell\}_{\ell=1}^r$ consists of w variables, then r can be at most w, as otherwise the system is not independent.

Similar to above, the solutions to the system of these nr polynomials in (7.2) can be represented by a lattice M of dimension $2n + nr + 2$ of the structure

$$M = \begin{pmatrix} E & R \\ 0 & P \end{pmatrix}$$

where E and P are as above, but P is of dimension nr, and R is a $(2n+2) \times nr$ matrix. For example, for $n = 2, r = 2$ we get the matrix

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & D_1^1 & D_1^2 & D_2^1 & D_2^2 \\ 0 & 2^{k-m} & 0 & 0 & 0 & 0 & C_1^1 & C_1^2 & 0 & 0 \\ 0 & 0 & 2^{k-m} & 0 & 0 & 0 & 0 & 0 & C_2^1 & C_2^2 \\ 0 & 0 & 0 & 2^{k-m} & 0 & 0 & B_1^1 & B_1^2 & B_2^1 & B_2^2 \\ 0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & A_1^1 & A_1^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2^{2(k-m)} & 0 & 0 & A_2^1 & A_2^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & p & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & p & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p \end{pmatrix}.$$

The analysis is as above where now the vector $\mathbf{v}$ is of the form

$$\mathbf{v} := \langle 1, e_1, \ldots, e_{d-1}, e_d, e_d e_1, \ldots, e_d e_{d-1}, k_1, \ldots, k_{(d-1)r} \rangle .$$

We get that $\mathbf{v}M =$

$$\langle 1, \frac{e_1}{2^{m-k}}, \ldots, \frac{e_{d-1}}{2^{m-k}}, \frac{e_d}{2^{m-k}}, \frac{e_d e_1}{2^{2(m-k)}}, \ldots, \frac{e_d e_{d-1}}{2^{2(m-k)}}, 0, \ldots, 0 \rangle$$

is a lattice point with $2n + 2$ non-zero entries, all of which are smaller than 1, so its Euclidean norm is smaller than $\sqrt{2n+2}$ as before. On the other hand, the determinant of M has grown be a factor of $p^{n(r-1)}$ as its value is $\frac{p^{nr}}{2^{(m-k)(3n+1)}}$.

Once again, we apply the heuristic assumption for short lattice vectors and expect that $\mathbf{v}M$ is the shortest vector if

$$\sqrt{2n+2} \ll \sqrt{2n+nr+2} \left(2^{(k-m)(3n+1)} p^{nr}\right)^{1/(2n+nr+2)} .$$

As above, we expect that $\mathbf{v}M$ is the shortest lattice vector when we get more than $\frac{(3-r)}{3}m$ bits, where $1 \leq r \leq 2$.

More generally, the heuristic implies that the shortest vector should satisfy

$$\|vec\| \ll \sqrt{dim}\ det^{1/dim} ,$$

where *vec* is the short vector, *dim* is the dimension of the lattice and *det* is the determinant of the lattice. Expressing the determinant as $2^{(k-m)(an+c)}p^{rn}$ and the dimension as $(b + r)n + d$ we get

$$\frac{\|vec\|}{\sqrt{dim}} \ll \left(2^{(k-m)(an+c)} p^{rn}\right)^{1/(b+r)n+d} .$$

As shown above, we can always have a vector which its Euclidean norm is smaller than

$\sqrt{dim}$, and so we eventually get that

$$k > \frac{a-r}{a}m\,,$$

where it always holds that $r \leq w < a$ for the number of variables w, as the system is non-linear.

An important conclusion from this method is that the form of the polynomials F crucially affects the result. The result becomes weaker (k becomes larger) the larger the degree is or the more monomials and variables there are. Indeed, all of the above affect the value a, and in the simplest case where $r = 1$, we see that one needs $k > (1-\frac{1}{a})\log(p)$.

7.1 Elliptic Curve Hidden Number Problem

In this section we study the *elliptic curve hidden number problem*. Here, the group in the hidden number problem is the group of elliptic curve points over a finite field and the operation is the elliptic curve addition law. This specialisation of the hidden number problem is the following.

> EC-HNP(DH): Fix a prime p, a positive integer m, an elliptic curve E over $\mathbb{F}_{p^m}$, a point $Q \in E$ and $R \in \langle Q \rangle$. Let f be a function over E, let $P \in E$ be unknown and let $\mathcal{O}_{P,R}$ be an oracle that on input t computes f on the sum $P + [t]R$ in E. That is, $\mathcal{O}_{P,R}(t) = f(P + [t]R)$. Recover P given query access to the oracle $\mathcal{O}_{P,R}$.

In this section we only consider elliptic curve over prime fields. This is the case of greatest interest. In Section 7.1.3 we address the general case. While elliptic curve points consist of 2 coordinates, it is very common to consider only one. One of the natural problems is to take f in EC-HNP(DH) to output some most significant bits of one of the coordinates of its input. We formulate the problem as follows for $\psi \in \{x, y\}$.

> EC-HNP$_\psi$(DH): Fix a prime p, an elliptic curve E over $\mathbb{F}_p$, a point $R \in E$ and a positive number k. Let $P \in E$ be unknown and let $\mathcal{O}_{P,R}$ be an oracle that on input t computes the k most significant bits of the ψ-coordinate of $P+[t]R$. That is, $\mathcal{O}_{P,R}(t) = \mathrm{MSB}_k((P + [t]R)_\psi)$. Recover P given query access to the oracle $\mathcal{O}_{P,R}$.

As elliptic curve cryptography is of great use, this problem is of great interest. Nevertheless, very little is known about this problem. There are no known (non-trivial) solutions to this problem, even for large k's. It was first addressed by Boneh, Halevi

and Howgrave-Graham [15], in a greater generality where "Diffie–Hellman access" is not given. They give no solution to the problem, but claim that it can be solved with $k > (1-\epsilon)\log(p)$ for $\epsilon \approx 0.02$. They also suggest that heuristically it can be solved with $k > \frac{3}{5}\log(p)$ using ideas based on Coppersmiths method, similar to those described above. However, in order to obtain more relations one has to assume their linear independence. Related problems have been addressed in [16, 49] (see Chapter 6); the extension-field case was considered in [48] (see Section 7.1.3).

We give a polynomial-time solution to EC-HNP$_x$(DH) for $k > \frac{5}{6}\log(p)$. That is, if for some fixed $\epsilon > 0$ one takes $k > (\frac{5}{6}+\epsilon)\log(p)$ for sufficiently large p. We divide the proof into two parts. The main part is Theorem 7.1, that gives an algorithm to solve EC-HNP$_x$(DH) up to a certain failure probability. This probability becomes lower than 1 once $k > \frac{5}{6}\log(p)$, and vanishes asymptotically with $\log(p)$. We give a precise statement of the latter in Corollary 7.5. As achieving bit security for Diffie–Hellman schemes can be reduced to hidden number problems (as explained in Chapter 4), we get a bit security result for prime-field elliptic curve Diffie–Hellman. We give a formal statement for this bit security result in Theorem 7.6.

7.1.1 Exposition

Turning the symbolic expressions in EC-HNP$_x$(DH) into algebraic expression, one has access to the function

$$\mathcal{O}_{P,R}(t) := \mathrm{MSB}_k((P+[t]R)_x \equiv \left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q - e \pmod p\,,$$

where $Q := [t]R$, and e, as a function of t, is a random integral value satisfying $|e| \leq \frac{p}{2^{k+1}}$ (here we turn MSB_k into APPR_k as discussed in Chapter 4).

The first apparent obstacle in approaching this problem is the non-linearity of this expression. Let $h := \mathrm{MSB}_k((P+[t]R)_x$. One can use the linearization approach to form a polynomial from the expression

$$(h+e+x_P+x_Q)(x_P-x_Q)^2 \equiv (y_P-y_Q)^2 \equiv y_P^2-2y_Py_Q+y_Q^2 \equiv x_p^3+ax_P+b-2y_Py_Q+y_Q^2 \pmod p\,,$$

that vanishes on the triple (e, x_P, y_P). However, we get products of the bounded e with the unbounded x_p and an unbounded unknown y_p. Since we can query on $t = 0$ to get $h_0 := \mathrm{MSB}_k(P_x) \equiv x_P - e_0$, we can substitute $x_P \equiv h_0 + e_0$. One way to eliminate y_P is to express

$$y_p \equiv (2y_Q)^{-1}\left(x_p^3 + ax_P + b + y_Q^2 - (h+e+x_P+x_Q)(x_P-x_Q)^2\right).$$

Then substitute this value in the other relations. Alternatively, squaring both sides will give $y_p^2 \equiv x_p^3 + ax_p + b$ on the left-hand side. Both of these approaches will yield very unappealing polynomials with many more monomials and either another bounded unknown or a larger degree. Recall that the polynomial structure, i.e. its degree and the number of monomials and variables, drastically affect the result that one can achieve.

We propose an alternative way to eliminate y_P, while minimizing the polynomial structure. Using two correlated relations, we form one polynomial of (total) degree 3 as above, but with only 6 monomials and 2 bounded unknowns, one of which is twice as large. The following describes this approach.

Eliminating y_P

For some integer t consider the pair $Q = [t]R, -Q = [-t]R \in E$, and suppose $P \neq \pm Q$. Let $P = (x_P, y_P)$ and $Q = (x_Q, y_Q)$, therefore $-Q = (x_Q, -y_Q)$, and write $s_{P+Q} = \frac{y_P - y_Q}{x_P - x_Q}$ and $s_{P-Q} = \frac{y_P - y_{-Q}}{x_P - x_{-Q}} = \frac{y_P + y_Q}{x_P - x_Q}$. The following operations take place in $\mathbb{F}_p$.

$$\begin{aligned}(P+Q)_x + (P-Q)_x &= s_{P+Q}^2 - x_P - x_Q + s_{P-Q}^2 - x_P - x_Q \\ &= \left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 + \left(\frac{y_P + y_Q}{x_P - x_Q}\right)^2 - 2x_P - 2x_Q \\ &= \frac{(y_P - y_Q)^2}{(x_P - x_Q)^2} - x_P - x_Q + \frac{(y_P + y_Q)^2}{(x_P - x_Q)^2} - x_P - x_Q \\ &= 2\left(\frac{y_P^2 + y_Q^2}{(x_P - x_Q)^2} - x_P - x_Q\right) = 2\left(\frac{x_Q x_P^2 + (a + x_Q^2)x_P + ax_Q + 2b}{(x_P - x_Q)^2}\right).\end{aligned} \tag{7.3}$$

Constructing Polynomials with Small Roots

Write $h_0 = \mathrm{MSB}_k(x_P) = x_P - e_0$, $h = \mathrm{MSB}_k((P+Q)_x) = (P+Q)_x - e$ and $h' = \mathrm{MSB}_k((P-Q)_x) = (P-Q)_x - e'$. Letting $\widetilde{h} = h + h'$ and $\widetilde{e} = e + e'$ and plugging $x_P = h_0 + e_0$ in (7.3) we get

$$\begin{aligned}\widetilde{h} + \widetilde{e} &= (P+Q)_x + (P-Q)_x \\ &= 2\left(\frac{x_Q(h_0 + e_0)^2 + (a + x_Q^2)(h_0 + e_0) + ax_Q + 2b}{(h_0 + e_0 - x_Q)^2}\right).\end{aligned}$$

Multiplying by $(h_0 + e_0 - x_Q)^2$ and rearranging we get that the following bivariate polynomial

$$\begin{aligned}F(X,Y) = X^2Y &+ (\widetilde{h} - 2x_Q)X^2 + 2(h_0 - x_Q)XY + 2[\widetilde{h}(h_0 - x_Q) - 2h_0x_Q - a - x_Q^2]X \\ &+ (h_0 - x_Q)^2Y + [\widetilde{h}(h_0 - x_Q)^2 - 2h_0^2x_Q - 2(a + x_Q^2)h_0 - 2ax_Q - 4b]\end{aligned}$$

satisfies $F(e_0, \widetilde{e}) \equiv 0 \pmod{p}$.

Repeating with n different Q_i leads to n polynomials of the form

$$F_i(X, Y) = X^2Y + A_iX^2 + A_{0,i}XY + B_iX + B_{0,i}Y + C_i\,, \tag{7.4}$$

that satisfy $F_i(e_0, \widetilde{e}_i) \equiv 0 \pmod{p}$.

Recall that our aim is to find small roots for F_i; if one of these roots satisfies $X = e_0$, we can substitute in h_0 and recover x_P. Heuristic arguments, similar to those presented for MIHNP above, can be used to show that one expects to find a short vector that contains these roots for $k > \frac{5}{6}\log(p)$. Details for this approach are described in [77, Section 4.2]. We now turn to the formal statements and proofs.

7.1.2 Main Results

Theorem 7.1. *Let E be an elliptic curve over a prime field $\mathbb{F}_p$, let n be an integer and k a real number. Let an unknown $P = (x_P, y_P) \in E\backslash\{O\}$ and a known generator $R \in E\backslash\{O\}$ be points on the curve. Let $\mathcal{O}$ be a function such that $\mathcal{O}(t) = MSB_k((P + [t]R)_x)$, and denote $Q_i := [t_i]R$. Then, given a γ-SVP algorithm, there exists a randomised polynomial-time algorithm that recovers the unknown x_P with $2n + 1$ calls to $\mathcal{O}$ and a single call to the γ-SVP algorithm on a $(3n + 3)$-dimensional lattice with polynomially bounded basis, except with probability*

$$\mathcal{P}_1 \leq \frac{8^n(6\eta\Delta + 1)^{6n+3}}{(p - 2\sqrt{p} - 2)^n} + \frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2} + \frac{2n + 3}{p - 2\sqrt{p}}$$

over the choices of $x_{Q_1}, \ldots, x_{Q_n}$, when it returns no answer or a wrong answer, where $\eta = 2\gamma\sqrt{3n + 1}$ and $\Delta = \lceil \frac{p}{2^{k+1}} \rceil$.[1] *If the correct x-coordinate x_P has been recovered, the algorithm determines which of the two candidates $\pm y_P$ is the correct y-coordinate, except with probability*

$$\mathcal{P}_2 \leq \frac{(16\Delta)^n}{(p - 2\sqrt{p} - 2)^n}$$

over the choices of $x_{Q_1}, \ldots, x_{Q_n}$.

Remark 7.2. *In Theorem 7.1, as in the Corollary 7.5 below, R is taken to be a generator of E in order to give precise bounds on the probabilities. Both results hold even if R is not a generator of E, as long as it generates a "large enough" subgroup. The size of the subgroup appears in the denominator of the probabilities bounds (see footnote 2), and so the results also hold if the subgroup's order is greater than $p/poly(\log(p))$, for example. For substantially smaller subgroups, one would need to adjust the value for k.*

[1] As the matter of exact precision is not important, we set Δ to be an integer.

The proof of Theorem 7.1 is rather technical. We first give an overview of the key steps in the proof.

Overview

In the algorithmic part:

- Exceptional values for P are those for which $y_P = 0$ (follows from Claim 7.3).
- Using $\mathcal{O}$, we construct the polynomial relations (as in (7.4) above)

$$F_i(X, Y) = X^2Y + A_iX^2 + A_{0,i}XY + B_iX + B_{0,i}Y + C_i$$

  for which $F_i(e_0, \widetilde{e}_i) \equiv 0 \pmod{p}$.
- Using these relations, we construct a lattice (see (7.6)), such that the vector

$$\mathbf{e} := (\Delta^3, \Delta^2 e_0, \Delta^2\widetilde{e}_1, \ldots, \Delta^2\widetilde{e}_n, \Delta e_0^2, \Delta e_0\widetilde{e}_1, \ldots, \Delta e_0\widetilde{e}_n, e_0^2\widetilde{e}_1, \ldots, e_0^2\widetilde{e}_n)$$

  is a short lattice vector.
- We run a γ-SVP algorithm on the lattice to receive a short lattice vector

$$\mathbf{f} := (\Delta^3 f_0', \Delta^2 f_0, \Delta^2 f_1 \ldots, \Delta^2 f_n, \Delta f_{0,0}, \Delta f_{0,1}, \ldots, \Delta f_{0,n}, f_{00,1}, \ldots, f_{00,n})\,.$$

  As $\mathbf{e}$ and $\mathbf{f}$ are two short lattice vectors, we expect them to be a (scalar) multiple of each other.
- Supposing this is the case, the scalar f_0' is found by observing the first coordinate of $\mathbf{e}$ and $\mathbf{f}$. We then compute $e_0 = f_0/f_0'$ provided $f_0' \neq 0$.
- From the relation $h_0 = x_P - e_0$ we derive $x_P = h_0 + e_0$.

The second part of the proof analyzes the success probability of the algorithm, as follows:

- We present 3 events for which the algorithm fails. The first ($y_P = 0$) has negligible probability.
- To derive the probability of the other events we form a certain family of low-degree polynomials (see (7.14)), for which we are interested in their set of zeros. The number of polynomials in the family is a function of $\Delta = \lceil \frac{p}{2^{k+1}} \rceil$, and so a function of k.
- Claim 7.3 shows that if $y_P \neq 0$, then the polynomials are not identically zero.

- We show that these events occur if the points x_{Q_i} are roots of some of these polynomials. Thus, we derive an exact expression of the probability of these events to hold.

The last part of the proof shows how one can determine the correct value for y_P using a consistency check with all of the given values.

Proof. Assume without loss of generality $3\eta\Delta \leq 3\eta\Delta^3 < p$, as otherwise the bound on the probability makes the claim trivial, and that the unknown P is chosen at random (as it can be self-randomised). Throughout, unless stated otherwise, i, j are indices such that $1 \leq i \leq n$ and $0 \leq j \leq n$. Set $t_0 = 0$, choose $t_i \in [1, \#E - 1]$ independently and uniformly at random, and query the oracle $\mathcal{O}$ on $\pm t_j$ to get the $2n + 1$ values $\mathcal{O}(\pm t_j)$ denoted by $h_0 = \mathrm{MSB}_k(P_x) = x_P - e_0$, $h_i = \mathrm{MSB}_k((P + Q_i)_x) = (P + Q_i)_x - e_i$ and $h_{i'} = \mathrm{MSB}_k((P - Q_i)_x) = (P - Q_i)_x - e_{i'}$, for some integers $-\Delta \leq e_j, e_{i'} \leq \Delta$. Denote $\widetilde{h}_i = h_i + h_{i'}$ and $\widetilde{e}_i = e_i + e_{i'}$, and suppose $P \neq \pm Q_i$.

The following has been shown in Section 7.1.1. For every $1 \leq i \leq n$, one has

$$\begin{aligned}\widetilde{h}_i + \widetilde{e}_i &= h_i + e_i + h_{i'} + e_{i'} = (P + Q_i)_x + (P - Q_i)_x \\ &\equiv 2\left(\frac{x_{Q_i}(h_0 + e_0)^2 + (a + x_{Q_i}^2)(h_0 + e_0) + a x_{Q_i} + 2b}{(h_0 + e_0 - x_{Q_i})^2}\right) \pmod{p}.\end{aligned}$$

Consider the polynomials

$$F_i(X, Y) := X^2 Y + A_i X^2 + A_{0,i} XY + B_i X + B_{0,i} Y + C_i\,,$$

where (all congruences hold mod p)

$$\begin{aligned} &A_i \equiv \widetilde{h}_i - 2x_{Q_i}\,, && A_{0,i} \equiv 2(h_0 - x_{Q_i})\,, \\ &B_i \equiv 2[\widetilde{h}_i(h_0 - x_{Q_i}) - 2h_0 x_{Q_i} - a - x_{Q_i}^2]\,, && B_{0,i} \equiv (h_0 - x_{Q_i})^2, \text{ and} \\ &C_i \equiv \widetilde{h}_i(h_0 - x_{Q_i})^2 - 2((h_0^2 + a)x_{Q_i} + (a + x_{Q_i}^2)h_0 + 2b)\,. \end{aligned}$$

It holds that $F(e_0, \widetilde{e}_i) \equiv 0 \pmod{p}$ for every $1 \leq i \leq n$. As $e_0, \widetilde{e}_i$ are relatively small, one hopes that finding a *small solution* to one of these polynomials would allow to recover e_0 and subsequently P. To achieve this goal, we use these relations to construct a lattice and apply the γ-SVP algorithm.

Formally, we start by 'balancing' the coefficients (as lattice basis reduction algorithms

work better where all the coefficients are of similar size). For every $1 \le i \le n$, set

$$\begin{aligned} a_i &\equiv \Delta^{-1} A_i \pmod{p}, \quad a_{0,i} \equiv \Delta^{-1} A_{0,i} \pmod{p}, \\ b_i &\equiv \Delta^{-2} B_i \pmod{p}, \quad b_{0,i} \equiv \Delta^{-2} B_{0,i} \pmod{p}, \text{ and} \\ c_i &\equiv \Delta^{-3} C_i \pmod{p}. \end{aligned} \tag{7.5}$$

The vector

$$\mathbf{e} = (\Delta^3, \Delta^2 e_0, \Delta^2 \widetilde{e}_1, \ldots, \Delta^2 \widetilde{e}_n, \Delta e_0^2, \Delta e_0 \widetilde{e}_1, \ldots, \Delta e_0 \widetilde{e}_n, e_0^2 \widetilde{e}_1, \ldots, e_0^2 \widetilde{e}_n)$$

belongs to the lattice L consisting of solutions

$$\mathbf{x} = (x_0', x_0, x_1, \ldots, x_n, x_{0,0}, x_{0,1}, \ldots, x_{0,n}, x_{00,1}, \ldots, x_{00,n}) \in \mathbb{Z}^{3n+3}$$

of the congruences

$$\begin{aligned} & c_i x_0' + b_i x_0 + b_{0,i} x_i + a_i x_{0,0} + a_{0,i} x_{0,i} + x_{00,i} \equiv 0 \pmod{p}, \ 1 \le i \le n, \\ & x_0' \equiv 0 \pmod{\Delta^3}, \\ & x_j \equiv 0 \pmod{\Delta^2} \quad 0 \le j \le n, \text{ and} \\ & x_{0,j} \equiv 0 \pmod{\Delta} \quad 0 \le j \le n. \end{aligned}$$

The lattice L is generated by the rows of a $(3n+3) \times (3n+3)$ matrix M of the following structure:

$$M = \begin{pmatrix} \mathbf{\Delta^2} & 0 & M_1 \\ 0 & \mathbf{\Delta} & M_2 \\ 0 & 0 & P \end{pmatrix} \tag{7.6}$$

where $\mathbf{\Delta^2}$, $\mathbf{\Delta}$ and P are diagonal square matrices of dimensions $n+2$, $n+1$ and n, respectively, such that the diagonal of P consists of the prime p, the matrix $\mathbf{\Delta}$ consists of Δ and the matrix $\mathbf{\Delta^2}$ of Δ^2, except of the first diagonal entry which is Δ^3; and the

matrices M_1 and M_2 are of dimensions $(n+2)\times n$ and $(n+1)\times n$ respectively, given by

$$M_1 = \begin{pmatrix} -C_1 & -C_2 & \dots & -C_n \\ -B_1 & -B_2 & & -B_n \\ -B_{0,1} & 0 & & 0 \\ 0 & -B_{0,2} & & \\ \vdots & 0 & \ddots & \\ & \vdots & & \\ 0 & 0 & & -B_{0,n} \end{pmatrix}, \qquad M_2 = \begin{pmatrix} -A_1 & -A_2 & \dots & -A_n \\ -A_{0,1} & 0 & & 0 \\ 0 & -A_{0,2} & & \vdots \\ \vdots & 0 & \ddots & \\ & \vdots & & \\ 0 & 0 & & -A_{0,n} \end{pmatrix}.$$

As $|\widetilde{e}_i| = |e_i + e_{i'}| \le 2\Delta$ for every $1 \le i \le n$, we have

$$\|\mathbf{e}\| \le \sqrt{3\Delta^6 + 12n\Delta^6} = \sqrt{3+12n}\Delta^3 \le 2\Delta^3\sqrt{3n+1}\,.$$

Run the γ-SVP algorithm and denote the vector it outputs by

$$\mathbf{f} = (\Delta^3 f_0', \Delta^2 f_0, \Delta^2 f_1 \dots, \Delta^2 f_n, \Delta f_{0,0}, \Delta f_{0,1}, \dots, \Delta f_{0,n}, f_{00,1}, \dots, f_{00,n})\,, \tag{7.7}$$

where $f_0', f_j, f_{0,j}, f_{00,i} \in \mathbb{Z}$. Notice that

$$\|\mathbf{f}\| \le \gamma\|\mathbf{e}\| \le 2\gamma\Delta^3\sqrt{3n+1} = \eta\Delta^3 \text{ for } \eta = 2\gamma\sqrt{3n+1}\,,$$

and also that

$$\begin{aligned} &|f_0'| \le \|\mathbf{f}\|\Delta^{-3} \le \eta\,, \\ &|f_j| \le \|\mathbf{f}\|\Delta^{-2} \le \eta\Delta\,, \\ &|f_{0,j}| \le \|\mathbf{f}\|\Delta^{-1} \le \eta\Delta^2\,, \text{ and} \\ &|f_{00,i}| \le \|\mathbf{f}\| \le \eta\Delta^3\,. \end{aligned}$$

As $\mathbf{e}, \mathbf{f}$ are both short lattice vectors, we expect them to be scalar multiples of each other. Therefore, let

$$\mathbf{d} = f_0'\mathbf{e} - \mathbf{f} = (0, \Delta^2 d_0, \Delta^2 d_1, \dots, \Delta^2 d_n, \Delta d_{0,0}, \Delta d_{0,1}, \dots, \Delta d_{0,n}, d_{00,1}, \dots, d_{00,n})\,,$$

where

$$
\begin{aligned}
&d_0 = f'_0 e_0 - f_0\,, && |d_0| = |f'_0 e_0 - f_0| \le \eta|e_0| + |f_0| \le \eta\Delta + \eta\Delta = 2\eta\Delta\,,\\
&d_i = f'_0 \widetilde{e}_i - f_i\,, && |d_i| = |f'_0 \widetilde{e}_i - f_i| \le \eta|\widetilde{e}_i| + |f_i| \le \eta 2\Delta + \eta\Delta = 3\eta\Delta\,,\\
&d_{0,0} = f'_0 e_0^2 - f_{0,0}\,, && |d_{0,0}| = |f'_0 e_0^2 - f_{0,0}| \le \eta|e_0|^2 + |f_{0,0}| \le \eta\Delta^2 + \eta\Delta^2 = 2\eta\Delta^2\,, \qquad (7.8)\\
&d_{0,i} = f'_0 e_0 \widetilde{e}_i - f_{0,i}\,, && |d_{0,i}| = |f'_0 e_0 \widetilde{e}_i - f_{0,i}| \le \eta|e_0\widetilde{e}_i| + |f_{0,i}| \le \eta 2\Delta^2 + \eta\Delta^2 = 3\eta\Delta^2\,, \text{ and}\\
&d_{00,i} = f'_0 e_0^2 \widetilde{e}_i - f_{00,i}\,, && |d_{00,i}| = |f'_0 e_0^2 \widetilde{e}_i - f_{00,i}| \le \eta|e_0^2\widetilde{e}_i| + |f_{00,i}| \le \eta 2\Delta^3 + \eta\Delta^3 = 3\eta\Delta^3\,.
\end{aligned}
$$

Notice that if $f'_0 \neq 0$ and also one of the coordinates of $\mathbf{d}$ (except of the first one) is zero, we can recover some previously unknown information. More precisely, suppose $f'_0 \neq 0$, then

$$\text{If } d_0 = 0, \text{ then } e_0 = f_0/f'_0\,; \tag{7.9}$$

$$\text{If } d_i = 0, \text{ then } \widetilde{e}_i = f_i/f'_0\,, \quad 1 \le i \le n\,; \tag{7.10}$$

$$\text{If } d_{0,0} = 0, \text{ then } e_0^2 = f_{0,0}/f'_0\,; \tag{7.11}$$

$$\text{If } d_{0,i} = 0, \text{ then } e_0\widetilde{e}_i = f_{0,i}/f'_0\,, \quad 1 \le i \le n\,; \tag{7.12}$$

$$\text{If } d_{00,i} = 0, \text{ then } e_0^2\widetilde{e}_i = f_{00,i}/f'_0\,, \quad 1 \le i \le n\,. \tag{7.13}$$

As $\widetilde{e}_i = e_i + e_{i'}$ it is unclear how to use these values in general to recover the secret x_P. We therefore focus on e_0, from which we derive x_P. One can recover e_0 from (7.9), by combining (7.10) and (7.12) ($e_0 = f_{0,i}/f_i$), combining (7.12) and (7.13) ($e_0 = f_{00,i}/f_{0,i}$), or, up to sign, from (7.11), or by combining (7.10) and (7.13) ($e_0^2 = f_{00,i}/f_i$). However, for the sake of the proof we only focus on (7.9), thus in case $f'_0 \neq 0$ we take $h_0 + f_0/f'_0$ as the candidate for x_P, and if $f'_0 = 0$, we fail. We remark that a more involved approach can be taken (to determine e_0 and in the case $f'_0 = 0$), using the consistency check we present below.

A pseudocode for the algorithm that recovers x_P is the following.

Algorithm 1: Find x_P

1: Construct a lattice, generated by the rows of the matrix M as in (7.6).
2: Run the γ-SVP algorithm on the lattice to get the vector $\mathbf{f}$ as in (7.7).
3: **if** $f'_0 \neq 0$ **then**
  return $h_0 + f_0/f'_0$
  else
  Fail

Probability of failure

We now define the following events:

(E-1) $y_P = 0$;

(E-2) $d_0 \neq 0$ and (E-1) does not hold;

(E-3) $f_0' = 0$ and (E-1) and (E-2) do not hold.

It is clear that if none of the events hold, one can recover x_P. The requirement $y_P \neq 0$ will be made clear in Claim 7.3 below.

As there are at most 3 values for $x_P \in \mathbb{F}_p$ that satisfy the equation $x_P^3 + ax_P + b \equiv 0 \pmod{p}$, and since P is assumed to be chosen uniformly at random, the probability that (E-1) holds satisfies

$$\Pr[\text{(E-1)}] \leq \frac{3}{\#E - 1} \leq \frac{3}{p - 2\sqrt{p}}\,.$$

In order to derive a bound on the probability of the other events we form some useful equations. As

$$c_i\Delta^3 + b_i\Delta^2 e_0 + b_{0,i}\Delta^2\widetilde{e}_i + a_i\Delta e_0^2 + a_{0,i}\Delta e_0\widetilde{e}_i + e_0^2\widetilde{e}_i \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,$$

and

$$c_i\Delta^3 f_0' + b_i\Delta^2 f_0 + b_{0,i}\Delta^2 f_i + a_i\Delta f_{0,0} + a_{0,i}\Delta f_{0,i} + f_{00,i} \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,$$

we get (by the definition of $\mathbf{d}$)

$$b_i\Delta^2 d_0 + b_{0,i}\Delta^2 d_i + a_i\Delta d_{0,0} + a_{0,i}\Delta d_{0,i} + d_{00,i} \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,$$

and therefore (using (7.5) above)

$$B_i d_0 + B_{0,i} d_i + A_i d_{0,0} + A_{0,i} d_{0,i} + d_{00,i} \equiv 0 \pmod{p},\ 1 \leq i \leq n\,.$$

Multiplying by $(x_P - x_{Q_i})^2$ and using the definitions for $A_i, A_{0,i}, B_i$ and $B_{0,i}$ we get

$$\begin{aligned}(x_P - x_{Q_i})^2\Big(&2[\widetilde{h}_i(h_0 - x_{Q_i}) - 2h_0x_{Q_i} - a - x_{Q_i}^2]d_0 + (h_0^2 - 2h_0x_{Q_i} + x_{Q_i}^2)d_i \\ &+ (\widetilde{h}_i - 2x_{Q_i})d_{0,0} + 2(h_0 - x_{Q_i})d_{0,i} + d_{00,i}\Big) \equiv 0 \pmod{p},\ 1 \leq i \leq n\,,\end{aligned}$$

which simplifies, as a polynomial in x_{Q_i}, to

$$U_i x_{Q_i}^4 - V_i x_{Q_i}^3 + W_i x_{Q_i}^2 + Y_i x_{Q_i} + Z_i \equiv 0 \pmod{p},\ 1 \leq i \leq n\,, \tag{7.14}$$

where (all congruences hold mod p)

$$
\begin{aligned}
U_i &\equiv d_i - 2d_0\,,\\
V_i &\equiv 2(2x_P - 2e_0 - \widetilde{e}_i)d_0 + (4x_P - 2e_0)d_i + 2d_{0,0} + 2d_{0,i}\,,\\
W_i &\equiv 2(3x_P^3 - 6e_0x_P - 3\widetilde{e}_ix_P + e_0\widetilde{e}_i - 3a)d_0 + (6x_P^2 - 6e_0x_P + e_0^2)d_i + (6x_P - \widetilde{e}_i)d_{0,0}\\
&\quad + (6x_P - 2e_0)d_{0,i} + d_{00,i}\,,\\
Y_i &\equiv 2(3\widetilde{e}_ix_P^2 - 2e_0\widetilde{e}_ix_P + 2ax_P - 2ae_0 - 4b)d_0 - 2(2x_P^3 - 3e_0x_P^2 + e_0^2x_P)d_i \qquad (7.15)\\
&\quad + (2\widetilde{e}_ix_P + 2a)d_{0,0} - (6x_P^2 - 4e_0x_P)d_{0,i} - 2x_Pd_{00,i}\,,\ \text{and}\\
Z_i &\equiv 2(-\widetilde{e}_ix_P^3 + e_0\widetilde{e}_ix_P^2 + ax_P^2 - 2ae_0x_P + 4bx_P - 4be_0)d_0 + (x_P^4 - 2e_0x_P^3 + e_0^2x_P^2)d_i\\
&\quad + (-\widetilde{e}_ix_P^2 + 2ax_P + 4b)d_{0,0} + (2x_P^3 - 2e_0x_P^2)d_{0,i} + x_P^2d_{00,i}\,.
\end{aligned}
$$

We now show that if for some $1 \le i \le n$ the left hand side of (7.14) is the constant zero polynomial, then $d_0 = 0 = d_{0,0}$. We conclude that if $d_0 \neq 0$ or $d_{0,0} \neq 0$, then the left hand side of (7.14) is a non-constant polynomial in x_{Q_i} (of degree at most 4) for every $1 \le i \le n$.

Claim 7.3. *Let $1 \le i \le n$, and assume $y_P \neq 0$. The left hand side of (7.14) is constant if and only if $d_0 = d_{0,0} = d_i = d_{0,i} = d_{00,i} = 0$.*

Proof. The first implication is clear from (7.15). Suppose that the left hand side of (7.14) is constant for some $1 \le i \le n$. Then $U_i \equiv V_i \equiv W_i \equiv Y_i \equiv Z_i \equiv 0 \pmod p$. One can express the latter as a system of 5 equations in the 5 variables $d_0, d_i, d_{0,0}, d_{0,i}$ and $d_{00,i}$. A non-zero solution exists if and only if the system is singular. We show that the system is nonsingular if and only if $y_P \neq 0$, which completes the proof.

We use the first 4 equations to eliminate $d_i, d_{0,i}, d_{00,i}$ and remain with the "global" variables $d_0, d_{0,0}$. One then has

$$-2(2x_P^3 + 3e_0x_P^2 + 2ax_P + ae_0 + 2b)d_0 + (3x_P^2 + a)d_{0,0} \equiv 0 \pmod p\,,$$

which simplifies to

$$-4y_Pd_0 - 2e_0(3x_P^2 + a)d_0 + (3x_P^2 + a)d_{0,0} \equiv 0 \pmod p\,.$$

If $3x_P^2 + a \equiv 0 \pmod p$, then $y_Pd_0 \equiv 0 \pmod p$. Otherwise, one can express $d_{0,0}$ in terms of d_0. Plugging this value, with the other recovered variables, to the last equation, one gets

$$(x_P^6 + 2ax_P^4 + 2bx_P^3 + a^2x_P^2 + 2abx_P + b^2)d_0 \equiv y_P^4d_0 \equiv 0 \pmod p\,.$$

In both cases, since $y_P \neq 0$, we have $d_0 \equiv d_{0,0} \equiv d_i \equiv d_{0,i} \equiv d_{00,i} \equiv 0 \pmod p$, and since

all of these values are of size smaller than p (as we suppose $3\eta\Delta < 3\eta\Delta^3 < p$), the claim follows. □

We use this claim to bound the probabilities of (E-2) and (E-3), which will prove the first claim in the theorem. The probability of events (E-2) and (E-3) is taken over the choice of the points Q_i for $1 \leq i \leq n$. That is, we consider the number of n-tuples

$$(x_{Q_1}, \ldots, x_{Q_n}) \in \big(E_x \setminus \{x_P\}\big)^n$$

such that (E-2) holds or (E-3) holds, where $E_x := \{z \in \mathbb{F}_p \mid \exists Q \in E, Q_x = z\}$.[2] Note that $\#E - 1 \leq 2|E_x| \leq \#E + 2$.

Probability of event (E-2). Assume (E-2) holds, that is $d_0 \neq 0$ and $y_P \neq 0$, and fix some values of $d_j, d_{0,j}$ for $0 \leq j \leq n$ and $d_{00,i}$ for $1 \leq i \leq n$. Let us consider the number of n-tuples

$$(x_{Q_1}, \ldots, x_{Q_n}) \in \big(E_x \setminus \{x_P\}\big)^n$$

satisfying (7.14).

Since $d_0 \neq 0$ Claim 7.3 shows that the left hand side of (7.14) is non-constant for all $1 \leq i \leq n$. Thus, as all the relations in (7.14) are satisfied, there are at most 4 values x_{Q_i} that satisfy each relation, and so there are at most 4^n n-tuples that satisfy these n non-constant polynomials.

From (7.8) above we get: as $d_0 \neq 0$ it can take at most $4\eta\Delta$ values, each d_i can take at most $6\eta\Delta + 1$ values, $d_{0,0}$ can take at most $4\eta\Delta^2 + 1$ values, each $d_{0,i}$ can take at most $6\eta\Delta^2 + 1$ values, and each $d_{00,i}$ can take at most $6\eta\Delta^3 + 1$ values. Therefore, there are at most

$$\begin{aligned} &4^n 4\eta\Delta(6\eta\Delta+1)^n(4\eta\Delta^2+1)(6\eta\Delta^2+1)^n(6\eta\Delta^3+1)^n < \\ &4^n 4\eta\Delta(6\eta\Delta+1)^n(4\eta\Delta+1)^2(6\eta\Delta+1)^{2n}(6\eta\Delta+1)^{3n} < 4^n(6\eta\Delta+1)^{6n+3} \end{aligned}$$

n-tuples $(x_{Q_1}, \ldots, x_{Q_n})$ for which event (E-2) happens. Denote them by $\mathcal{Q}$. The probability that $d_0 \neq 0$ (given $y_P \neq 0$) satisfies

$$\Pr[\text{(E-2)}] \leq \frac{|\mathcal{Q}|}{\big|E_x \setminus \{x_P\}\big|^n} < \frac{4^n(6\eta\Delta+1)^{6n+3}}{\big(\frac{1}{2}(\#E-1)-1\big)^n} \leq \frac{8^n(6\eta\Delta+1)^{6n+3}}{(p-2\sqrt{p}-2)^n}\,.$$

Probability of event (E-3). Assume (E-3) holds, that is $f'_0 = 0, d_0 = 0$ and $y_P \neq 0$. We may suppose that for all the n-tuples in $\mathcal{Q}$ event (E-3) holds, and thus consider the remaining n-tuples which are not in $\mathcal{Q}$. We first notice that $d_{0,0} = 0$. Indeed, if $d_{0,0} \neq 0$,

[2] In the case that R is not a generator of E, one would define $E_x := \{z \in \mathbb{F}_p \mid \exists Q \in \langle R \rangle, Q_x = z\}$. Proving the theorem for *any* R boils down to proving that the roots of (7.14) are not restricted to E_x.

then by Claim 7.3 the left hand side of (7.14) is non-constant for all $1 \leq i \leq n$. In that case, the only n-tuples that satisfy (7.14) are in $\mathcal{Q}$. We therefore have $f_0 = f_0' e_0 - d_0 = 0 = f_0' e_0^2 - d_{0,0} = f_{0,0}$.

Consider the set $S = \{i \in \{1, \ldots, n\} \mid d_i = d_{0,i} = d_{00,i} = 0\}$. Let $l = |S|$, and notice that if $l = n$ then $f_0 = f_i = f_{0,0} = f_{0,i} = f_{00,i} = 0$, and since $f_0' = 0$ by assumption then $\mathbf{f} = 0$. As $\mathbf{f}$ is a non-zero vector by construction, $l < n$.

Fix some values of $d_i, d_{0,i}, d_{00,i}$ for $1 \leq i \leq n$. We now consider the number of n-tuples

$$(x_{Q_1}, \ldots, x_{Q_n}) \notin \mathcal{Q}$$

satisfying (7.14). If $i \in S$ then the left hand side of (7.14) is the constant zero, and so there are $|E_x| - 1$ possible values for x_{Q_i} satisfying (7.14). If $i \notin S$ then either $d_i \neq 0$ or $d_{0,i} \neq 0$ or $d_{00,i} \neq 0$ and by Claim 7.3 the left hand side of (7.14) is non-constant, so there are at most 4 solutions x_{Q_i} to the corresponding equation in (7.14).

Overall, there are at most $4^{n-l}(|E_x| - 1)^l$ n-tuples $(x_{Q_1}, \ldots, x_{Q_n}) \notin \mathcal{Q}$ that satisfy (7.14). The possible values for each $d_i, d_{0,i}, d_{00,i}$ for each $i \notin S$ are given above. So overall there are at most

$$4^{n-l}(|E_x| - 1)^l (6\eta\Delta + 1)^{n-l}(6\eta\Delta^2 + 1)^{n-l}(6\eta\Delta^3 + 1)^{n-l} <$$
$$4^{n-l}(|E_x| - 1)^l (6\eta\Delta + 1)^{n-l}(6\eta\Delta + 1)^{2(n-l)}(6\eta\Delta + 1)^{3(n-l)} = 4^{n-l}(|E_x| - 1)^l (6\eta\Delta + 1)^{6(n-l)}$$

n-tuples $(x_{Q_1}, \ldots, x_{Q_n}) \notin \mathcal{Q}$ for which event (E-3) happens. Denote them by $\mathcal{Q}'$. Over these tuples (not in Q), the probability that $f_0' = 0$ (given $d_0 = 0$ and $y_P \neq 0$) is bounded by

$$\frac{|\mathcal{Q}'|}{\left|E_x \setminus \{x_P\}\right|^n} \leq \sum_{l=0}^{n-1} \left(\frac{4(6\eta\Delta + 1)^6}{|E_x| - 1} \right)^{n-l} \leq \sum_{l=1}^{n} \left(\frac{4(6\eta\Delta + 1)^6}{\frac{1}{2}(\#E - 1) - 1} \right)^{l} =$$
$$\sum_{l=1}^{n} \left(\frac{1}{2} \frac{16(6\eta\Delta + 1)^6}{\#E - 3} \right)^{l} \leq \sum_{l=1}^{n} \left(\frac{1}{2} \right)^{l} \left(\frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2} \right)^{l}.$$

If $\frac{16(6\eta\Delta+1)^6}{p-2\sqrt{p}-2} < 1$, then the latter is smaller than $\frac{16(6\eta\Delta+1)^6}{p-2\sqrt{p}-2}$. In any case we get that this probability is bounded by

$$\frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2}.$$

We finally get that the probability that event (E-3) happens satisfies

$$\Pr[\text{(E-3)}] \leq \frac{|\mathcal{Q}|}{\left|E_x \setminus \{x_P\}\right|^n} + \frac{|\mathcal{Q}'|}{\left|E_x \setminus \{x_P\}\right|^n} < \frac{8^n(6\eta\Delta + 1)^{6n+3}}{(p - 2\sqrt{p} - 2)^n} + \frac{16(6\eta\Delta + 1)^6}{p - 2\sqrt{p} - 2}.$$

Notice that the probability that $Q_i = \pm P$ for some $1 \leq i \leq n$ is

$$\frac{2}{\#E - 1} \leq \frac{2}{p - 2\sqrt{p}} .$$

Thus, the probability that $Q_i = \pm P$ for any $1 \leq i \leq n$ is bounded by

$$\frac{2n}{p - 2\sqrt{p}} .$$

This concludes the first claim in the theorem.

Now suppose x_P has been recovered. In order to determine which of the two values $\pm\sqrt{x_P^3 + ax_P + b}$ is the correct y-coordinate of P, we run the consistency check, which is presented below, on both candidates. It is clear that the correct candidate will pass the test. If both candidates pass the consistency check then we cannot determine the point P. We analyze the probability of the event in which the incorrect candidate $-P = (x_P, -y_P)$ passes the test.

We consider how many Q_i lead the system to be consistent with both $\pm y_P$. Recall that

$$h_i + e_i = \left(\frac{y_{Q_i} - y_P}{x_{Q_i} - x_P}\right)^2 - x_P - x_{Q_i} = \frac{x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b - 2y_{Q_i}y_P}{(x_{Q_i} - x_P)^2} .$$

If $-P$ passes the test, then there exist $\bar{e}_i$ with $|\bar{e}_i| \leq \Delta$ such that $h_i = (P - Q_i)_x - \bar{e}_i$, for all $1 \leq i \leq n$. We therefore have

$$h_i + \bar{e}_i = \left(\frac{y_{Q_i} + y_P}{x_{Q_i} - x_P}\right)^2 - x_P - x_{Q_i} = \frac{x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b + 2y_{Q_i}y_P}{(x_{Q_i} - x_P)^2} .$$

Subtracting one from the other and multiplying by $(x_P - x_{Q_i})^2$ we get

$$(e_i - \bar{e}_i)(x_P - x_{Q_i})^2 = -4y_P y_{Q_i} .$$

Squaring both sides and rearranging results in

$$(e_i - \bar{e}_i)^2(x_P - x_{Q_i})^4 - 16y_P^2(x_{Q_i}^3 + ax_{Q_i} + b) \equiv 0 \pmod{p} .$$

This is a non-constant polynomial in x_{Q_i} of degree 4 and therefore for every $\bar{e}_i$ there are at most 4 values for x_{Q_i} that satisfy this equation. Since there are at most 2Δ possible values for each $\bar{e}_i$, and since we can form n such equations,[3] we conclude that the probability

[3]Notice that we can also form n equations from the values $h_{i'}$. For each i each solution x_{Q_i} should

that the point $(x_P, -y_P)$ passes the consistency check is bounded by

$$\frac{4^n(2\Delta)^n}{(|E_x|-1)^n} \leq \frac{(16\Delta)^n}{(p-2\sqrt{p}-2)^n}\,.$$

This concludes the proof.

Consistency Check – Filtering Impossible Secrets.

We introduce a test that takes a candidate P' for the secret point P, and determines whether P' is not the secret. That is, after running the test, P' is either guaranteed not to be P or it is potentially the secret point P. We give a bound on the probability that the outcome of the test is inconclusive, for $P' \neq P$ (it is clear that if $P' = P$ the test is inconclusive). Specifically, given the candidate for x_P from Theorem 7.1, one can test which value (if any) is the correct y-coordinate y_P. Moreover, one can test whether $y_P \neq 0$ or $P \neq \pm Q_i$.

Given a candidate $P' = (x_{P'}, y_{P'})$, the consistency check goes over the pairs $(Q, h = \mathrm{MSB}_k((P+Q)_x))$ and checks if these values are consistent with the problem's settings. That is, we use h to derive a candidate $\bar{e}$ for e, and check if $|\bar{e}| \leq \Delta$. Formally, using $h_0 = x_P - e_0$ we compute

$$\bar{e}_0 := x_{P'} - h_0 \mod p\,,$$

and check if $|\bar{e}_0| \leq \Delta$. If so then for every $1 \leq i \leq n$ using $h_i = \mathrm{MSB}_k((P+Q_i)_x)$ we compute

$$\bar{e}_i := \left(\frac{y_{P'} - y_Q}{x_{P'} - x_Q}\right)^2 - x_{P'} - x_Q - h_i \mod p\,,$$

and check if $|\bar{e}_i| \leq \Delta$. We do the same process with $h_{i'}$. If at any point this inequality does not hold, we can stop the test and determine that $P' \neq P$. Otherwise, P' passes the consistency check and is potentially the secret point P.

For completeness, we analyze the probability (over the samples Q_i) of the event in which a candidate $P' \neq P$ passes the consistency check. Hence, suppose that $P' = (x_{P'}, y_{P'})$ passed the consistency check.

Probability of $x_{P'} \neq x_P$. Given $h_i, h_{i'}$, from Section 7.1.1 above we have

$$h_i + h_{i'} = 2\left(\frac{x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + a x_P + 2b}{(x_P - x_{Q_i})^2}\right) - e_i - e_{i'}\,.$$

satisfy an additional equation $(e_{i'} - \bar{e}_{i'})(x_P - x_{Q_i})^2 = 4 y_P y_{Q_i}$. However, adding the two equations results in the condition $e_i + e_{i'} - \bar{e}_i - \bar{e}_{i'} = 0$. While this condition can be always satisfied (e.g. $\bar{e}_{i'} = e_i, \bar{e}_i = e_{i'}$), the probability it holds depends on the model for the oracle, i.e. how the terms $e_i, e_{i'}$ are generated.

Since P' passed the consistency check there exist $|\bar{e}_i|, |\bar{e}_{i'}| \leq \Delta$ such that

$$h_i + h_{i'} = 2\left(\frac{x_{P'}x_{Q_i}^2 + (a + x_{P'}^2)x_{Q_i} + ax_{P'} + 2b}{(x_{P'} - x_{Q_i})^2}\right) - \bar{e}_i - \bar{e}_{i'} .$$

Subtracting these two equations and multiplying by $(x_P - x_{Q_i})^2(x_{P'} - x_{Q_i})^2$ we get

$$\begin{aligned}
&(e_i + e_{i'} - \bar{e}_i - \bar{e}_{i'})(x_P - x_{Q_i})^2(x_{P'} - x_{Q_i})^2 = \\
&2\Big((x_P x_{Q_i}^2 + (a + x_P^2)x_{Q_i} + ax_P + 2b)(x_{P'} - x_{Q_i})^2 \\
&- (x_{P'}x_{Q_i}^2 + (a + x_{P'}^2)x_{Q_i} + ax_{P'} + 2b)(x_P - x_{Q_i})^2\Big) .
\end{aligned}$$

By rearranging we get a polynomial in x_{Q_i} of degree 4. By simple algebra one can check that this polynomial is identically zero if and only if $x_{P'} = x_P$ (thus $e_i + e_{i'} - \bar{e}_i - \bar{e}_{i'} = 0$). We assume $x_{P'} \neq x_P$. Therefore for every $\bar{e}_i, \bar{e}_{i'}$ there are at most 4 values for x_{Q_i} that satisfy this equation. Since there are $2\Delta + 1$ possible values for each $\bar{e}_i, \bar{e}_{i'}$ we conclude that the probability that $x_{P'} \neq x_P$ is bounded by

$$\frac{4^n(2\Delta + 1)^{2n}}{(|E_x| - 1)^n} \leq \frac{2^n(4\Delta + 2)^{2n}}{(p - 2\sqrt{p} - 2)^n} .$$

Probability of $x_{P'} = x_P$ and $y_{P'} \neq y_P$. The probability that $P' = (x_P, -y_P)$ passes the consistency check, is analyzed at the end of the proof of Theorem 7.1, and shown to be bounded by

$$\frac{4^n(2\Delta)^n}{(|E_x| - 1)^n} \leq \frac{(16\Delta)^n}{(p - 2\sqrt{p} - 2)^n} .$$

□

Remark 7.4. *In the case in which the value $d_0 \neq 0$, the recovered value $e := f_0/f_0' \neq e_0$, and therefore $x_{P'} := h + e \neq x_P$. Running the consistency check on P' might reveal that indeed $P' \neq P$. One can derive from equations (7.10)-(7.13) other candidates for e_0 and subsequently candidates for x_P, and apply the consistency check on them. If none of these candidates pass the consistency check, then one can test P' where $y_{P'} = 0$ and $P' = \pm Q_i$. We analyze the probability that there exists $P' \neq P$ that is consistent with all $2n + 1$ samples.*

We use the analysis above which shows that the probability that a candidate P' with $x_{P'} \neq x_P$ passes the test with the $2n$ equations is bounded by

$$\frac{(4\Delta + 2)^{2n}}{(|E_x| - 1)^n} \leq \frac{2^n(4\Delta + 2)^{2n}}{(p - 2\sqrt{p} - 2)^n} .$$

We also have $x_{P'} - \bar{e}_0 = h_0 = x_P - e_0$, so $x_{P'} = x_P - e_0 + \bar{e}_0$ can take 2Δ values. Thus, the probability that any P' with $x_{P'} \neq x_P$ passes the consistency check is bounded by

$$\frac{2^{n+1}\Delta(4\Delta+2)^{2n}}{(p-2\sqrt{p}-2)^n}\,.$$

With the above bound for $y_{P'} \neq -y_P$ we get that the probability that there exists $P' \neq P$ that passes the consistency check is bounded by

$$\frac{2^{n+1}\Delta(4\Delta+2)^{2n}}{(p-2\sqrt{p}-2)^n} + \frac{(16\Delta)^n}{(p-2\sqrt{p}-2)^n}\,.$$

A corollary of this result is that for $k > \frac{5}{6}\log(p)$ the success probability becomes nontrivial, and so we get a polynomial-time solution to EC-HNP$_x$(DH), as we now state formally. We consider two different SVP approximation algorithms to show the influence of ϵ on the running time and the minimum allowed value for p.

Corollary 7.5. *Fix $0 < \delta \leq 3\epsilon < 1/2$. Let $n_0 = \lceil \frac{1}{6\epsilon} \rceil$, p be an m-bit prime, E be an elliptic curve over $\mathbb{F}_p$ and $k > (5/6+\epsilon)m$. There exist randomised algorithms A_i, for $i = 1, 2$, that solve EC-HNP$_x$(DH) (with MSB$_k$ and a generator R) for $m \geq m_i$, with probability at least $1 - p^{-\delta}$ over the choices of $x_{Q_1}, \ldots, x_{Q_{n_0}}$ where*

$$m_1 = \lceil c_1 \epsilon^{-1} \log \epsilon^{-1} \rceil \quad \text{and} \quad m_2 = \lceil c_2 \epsilon^{-2} \frac{(\log\log \epsilon^{-1})^2}{\log \epsilon^{-1}} \rceil\,,$$

for some absolute effectively computable constants c_1, c_2, and their running time is T_i where

$$T_1 = (2^{\epsilon^{-1}} m)^{O(1)} \quad \text{and} \quad T_2 = (\epsilon^{-1} m)^{O(1)}\,.$$

Proof. Consider the bounds on $\mathcal{P}_1$ and $\mathcal{P}_2$ in Theorem 7.1. One needs $1 - \mathcal{P}_1 - \mathcal{P}_2 \geq 1 - p^{-\delta}$, therefore $\mathcal{P}_1 + \mathcal{P}_2 \leq p^{-\delta}$, for the claim to hold. As $\mathcal{P}_2$ is smaller than the first bound on $\mathcal{P}_1$ in Theorem 7.1 we get that $\mathcal{P}_1 + \mathcal{P}_2$ is bounded by

$$2\frac{8^n(6\eta\Delta+1)^{6n+3}}{(p-2\sqrt{p}-2)^n} + \frac{16(6\eta\Delta+1)^6}{p-2\sqrt{p}-2} + \frac{2n+3}{p-2\sqrt{p}}\,. \tag{7.16}$$

It is sufficient to bound the latter by $p^{-\delta}$.

Consider the third term in (7.16). For the claim to hold, one needs

$$\frac{2n_0+3}{p-2\sqrt{p}} < \frac{1}{p^\delta}\,,$$

from which it is easy to derive the minimal p (thus the minimal bit size m of p) for the

condition to hold. We therefore let δ' such that $p^{-\delta'} = p^{-\delta} - \frac{2n_0+3}{p-2\sqrt{p}}$ (assuming the later is positive) and bound each of the other terms in (7.16) by $\frac{p^{-\delta'}}{2}$. Notice that $\delta' > \delta$.

Plugging $p = 2^{m+O(1)}$ and $\Delta = 2^{m-k+O(1)}$ in the first term (7.16), and since $k > (5/6+\epsilon)m$, we have

$$\begin{aligned}\frac{2\cdot 8^n(6\eta\Delta+1)^{6n+3}}{(p-2\sqrt{p}-2)^n} &= \frac{2^{3n+1}(2^{O(1)}\eta 2^{m-k+O(1)}+1)^{6n+3}}{(2^{m+O(1)}-2^{m/2+O(1)}-2)^n} = \eta^{6n+3}2^{(6n+3)(m-k+O(1))-(m+O(1))n} \\ &\le \eta^{6n+3}2^{(6n+3)(m/6-m\epsilon+O(1))-(m+O(1))n} = 2^{(6n+3)(\log\eta-m\epsilon)+m/2+O(n)}\,.\end{aligned}$$

The latter is smaller than $\frac{p^{-\delta'}}{2} = 2^{-\delta'(m-1+O(1))}$ if $(6n+3)(\log\eta-\epsilon m)+m/2+O(n) \le -\delta'(m+O(1))$, which simplifies to (for some sufficiently large absolute constant C_0)

$$(6n+3)(\epsilon - m^{-1}(\log\eta + C_0)) \ge \delta' + \frac{1}{2} > \delta + \frac{1}{2}\,. \tag{7.17}$$

Using $3\epsilon \ge \delta$ and $n \ge n_0$, it is easy to verify that (for a sufficiently large absolute constant C_1)

$$m > \epsilon^{-1}(2\log\eta + C_1) \tag{7.18}$$

implies (7.17).

Similarly, to show that the second term in (7.16) is bounded by $\frac{p^{-\delta'}}{2}$ one gets the condition (for some sufficiently large absolute constant C_2)

$$6(\epsilon - m^{-1}(\log\eta + C_3)) \ge \delta' > \delta\,,$$

which can be shown to hold when (for a sufficiently large absolute constant C_3)

$$m > (6\log\eta + C_3)(6\epsilon-\delta)^{-1}\,.$$

The latter is implied by (7.17), therefore by (7.18), provided C_0 is large enough.

For A_1 we apply the 1-SVP algorithm (with running time $\tilde{O}(2^{2d})$) of Micciancio and Voulgaris [64] to a lattice of dimension $d = 3n_0+3$, which gives $\eta = 2\sqrt{3n_0+1}$. For A_2, we use the $2^{O(d(\log\log d)^2/\log d)}$-SVP algorithm (with running time $\tilde{O}(d)$) of Schnorr [76] for the dimension $d = 3n_0+3$, which gives $\eta = 2^{n_0+2}\sqrt{3n_0+1}$. Using $n_0 = \lceil\frac{1}{6\epsilon}\rceil$, the bounds m_i follow. □

Hardcore Bits for Elliptic Curve Diffie–Hellman

As a consequence, we get a hardcore function for the elliptic curve Diffie–Hellman problem and the following bit security result for elliptic curve Diffie–Hellman key exchange.

Theorem 7.6. *Fix $0 < \delta \leq 3\epsilon < 1/2$. Let p be an m-bit prime, E be an elliptic curve over $\mathbb{F}_p$, a point $P \in E \setminus \{O\}$ of order at least $p/poly(\log(p))$ and $k > (5/6 + \epsilon)m$. Given an efficient algorithm to compute $MSB_k\left(([ab]P)_x\right)$ from $[a]P$ and $[b]P$, there exists a polynomial-time algorithm that computes $[ab]P$ with probability at least $1 - p^\delta$.*

Least Significant Bits of Elliptic Curve Diffie–Hellman Keys

Bit security results are usually symmetric for the most and least significant bits. The aim of this part is to sketch how one would provide similar results for the lower 5/6 bits of elliptic curve Diffie–Hellman keys. We consider an elliptic curve hidden number problem with the oracle $\mathcal{O}(t) = \mathrm{LSB}_k((P + [t]R)_x)$.

Recall that as we allow k to take any (positive) real value, we define LSB_k by $\mathrm{LSB}_k(x) := x \pmod{\lceil 2^k \rceil}$. In other words, $\mathrm{LSB}_k(x)$ gives $x \bmod l$ for $2 \leq l = \lceil 2^k \rceil \leq p$, not necessarily a power of 2.

Let $h = \mathrm{LSB}_k((P+Q)_x) = (P+Q)_x \mod l = (s_{P+Q}^2 - x_P - x_Q - qp) - le$ for some q and $|e| < \frac{p}{2l} \leq \frac{p}{2^{k+1}}$. For $u = l^{-1} \in \mathbb{Z}_p^*$ we have (where the operations are in $\mathbb{F}_p$)

$$\begin{aligned}\overline{h} := hu &= \left(\left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q - qp - le\right) u \\ &= u\left(\left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q\right) - q'p - e \equiv u\left(\left(\frac{y_P - y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q\right) - e\,.\end{aligned}$$

Now let $h_0 = \mathrm{LSB}_k(x_P) = x_P - le_0$ and $h' = \mathrm{LSB}_k((P-Q)_x) = (P-Q)_x \mod l = (s_{P-Q}^2 - x_P - x_Q - rp) - le'$ for some r and $|e_0|, |e'| < \frac{p}{2l} \leq \frac{p}{2^{k+1}}$. Then

$$\overline{h'} := h'u \equiv u\left(\left(\frac{y_P + y_Q}{x_P - x_Q}\right)^2 - x_P - x_Q\right) - e' \pmod{p}\,.$$

Letting $\widetilde{h} = \overline{h} + \overline{h'}$ and $\widetilde{e} = e + e'$ and plugging $x_P = h_0 + le_0$ in (7.3) above we get

$$\begin{aligned}\widetilde{h} + \widetilde{e} &= u\left((P+Q)_x + (P-Q)_x\right) \\ &\equiv 2u\left(\frac{x_Q(h_0 + le_0)^2 + (a + x_Q^2)(h_0 + le_0) + ax_Q + 2b}{(h_0 + le_0 - x_Q)^2}\right) \pmod{p}\,.\end{aligned}$$

Multiplying by $(h_0 + le_0 - x_Q)^2$ results in a bivariate polynomial in $e_0, \widetilde{e}$ of degree 3, similar to (7.4) above. We expect to get similar results to those presented above.

The y-coordinate A natural question is whether a similar strong bit security result can be shown for the y-coordinate of the elliptic curve Diffie–Hellman key. Unfortunately, the trick presented in this paper, using 2 correlated equations to eliminate one variable, seems out of reach when one works with the y-coordinate. In this case, one would like to eliminate the x-coordinate x_P, but it seems to be very "hardcoded" into the addition formula of the Weierstrass representation. We remark that one can still get some results using the approaches described in the beginning of the section, but they ought to be very weak results.

7.1.3 Extension Fields

The main results in the previous section address the prime-field elliptic curve hidden number problem. In this section we address the case of elliptic curve points that lie in a non-prime extension field.

Recall that the field $\mathbb{F}_q = \mathbb{F}_{p^d}$ is a d-dimensional vector space over $\mathbb{F}_p$. We fix a basis $\{\mathbf{b_1}, \ldots, \mathbf{b_d}\}$ for $\mathbb{F}_q$, and represent points $x \in \mathbb{F}_q$ with respect to that basis: for $\mathbf{x} = \sum_{i=1}^{d} x^i \mathbf{b_i}$ we write $\mathbf{x} = (x^1, \ldots, x^d)$. We consider $E(\mathbb{F}_q)$, the group of elliptic curve points over $\mathbb{F}_q$. As in previous sections we are interested in EC-HNP(DH) where the oracle provides partial information on one component of one of the coordinates. For example, one gets $\mathrm{MSB}_k\left((P + [t]R)^i_x\right)$ for some component $1 \leq i \leq d$.

A natural and important question is whether the ability to recover one component – that is, when $k = \log(p)$ in the example above – allows to recover the entire secret point. This question is natural as extension fields provide additional algebraic structure that can be exploited, and it is important since it potentially allows to prove stronger results than in prime-field cases as one component represents only a fraction of $1/d$ of all bits.

The bit security of elliptic curve Diffie–Hellman keys over extension fields was studied by Jao, Jetchev and Venkatesan [48]. They consider the following hidden number problem for elliptic curves, which they call *multiplier elliptic curve hidden number problem*:[4] Given an oracle $\mathcal{O}$ that computes a single component of the x-coordinate of the map $r \to [r]P$, that is $\mathcal{O}(r) = ([r]P)^i_x$, recover the point P. They present a heuristic algorithm to this problem, except for the case $d = 2$ where a complete proof is given (they also provide a proof for the case of $d = 3$, but it is incomplete (see [77] and below)). This algorithm is polynomial in $\log(p)$ but not in d, and therefore suits problems where one fixes the degree d and let $\log(p)$ grow. That is, for extension fields $\mathbb{F}_{p^d}$ of a constant degree.

In this section we generalise the method that was suggested in [48, Section 3.3]. We then show that the approach taken for EC-HNP(DH) in prime fields in the previous

[4]This is in fact the exponent version of HNP that we introduced in Chapter 4.

chapter can also be used for the extension-field case to obviate the heuristics in the previous work, and thus provide a complete bit security result for any elliptic curve over any field of a constant extension degree. This solution applies to both the x and the y coordinates of the elliptic curve Diffie–Hellman key.

The following section presents a general method to solve (non-linear) hidden number problems over non-prime fields, where one component is given. It is followed by the results of [48] and the new improved bit security result given here.

A General Method

The method presented in this section applies for both variants of the elliptic curve hidden number problem, i.e. where the operation is elliptic curve addition or point multiplication. To make this section more accessible, we first give a few concrete approaches to the problem. Fix a point $Q \in E$, let $P = (\mathbf{x}, \mathbf{y}) \in E$ be a variable, and let $\psi \in \{x, y\}$. The following facts give rise to three approaches.

natural: Both point addition $(P + Q)_\psi$ and point multiplication $([r]P)_\psi$ can be represented by rational functions in $\mathbf{x}, \mathbf{y}$; the former has small degree, and the degree of the latter is polynomial in r.

JJV: Point multiplication $([r]P)_x$ can be represented by a rational function only in $\mathbf{x}$ using the elliptic curve division polynomials, where its degree is polynomial in r.[5]

our: The function $(P + Q)_x + (P - Q)_x$ and the function $(P + Q)_y - (P - Q)_y$ can be represented by small-degree rational functions in $\mathbf{x}$.

To unify these different approaches, we let an unknown $P \in E$ and a known $R \in E$ be points and consider a family of functions $f_t : E \to \mathbb{F}_q$. For each approach we have

natural: $f_t(P) = (P + [t]R)_\psi$ or $f_t(P) = ([t]P)_\psi$.

JJV: $f_t(P) = ([t]P)_x$.

our: $f_t(P) = (P + [t]R)_x + (P + [-t]R)_x$ or $f_t(P) = (P + [t]R)_y - (P + [-t]R)_y$.

We also consider an oracle $\mathcal{O} : [0, \#E - 1] \to \mathbb{F}_p$ that outputs a single component of $f_t(P)$, that is $\mathcal{O}(t) = \mathcal{O}_P(t) = f_t^i(P)$. It is clear that in these three approaches given an oracle for a component of the Diffie–Hellman key, one can form $\mathcal{O}$.

[5] Also $([r]P)_y$ can be represented by a rational function using the division polynomials, in variables $\mathbf{x}, \mathbf{y}$.

Suppose we can represent $f(P) = f_t(P)$ as a rational function, that is

$$f(P) = \frac{R_1(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)}{R_2(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)},$$

where R_1, R_2 are polynomials in $\mathbb{F}_q[z_1, \ldots, z_{2d}]$. Rewrite

$$\frac{R_1(z_1, \ldots, z_{2d})}{R_2(z_1, \ldots, z_{2d})} = \frac{R_1^1\mathbf{b}_1 + \ldots + R_1^d\mathbf{b}_d}{R_2^1\mathbf{b}_1 + \ldots + R_2^d\mathbf{b}_d},$$

where for $1 \leq j \leq d$ each polynomial $R_1^j(z_1, \ldots, z_{2d}), R_2^j(z_1, \ldots, z_{2d})$ has coefficients in $\mathbb{F}_p$. We "rationalise" the denominator (by multiplying the numerator and denominator by a polynomial such that the denominator belongs to $\mathbb{F}_p[z_1, \ldots, z_{2d}]$) to express

$$\frac{R_1(z_1, \ldots, z_{2d})}{R_2(z_1, \ldots, z_{2d})} = r^1(z_1, \ldots, z_{2d})\mathbf{b}_1 + \ldots + r^d(z_1, \ldots, z_{2d})\mathbf{b}_d,$$

where r^j are rational functions with coefficients in $\mathbb{F}_p$. We assume to have access to component i of $f_t(P)$, that is, we have

$$\mathcal{O}(t) = f_t^i(P) = r_t^i(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d) = \frac{r_{t,1}^i(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)}{r_{t,2}^i(x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)}.$$

Multiplying by $r_{t,2}^i$ and rearranging we get that the following polynomials

$$g_t(z_1, \ldots, z_{2d}) := r_{t,1}^i(z_1, \ldots, z_{2d}) - r_{t,2}^i(z_1, \ldots, z_{2d}) f_t^i(P)$$

are polynomials in $\mathbb{F}_p[z_1, \ldots, z_{2d}]$, for which the point $P = (x_P^1, \ldots, x_P^d, y_P^1, \ldots, y_P^d)$ is a simultaneous solution, and so if one can find the solutions to this system, then one can recover P.

Notice that in the rationalisation step one multiplies the denominator by all of its $d-1$ conjugates (to admit the norm). Thus, if the denominator's degree is l, one would multiply in general by a polynomial of degree $l(d-1)$. Hence, each of the polynomials g_t is in general of degree at least d. Furthermore, we need at least $2d$ equations to solve a system with $2d$ variables. Elimination techniques can then be used to reduce the number of variables. This elimination process is polynomial in terms of the ground field, that is in $\log(p)$, but not necessarily in the degrees. The current algorithms to solve systems of $2d$ equations, all of degree at least d, do not run in time polynomial in d. Therefore, this method is efficient if one fixes the extension degree d, as then the affect of d on the elimination process running time is constant.

Previous Results

The JJV approach described above is the one taken by Jao, Jetchev and Venkatesan [48], namely they use the elliptic curve division polynomials to represent point multiplication, which allows them to represent the x-coordinate of the point multiplication mapping as a function in $\mathbf{x}$, and so the system has d variables (see [85, III, Ex.3.7] for more details). Doing so imposes a very strong constraint: the multiplication by t map has degree t^2 ([85, III, Ex.3.7(e)]), and so writing down the explicit algebraic expressions for these mappings can be done efficiently only for small values of t. As a consequence, it is not clear that by considering only small multipliers t, the multiplier elliptic curve hidden number problem has a unique solution, or a small set of solutions. For comparison, it is easy to show that restricting to small multipliers in $\mathbb{F}_p$-HNP yields exponentially many solutions.

As a result, they give a precise statement for degrees $d = 2$ [48, Propositions 3.1] (using the point doubling formula and one component of the secret, one generates a non-zero low-degree polynomial in the other component) and $d = 3$ [48, Propositions 3.2]. The latter case uses the resultant of two bivariate polynomials to get a single constant-degree (univariate) polynomial. However, it is not shown that this resultant cannot be the zero polynomial, and so it is not clear that one can efficiently solve the system. For the general constant-degree case [48, Section 3.3], only a heuristic result is given.

New Results

As mentioned above, given a component of the x-coordinate, one can consider the second-degree rational function given by $(P+Q)_x + (P-Q)_x$:

$$(P+Q)_x + (P-Q)_x = 2\left(\frac{\mathbf{x}_Q\mathbf{x}_P^2 + (\mathbf{a}+\mathbf{x}_Q^2)\mathbf{x}_P + \mathbf{a}\mathbf{x}_Q + 2\mathbf{b}}{(\mathbf{x}_P - \mathbf{x}_Q)^2}\right) = \frac{R_1(x_P^1, \ldots, x_P^d)}{R_2(x_P^1, \ldots, x_P^d)},$$

and by the aforementioned method one defines

$$g_Q(x^1, \ldots, x^d) := r_{Q,1}^i(x^1, \ldots, x^d) - r_{Q,2}^i(x^1, \ldots, x^d)\left((P+Q)_x^i + (P-Q)_x^i\right),$$

where $r_{Q,1}^i, r_{Q,2}^i$, and so g_Q, are polynomials of degree at most $2d$.

Given a component of the y-coordinate, one can consider the third-degree rational function given by $(P+Q)_y - (P-Q)_y$:

$$(P+Q)_y - (P-Q)_y = 2\mathbf{y}_Q\left(\frac{\mathbf{x}_P^3 + 3\mathbf{x}_Q\mathbf{x}_P^2 + 3\mathbf{a}\mathbf{x}_P + \mathbf{a}\mathbf{x}_Q + 4\mathbf{b}}{(\mathbf{x}_P - \mathbf{x}_Q)^3}\right),$$

where now one defines

$$g_Q(x^1,\ldots,x^d) := r^i_{Q,1}(x^1,\ldots,x^d) - r^i_{Q,2}(x^1,\ldots,x^d)\left((P+Q)_y - (P-Q)_y\right),$$

where $r^i_{Q,1}, r^i_{Q,2}, g_Q$, are polynomials of degree at most $3d$.

Notice that in both cases g_Q is a polynomial in $\mathbf{x}$. The benefit of this approach is that the polynomials g_Q are of low degree for any Q, and thus the points Q are not restricted to any (short) interval. Standard arguments, like root counting (as done for the prime-field case in Section 7.1), can be used to show that for uniform and independent Q's, a sufficiently large system $\{g_Q\}$ is expected to have a unique (simultaneous) root. This leads to the following results.

Proposition 7.7. *Let E be an elliptic curve over an extension field $\mathbb{F}_{p^d}$. There exists an algorithm, polynomial in $\log(p)$, that solves EC-HNP(DH) given an oracle that outputs a complete component of either the x or y coordinates.*

Corollary 7.8. *For an elliptic curve defined over a constant-degree extension field, computing a single component of the Diffie–Hellman key (for either the x or y coordinates) is as hard as computing the entire key.*

We remark that besides the bit security result, the approach presented here gives the most efficient algorithm among the three aproaches presented above. Addressing elimination problems, the polynomials' degree, the number of variables in the system and the number of solutions to the polynomial system play a main role in the complexity of this task. In all of these aspects our approach is never worse than the other two. See [77, Section 6.2.3] for a few examples.

7.2 Algebraic Torus Hidden Number Problem

In this section we study the *algebraic torus hidden number problem*. Recall that the cyclotomic subgroup $G_{q,n}$ of $\mathbb{F}^*_{q^n}$ is isomorphic to the algebraic torus $\mathbb{T}_n(\mathbb{F}_q)$, and that if $\mathbb{T}_n$ is rational then it has an explicit rational parameterization. The latter allows us to compactly represent almost all elements of $G_{q,n}$ in $\mathbb{A}^{\varphi(n)}(\mathbb{F}_q)$. Recall that a point $g \in \mathbb{T}_n$ is regular if it can be represented in $\mathbb{A}^{\varphi(n)}$ using the birational equivalence. In the algebraic torus hidden number problem the group is the points on the torus $\mathbb{T}_n$ represented in $\mathbb{A}^{\varphi(n)}$, and the operation is given by the partial group law $\star$ on $\mathbb{A}^{\varphi(n)}$. One can consider a Diffie–Hellman variant of this problem, however we solve a stronger variant where no "Diffie–Hellman access" is given. We specifically consider the case where the oracle gives some most significant bits (for elements of an extension field, the oracle gives the most

significant bits of one component in the ground field). We formulate the problem as follows.

> $\mathbb{T}_n$-HNP: Fix a prime p, an integer n such that the torus $\mathbb{T}_n$ has an explicit rational parameterization, an integer $1 \leq i \leq n$, a regular point $h \in \mathbb{A}^m(\mathbb{F}_p)$, where $m := \varphi(n)$, and a positive number k. Let $t_1, \ldots, t_d \in \mathbb{A}^m(\mathbb{F}_p)$ be regular points and let $s \in \mathbb{A}^m(\mathbb{F}_p)$ be an unknown regular point. Recover s given the d pairs $(t_i, \mathrm{MSB}_k((s \star t)_i)$.

There is no literature about this problem nor any known bit security results for the $\mathbb{T}_2, \mathbb{T}_6$-cryptosystems. The main interest in the literature are for $n = 2, 6$, that is the tori $\mathbb{T}_2$ and $\mathbb{T}_6$, and so we focus on them. For $n = 2$ we show that there exists a polynomial-time solution to $\mathbb{T}_2$-HNP for $k > \frac{2}{3}\log(p)$. That is, if for some fixed $\epsilon > 0$ one takes $k > (\frac{2}{3} + \epsilon)\log(p)$ for sufficiently large p. This result follows almost immediately from the solution to MIHNP, once the equations are developed. For $n = 6$ we develop the equations, and conclude that in the most general representation of the fields involved these techniques does not lead to any significant results. One can show that for $k = \log(p)$, i.e. a complete component is given, it is possible to solve $\mathbb{T}_n$-HNP. The following case of most interest is for $n = 30$, but as the results for $\mathbb{T}_6$ are very weak, it suggested that also the results for $\mathbb{T}_{30}$ would be very weak, and so we do not explore this case.

7.2.1 Algebraic Torus $\mathbb{T}_2$

With notation as above we have $n = 2$ and so $m = 1$. Therefore, the compact representation is given in $\mathbb{A}^1(\mathbb{F}_p)$. Recall that we let $\mathbb{F}_{p^2} = \mathbb{F}_p(\theta)$, where $\theta^2 + A\theta + B = 0$, with $A, B \in \mathbb{F}_p$ such that $x^2 + Ax + B$ is irreducible over $\mathbb{F}_p$. Recall that for any two (regular) points $a, b \in \mathbb{A}^1$, the partial group law on $\mathbb{A}^1$ is given by $a \star b = \dfrac{ab - B}{a + b - A}$.

Consider $\mathbb{T}_2$-HNP, and write

$$\mathrm{MSB}_k(s \star t) \equiv \frac{st - B}{s + t - A} - e \pmod{p},$$

where e, as a function of x, is a random integral value satisfying $|e| \leq \frac{p}{2^{k+1}}$.

We assume without loss of generality to have $\mathrm{MSB}_k(s)$: concerned with the bit security of the $\mathbb{T}_n$-cryptosystem, without the abstraction of $\mathbb{T}_n$-HNP, it is clear that one can use the oracle to obtain partial knowledge about g^{ab}. Furthermore, one can define a new secret $s' := s \star t_1$, and interpret the other samples as $(s \star t_1) \star (t_1^{-1} \star t_i) = s' \star t'$.

Thus, we have

$$h_0 := \mathrm{MSB}_k(s) \equiv s - e_0 \pmod{p}.$$

Let $h := \mathrm{MSB}_k(s \star t)$, and write

$$(h+e)(s+t-A) \equiv st - B \pmod{p}\,.$$

Therefore

$$(h+e)(h_0+e_0+t-A) \equiv (h_0+e_0)t - B \pmod{p}\,.$$

Hence, the function

$$F(X,Y) = XY + (h-t)X + (h_0+t-A)Y + [h(h_0+t-A) - h_0 t + B]$$

satisfies $F(e_0, e) \equiv 0 \pmod{p}$.

Repeating with different t_i leads to polynomials of the form

$$F_i(X,Y) = XY + B_{1,i}X + B_{i,i}Y + A_i\,,$$

that satisfy $F(e_0, e) \equiv 0 \pmod{p}$.

One would like to use the approach developed in the previous sections to prove that it is possible to recover the roots with probability that depends on k. Luckily, such a proof already exists for polynomials of the general form

$$F_i(X,Y) = C_i XY + B_{1,i}X + B_{i,i}Y + A_i\,,$$

for specific coefficient values, namely the modular inversion hidden number problem, with $k > \frac{2}{3}\log(p)$, given that the multipliers t_i are chosen uniformly at random in $\mathbb{A}^1$.

The proof follows the steps presented above. Specifically, the algorithmic part follows exactly the same procedure as in [61, Theorem 1], with the change of coefficients

$$A_i \equiv h_i(h_0+t_i-A) - h_0 t_i + B \pmod{p}\,, \qquad B_{i,i} \equiv h_0 + t_i - A\,, \pmod{p}$$
$$B_{1,i} \equiv h_i - t_i \pmod{p}\,, \qquad C_i = 1\,.$$

In the part of the success probability analysis one needs to construct a different family of polynomials, which leads to some exceptional set for the secret values. Once this is done, and this exceptional set is shown to be of small cardinality, the rest of the proof follows the same arguments of [61, Theorem 1]. We now turn to close this gap.

From Eq. (12) in [61] we have

$$B_{1,i}d_1 + B_{i,i}d_i + C_i d_{1,i} \equiv 0 \pmod{p}\,.$$

In other words,

$$(h_i - t_i)d_1 + (h_0 + t_i - A)d_i + d_{1,i} \equiv 0 \pmod{p},$$

or

$$\left(\frac{st_i - B}{s + t_i - A} - e_i - t_i\right) d_1 + (s - e_0 + t_i - A)d_i + d_{1,i} \equiv 0 \pmod{p}.$$

Multiplying by $s+t_i-A$ and rearranging, we get the following, as a quadratic polynomial in t_i,

$$U_i t_i^2 + V_i t_i + W_i \equiv 0 \pmod{p}, \tag{7.19}$$

where

$$\begin{aligned}
U_i &\equiv d_i - d_1 \pmod{p}, \\
V_i &\equiv (A - e_i)d_1 + (2s - e_0 - 2A)d_i + d_{1,i} \pmod{p}, \text{ and} \\
W_i &\equiv (-B - e_i(s - A))d_1 + (s - A)(s - e_0 - A)d_i + (s - A)d_{1,i} \pmod{p}.
\end{aligned}$$

Similar to Claim 7.3 above, we want to analyse when the left hand side of (7.19) is the constant zero, to derive some exceptional values for s. We see that if $U_i \equiv 0$ then $d_1 \equiv d_i$. Plugging to $V_i \equiv 0$ gives the relation $d_{1,i} \equiv (A+e_0+e_i-2s)d_i$. Finally, plugging these relations to $W_i \equiv 0$ gives $(s^2 - As + B)d_{1,i} \equiv 0 \pmod{p}$.

Notice that if $s^2 - As + B \equiv 0$ then $(-s)^2 + A(-s) + B = s^2 - As + B \equiv 0 \pmod{p}$, which violates the fact that $x^2 + As + B$ is irreducible over $\mathbb{F}_p$. Thus, we get that $d_{1,i} \equiv 0$, which implies $d_1 \equiv d_i \equiv d_{1,i} \equiv 0 \pmod{p}$. As all of these values are smaller than p, we get an equality (over the integers). This implies that there are no exceptional values for s. The rest of the proof follows as in [61]: the probability the algorithm fails depends on the probability that some t_i satisfy (7.19), and vanishes asymptotically for $k > 2\log(p)/3$.

Hardcore Bits for $\mathbb{T}_2$ Diffie–Hellman

We see that one can solve $\mathbb{T}_2$-HNP in polynomial time for $k > (2/3 + \epsilon)\log(p)$. As a consequence, we get a hardcore function for the Diffie–Hellman problem on the algebraic torus $\mathbb{T}_2$ and the following bit security result for $\mathbb{T}_2$ Diffie–Hellman key exchange.

Theorem 7.9. *Fix $0 < \delta \leq \epsilon < 1/2$. Let p be an m-bit prime, a regular point $g \in \mathbb{A}^1(\mathbb{F}_p)$ of "order"[6] at least $p/poly(\log(p))$ and $k > (2/3 + \epsilon)m$. Given an efficient algorithm to compute $MSB_k(g^{ab})$ from g^a and g^b, there exists a polynomial-time algorithm that computes g^{ab} with probability at least $1 - p^{\delta}$.*

[6]See Remark 7.2 and footnote 2; the order can be thought of as the order of $\psi(g) \in \mathbb{T}_2(\mathbb{F}_p)$.

Using the Torus Representation

We briefly consider an alternative approach. Instead of working in $\mathbb{A}^1$, we use the mapping $\psi : \mathbb{A}^1 \to \mathbb{T}_2$ (and represent the image in $\mathbb{F}_{p^2}$)

$$\psi(s \star t) = \frac{(s \star t)^2 - B}{(s \star t)^2 - (s \star t)A + B} + \frac{2(s \star t) - A}{(s \star t)^2 - (s \star t)A + B}\theta\,.$$

Since we have an approximation of $s \star t$, namely $h = s \star t - e$, thus

$$\psi(s \star t) = \frac{(h+e)^2 - B}{(h+e)^2 - (h+e)A + B} + \frac{2(h+e) - A}{(h+e)^2 - (h+e)A + B}\theta\,. \tag{7.20}$$

On the other hand, using $h_0 = s - e_0$, we have

$$\begin{aligned}\psi(s)\psi(t) &= \psi(h_0 + e_0)\psi(t) \\ &= \left(\frac{(h_0+e_0)^2 - B}{(h_0+e_0)^2 - (h_0+e_0)A + B} + \frac{2(h_0+e_0) - A}{(h_0+e_0)^2 - (h_0+e_0)A + B}\theta\right) \cdot \\ &\quad \left(\frac{t^2 - B}{t^2 - tA + B} + \frac{2t - A}{t^2 - tA + B}\theta\right).\end{aligned} \tag{7.21}$$

Since ψ is a birational isomorphism $\psi(s)\psi(t) = \psi(s \star t)$, so one can equate the right-hand sides of (7.20) and (7.21). Computing the product on the right hand side of (7.21) and clearing denominators will result with the relation

$$F(e_0, e) = F_1(e_0, e) + F_2(e_0, e)\theta = 0\,,$$

where $F \in \mathbb{F}_{p^2}[X, Y]$ (so the equality is in $\mathbb{F}_{p^2}$) and $F_1, F_2 \in \mathbb{F}_p[X, Y]$ are some polynomials of the form

$$\begin{aligned}F_1(X,Y) &= aX^2Y^2 + b_1X^2Y + b_2XY^2 + c_1X^2 + c_2XY + c_3Y^2 + d_1X + d_2Y + f\,, \\ F_2(X,Y) &= \bar{a}X^2Y^2 + \bar{b}_1X^2Y + \bar{b}_2XY^2 + \bar{c}_1X^2 + \bar{c}_2XY + \bar{c}_3Y^2 + \bar{d}_1X + \bar{d}_2Y + \bar{f}\,.\end{aligned}$$

Applying the heuristic arguments above, for a single equation one expects to find a root for $|e_0|, |e| \leq \frac{p}{2^{k+1}}$ and $k > \frac{14}{15}\log(p)$. Now, since we have 2 equations, we apply the analysis from above to conclude that a root could be found for $k > \frac{14-1}{15}\log(p) = \frac{13}{15}\log(p)$. Thus, taking this approach would not lead to a better result than the one above.

7.2.2 Algebraic Torus $\mathbb{T}_6$

With notation as above we have $n = 6$ and so $m = 2$. Therefore, the compact representation is given in $\mathbb{A}^2(\mathbb{F}_p)$. We represent $\mathbb{F}_{p^6} = \mathbb{F}_{p^3}(\theta)$ for $\theta \in \mathbb{F}_{p^2}$, as we can therefore use the result from $\mathbb{T}_2(\mathbb{F}_{p^3})$. Let $\{\alpha_1, \alpha_2, \alpha_3\}$ be a basis for $\mathbb{F}_{p^3}$.

In order to study $\mathbb{T}_6$-HNP we study the partial group law $\star$ in $\mathbb{A}^2$. Recall that the birational map $\rho : \mathbb{T}_6 \to \mathbb{A}^2$ is given by $\rho = p_U \circ \rho_2$, and its inverse $\psi : \mathbb{A}^2 \to \mathbb{T}_6$ by $\psi = \psi_2 \circ p_U^{-1}$. Recall that for two (regular) points $a = (a_1, a_2), b = (b_1, b_2) \in \mathbb{A}^2$, the partial group law in $\mathbb{A}^2$ can be computed by $\rho\left(\psi(a)\psi(b)\right)$ or by

$$a \star b = p_U\left(p_U^{-1}(a) \star_2 p_U^{-1}(b)\right) = p_U\left(\frac{p_U^{-1}(a)p_U^{-1}(b) - B}{p_U^{-1}(a) + p_U^{-1}(b) - A}\right).$$

We consider the latter. We have

$$\begin{aligned} p_U^{-1}(a_1, a_2) &= P + \frac{g(a_1, a_2)}{h(a_1, a_2)}(1, a_1, a_2) \\ &= \left(\frac{g(a_1, a_2)}{h(a_1, a_2)} + x_P, \frac{g(a_1, a_2)}{h(a_1, a_2)}a_1 + y_P, \frac{g(a_1, a_2)}{h(a_1, a_2)}a_2 + z_P\right) \in \mathbb{A}^3 \end{aligned}$$

and the latter can be written as

$$\gamma_a := \left(\frac{g(a_1, a_2)}{h(a_1, a_2)} + x_P\right)\alpha_1 + \left(\frac{g(a_1, a_2)}{h(a_1, a_2)}a_1 + y_P\right)\alpha_2 + \left(\frac{g(a_1, a_2)}{h(a_1, a_2)}a_2 + z_P\right)\alpha_3 \in \mathbb{F}_{p^3}.$$

Similarly, we can write $p_U^{-1}(b_1, b_2)$ as

$$\gamma_b := \left(\frac{g(b_1, b_2)}{h(b_1, b_2)} + x_P\right)\alpha_1 + \left(\frac{g(b_1, b_2)}{h(b_1, b_2)}b_1 + y_P\right)\alpha_2 + \left(\frac{g(b_1, b_2)}{h(b_1, b_2)}b_2 + z_P\right)\alpha_3 \in \mathbb{F}_{p^3}.$$

Therefore

$$a \star b = p_U\left(\gamma_a \star_2 \gamma_b\right) = p_U\left(\frac{\gamma_a\gamma_b - B}{\gamma_a + \gamma_b - A}\right),$$

where the arguments are represented in $\mathbb{A}^3(\mathbb{F}_p)$. The next step is to compute $\gamma_a\gamma_b \in \mathbb{F}_{p^3}$.

Let $F = \frac{g}{h}$, then

$$
\begin{aligned}
\gamma_a\gamma_b = &\big((F(a)+x_P)\alpha_1 + (F(a)a_1+y_P)\alpha_2 + (F(a)a_2+z_P)\alpha_3\big)\cdot\\
&\big((F(b)+x_P)\alpha_1 + (F(b)b_1+y_P)\alpha_2 + (F(b)b_2+z_P)\alpha_3\big)\\
&= \big(F(a)F(b) + (F(a)+F(b))x_P + x_P^2\big)\,\alpha_1^2\\
&\quad + \big(F(a)F(b)b_1 + F(a)y_P + F(b)b_1x_P + x_Py_P\big)\,\alpha_1\alpha_2\\
&\quad + \big(F(a)F(b)b_2 + F(a)z_P + F(b)b_2x_P + x_Pz_P\big)\,\alpha_1\alpha_3\\
&\quad + \big(F(a)F(b)a_1 + F(a)a_1x_P + F(b)y_P + x_Py_P\big)\,\alpha_1\alpha_2\\
&\quad + \big(F(a)F(b)a_1b_1 + (F(a)a_1+F(b)b_1)y_P + y_P^2\big)\,\alpha_2^2\\
&\quad + \big(F(a)F(b)a_1b_2 + F(a)a_1z_P + F(b)b_2y_P + y_Pz_P\big)\,\alpha_2\alpha_3\\
&\quad + \big(F(a)F(b)a_2 + F(a)a_2x_P + F(b)z_P + x_Pz_P\big)\,\alpha_1\alpha_3\\
&\quad + \big(F(a)F(b)a_2b_1 + F(a)a_2y_P + F(b)b_1z_P + y_Pz_P\big)\,\alpha_2\alpha_3\\
&\quad + \big(F(a)F(b)a_2b_2 + (F(a)a_2+F(b)b_2)z_P + z_P^2\big)\,\alpha_3^2\\
&= \big(F(a)F(b) + (F(a)+F(b))x_P + x_P^2\big)\,\alpha_1^2\\
&\quad + \big(F(a)F(b)a_1b_1 + (F(a)a_1+F(b)b_1)y_P + y_P^2\big)\,\alpha_2^2\\
&\quad + \big(F(a)F(b)a_2b_2 + (F(a)a_2+F(b)b_2)z_P + z_P^2\big)\,\alpha_3^2\\
&\quad + \big(F(a)F(b)(a_1+b_1) + F(a)(a_1x_P+y_P) + F(b)(b_1x_P+y_P) + 2x_Py_P\big)\,\alpha_1\alpha_2\\
&\quad + \big(F(a)F(b)(a_1b_2+a_2b_1) + F(a)(a_2y_P+a_1z_P) + F(b)(b_2y_P+b_1z_P) + 2y_Pz_P\big)\,\alpha_2\alpha_3\\
&\quad + \big(F(a)F(b)(a_2+b_2) + F(a)(a_2x_P+z_P) + F(b)(b_2x_P+z_P) + 2x_Pz_P\big)\,\alpha_1\alpha_3\ .
\end{aligned}
$$

We see that we need the multiplication table on $\{\alpha_1,\alpha_2,\alpha_3\}$ to evaluate this as an element of $\mathbb{F}_{q^3}$ with respect to the chosen basis. Notice that also if we further take the product of $\psi_2(\gamma_a)$ and $\psi_2(\gamma_b)$ in $\mathbb{F}_{p^6}$, it will be defined in terms of the arithmetic in $\mathbb{F}_{p^6}$. This obstacle for deriving a uniform formula for the partial group law is already noted in [42, 73].

Suppose now that we have evaluated $\gamma_a\gamma_b$ under the concrete basis $\{\alpha_1,\alpha_2,\alpha_3\}$. Consider again

$$\gamma_a \star_2 \gamma_b = \frac{\gamma_a\gamma_b - B}{\gamma_a+\gamma_b-A}\,.$$

Recall that g and h are linear and quadratic polynomials (in (x_1,x_2)), respectively. Therefore, γ_a is rational in a_1,a_2 of degree 2 and γ_b is rational in b_1,b_2 of degree 2. Therefore the numerator of the right-hand side is rational in a_1,a_2,b_1,b_2 of degree 4, and the denominator is rational of degree 2. Hence we can represent $\gamma_a \star_2 \gamma_b$ (as element of $\mathbb{F}_{p^3}$) by $A_1\alpha_1 + A_2\alpha_2 + A_3\alpha_3$ where A_i are rational in a_1,a_2,b_1,b_2 of degree (at most) 4. Finally,

to go back to $\mathbb{A}^2(\mathbb{F}_p)$ we apply p_U on $\gamma_a \star_2 \gamma_b$ and get

$$a \star b = p_U(\gamma_a \star_2 \gamma_b) = p_U(A_1, A_2, A_3) = \left(\frac{A_2 - y_P}{A_1 - x_P}, \frac{A_3 - z_P}{A_1 - x_P}\right).$$

We now consider $\mathbb{T}_6$-HNP. Given $t \in \mathbb{A}^2(\mathbb{F}_p)$ and bits of component j of $s \star t$, then

$$s \star t = p_U(\gamma_s \star_2 \gamma_t) = p_U(A_1, A_2, A_3) = \left(\frac{A_2 - y_P}{A_1 - x_P}, \frac{A_3 - z_P}{A_1 - x_P}\right),$$

where A_i are rational in s_1, s_2 of degree 2.

Suppose that $j = 1$, that is we get some most significant bits of the first component of $s \star t$. We have

$$h = \mathrm{MSB}_k((s \star t)_1) = \frac{A_2 - y_P}{A_1 - x_P} - e\,,$$

so

$$(h + e)(A_1 - x_P) = A_2 - y_P\,,$$

Where A_1, A_2 are rational functions of degree 2 in s_1, s_2. As before, we assume to have $h_0 = \mathrm{MSB}_k(s_1) = s_1 - e_0$, so we can substitute $s_1 = h_0 + e_0$. Multiplying by all the involved denominators we end with a bivariate polynomial of degree 4, for which (e, s_2) is a root. Since s_2 is an unbounded unknown, we need to eliminate this unknown in all its degrees. This is expected to lead to a very large polynomial, which will result in a very weak bit security result, using the solution for MIHNP we presented above. The same analysis holds for the second component $j = 2$.

Complete Component Notice that if we get a complete component of $s \star t$, that is,

$$h = \frac{A_2 - y_P}{A_1 - x_P}\,,$$

and so

$$h(A_1 - x_P) = A_2 - y_P\,.$$

Again, multiplying by all the denominators we have a degree 4 polynomial in s_1, s_2, t_1, t_2, for which (s_1, s_2, t_1, t_2) is a root. We assume to know s_1, t_1, t_2 so we end with 4 possibilities for s_2, if this polynomial is non-zero. We can therefore have a list of at most 4 candidates for the other component.

Chapter 8

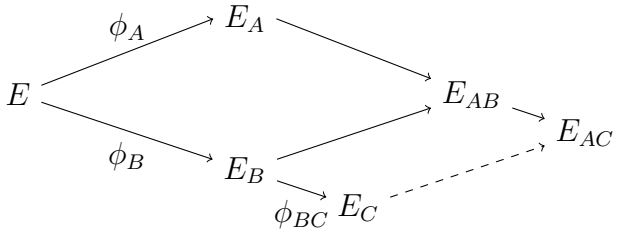
Isogeny Hidden Number Problem

This chapter is dedicated to the study of the bit security of shared keys arising from the supersingular isogeny Diffie–Hellman key exchange. To study this matter, we introduce the *isogeny hidden number problem* as a useful abstraction that allows to focus on the main reduction. The problem is formulated as follows.

> Isogeny-HNP: Fix a prime p, let f be a function over $\mathbb{F}_{p^2}$, let $E_s(\mathbb{F}_{p^2})$ be an unknown supersingular elliptic curve and let $\mathcal{O}$ be an oracle that on input r computes f on the j-invariant of an r-isogenous curve to E_s. That is, $\mathcal{O}(r) = f(j(E'))$ for some curve E' which is r-isogenous to E_s. Recover $j(E_s)$ given query access to the oracle $\mathcal{O}$.

Let us explain how the oracle $\mathcal{O}$ in this problem can be formed in the context of supersingular isogeny Diffie–Hellman key exchange. Recall that the public parameters are $P_A, Q_A, P_B, Q_B \in E$, while Alice's public keys are $E_A, \phi_A(Q_B), \phi_A(P_B)$, and Bob's public keys are $E_B, \phi_B(P_A), \phi_B(Q_A)$. The main question in bit security is what kind of information one can derive from these values, so suppose we have an oracle $\mathcal{O}'$ that takes these values and produces some partial information on $j(E_{AB})$. Denote $E_s := E_{AB}$ as the unknown elliptic curve in Isogeny-HNP. Getting the partial information from $\mathcal{O}'$ on $j(E_s) = j(E_{AB})$ is interpreted as the oracle query $\mathcal{O}(1)$.

A possible way to interact with the oracle is the following. Choose a small integer r (coprime to Alice's prime ℓ_A if working on E_B) and a point $R \in E_B[r]$ of full order. Let $\phi_{BC} : E_B \to E_C$ be an isogeny of degree r with kernel $\langle R \rangle$, that is $E_C = E_B/\langle R \rangle$. Note that there is a curve $E' := E_{AC}$ and an r-isogeny $E_{AB} \to E_{AC}$ corresponding to the image of R under the isogeny from E_B to E_{AB}. We also have that $E_{AC} = E_C/\phi_C(G_A)$ where G_A is the kernel of ϕ_A and $\phi_C = \phi_{BC} \circ \phi_B$. This situation is pictured below.



The curves E_A, E_C and the corresponding auxiliary points $\phi_A(P_B)$, $\phi_A(Q_B)$, $\phi_C(P_A) = \phi_{BC}(\phi_B(P_A))$, $\phi_C(Q_A) = \phi_{BC}(\phi_B(Q_A))$ can be used to perform a key exchange, which will constitute the curve E_{AC} (this is the dotted arrow in the figure). Querying the oracle $\mathcal{O}'$ on these values results in some partial information on $j(E_{AC})$. We interpret this as the oracle query $\mathcal{O}(r)$.

Recall that in this scheme the elliptic curves are defined over $\mathbb{F}_{p^2}$, and so their j-invariants are elements in $\mathbb{F}_{p^2}$. We represent $\mathbb{F}_{p^2}$ as a (2-dimensional) vector space over $\mathbb{F}_p$. We solve Isogeny-HNP where the partial information is a single component in $\mathbb{F}_p$ of the shared j-invariant. That is, when the oracle gives one component of $j(E_{AB})$. No other results are known for this problem, or on the bit security of supersingular isogeny Diffie–Hellman key exchange.

We first remark that since there are only around $p/12$ supersingular j-invariants, one might expect that knowledge of one component uniquely determines the entire j-invariant. This is not true in general, and it seems to be the case that there is no bound independent of p on the number of supersingular j-invariants in $\mathbb{F}_{p^2}$ with a fixed value for j_i (one exception is the rare class of j-invariants that actually lie in $\mathbb{F}_p$ and so are uniquely determined by their first component; the number of such j-invariants grows proportional to $\sqrt{p}$). Furthermore, there seems to be no known efficient algorithm that given one component of a j-invariant of a supersingular elliptic curve, computes the other component. Hence, this result is not trivial.

8.1 Exposition

Recall that we let $\mathbb{F}_{p^2} = \mathbb{F}_p(\theta)$, where $\theta^2 + A\theta + B = 0$, with $A, B \in \mathbb{F}_p$ such that $x^2 + Ax + B$ is irreducible over $\mathbb{F}_p$. The main idea is to use the modular polynomials $\Phi_r(x, y)$ as an algebraic tool to relate different answers by the oracle, i.e. the given knowledge on different j-invariants. Recall that the modular polynomial $\Phi_r(x, y)$ has the following property: there exists an isogeny $\phi : E \to E'$ of degree r with cyclic kernel if and only if $\Phi_r(j(E), j(E')) = 0$.

The framework is the following. For any $z \in \mathbb{F}_{p^2}$ we write $z = z_1 + z_2\theta$. We call z_1 a

"coefficient of 1" and z_2 a "coefficient of θ". Then

$$\Phi_2(x,y) = F_1(x_1,x_2,y_1,y_2) + F_2(x_1,x_2,y_1,y_2)\theta$$

for $F_1, F_2 \in \mathbb{F}_p[x_1,x_2,y_1,y_2]$. Let E, E' be elliptic curves with j-invariants $j = j(E)$ and $j' = j(E')$. Suppose that there exists an isogeny $\phi : E \to E'$ of degree r with cyclic kernel, then

$$\Phi_2(j,j') = F_1(j_1,j_2,j_1',j_2') + F_2(j_1,j_2,j_1',j_2')\theta = 0\,,$$

and so $F_1(j_1,j_2,j_1',j_2') \equiv F_2(j_1,j_2,j_1',j_2') \equiv 0 \pmod{p}$. One can now take a Weil descent approach with the known information about the values j_1, j_2, j_1', j_2'. Specifically, if two of these values are completely known, then one gets two bivariate polynomials. Taking their resultant yields a univariate polynomial, where one can range over its roots, and compute the other unknown.

It is clear from this method that F_1, F_2 need to be of small degree. The smallest degree of $\Phi_r(x,y)$ is for $r = 2$, for which

$$\begin{aligned}\Phi_2(x,y) = {} & x^3 + y^3 - x^2y^2 + 1488x^2y + 1488xy^2 - 162000x^2 - 162000y^2 + 40773375xy \\ & + 8748000000x + 8748000000y - 157464000000000\,.\end{aligned}$$

In this case the polynomials F_1, F_2 are then given by

$$\begin{aligned}F_1(x_1,x_2,y_1,y_2) = {} & x_1^3 + y_1^3 + ABx_2^3 + ABy_2^3 - x_1^2y_1^2 - (B^2 - A^2B)x_2^2y_2^2 + Bx_1^2y_2^2 + Bx_2^2y_1^2 \\ & + 1488x_1^2y_1 + 1488x_1y_1^2 - 1488Bx_2^2y_1 - 1488x_1y_2^2 + 1488ABx_2^2y_2 \\ & + 1488ABx_2y_2^2 + 2Ax_2^2y_1y_2 + 2Ax_1x_2y_2^2 - 162000x_1^2 - 162000y_1^2 \\ & - 162000Bx_2^2 + 162000By_2^2 - 3Bx_1x_2^2 - 3By_1y_2^2 - 4x_1x_2y_1y_2 \\ & - 2976Bx_1x_2y_2 - 2976Bx_2y_1y_2 + 40773375x_1y_1 - 40773375Bx_2y_2 \\ & + 8748000000x_1 + 8748000000y_1 - 157464000000000\,.\end{aligned}$$

$$\begin{aligned}F_2(x_1,x_2,y_1,y_2) = {} & (A^2 - B)x_2^3 + (A^2 - B)y_2^3 + Ax_1^2y_2^2 + Ax_2^2y_1^2 + 3x_1^2x_2 + 3y_1^2y_2 \\ & - (2AB - A^3)x_2^2y_2^2 + 1488x_1^2y_2 + 1488x_2y_1^2 - 3Ax_1x_2^2 - 3Ay_1y_2^2 - Ax_2^2y_1 \\ & - 1488Ax_1y_2^2 + 1488(A^2 - B)x_2^2y_2 + 1488(A^2 - B)x_2y_2^2 - (2A^2 - 2B)x_2^2y_1y_2 \\ & - 2(A^2 - 2B)x_1x_2y_2^2 - Ax_2^2 - Ay_2^2 + 4Ax_1x_2y_1y_2 + 2976x_1x_2y_1 \\ & - 2976Ax_1x_2y_2 - 2x_1^2y_1y_2 - 2x_1x_2y_1^2 + 2976(1 - A)x_1y_1y_2 - 324000x_1x_2 \\ & - 324000y_1y_2 + 40773375x_1y_2 + 40773375x_2y_1 - 40773375Ax_2y_2 \\ & + 8748000000x_2 + 8748000000y_2\,.\end{aligned}$$

Lemma 8.1. *Fix $k, l \in \{1, 2\}$ and consider the variables x_{3-k}, y_{3-l} as parameters for Φ_2. For $i \in \{1, 2\}$ define $G_i(x_k, y_l) := F_i(x_1, x_2, y_1, y_2)$, then for any x_{3-k}, y_{3-l} the resultant $\mathrm{Res}(G_1, G_2, y_l)$ is not identically zero.*

Proof. We use the fact that the modular polynomial $\Phi_r(X, Y) \in \mathbb{F}_p[X, Y]$ is absolutely irreducible (irreducible over the algebraic closure). We therefore consider Φ_r, as well as G_1, G_2, in $\overline{\mathbb{F}}_p[X, Y]$. Recall that there are four cases depending on the values of (k, l). For example when $(k, l) = (1, 2)$ we have $G_1(x_1, y_2) + G_2(x_1, y_2)\theta = \Phi_2(x_1 + j_2\theta, j_1' + y_2\theta)$.

Assume for contradiction that $\mathrm{Res}(G_1, G_2, y_l) \equiv 0$. From Lemma 2.3, the resultant $\mathrm{Res}(G_1, G_2, y_l) \equiv 0$ if and only if there exists a polynomial $h \in \overline{\mathbb{F}}_p[x_k, y_l]$ with positive degree in y_l such that $h \mid G_1$ and $h \mid G_2$.

Consider the following linear substitution of variables:

- If $k = 1$ then set $x_1 = X - j_2\theta$ and if $k = 2$ then set $x_2 = (X - j_1)\theta^{-1}$.
- If $l = 1$ then set $y_1 = Y - j_2'\theta$ and if $l = 2$ then set $y_2 = (Y - j_1')\theta^{-1}$.

These substitutions give

$$G_1(x_k, y_l) + G_2(x_k, y_l)\theta = \Phi_r(X, Y)\,.$$

Hence, letting $\bar{h}(X, Y)$ be the polynomial obtained by evaluating $h(x_k, y_l)$ with these substitutions we have

$$\bar{h}(X, Y) \mid \Phi_r(X, Y)\,.$$

From the facts that the degree of $\bar{h}$ is equal to the degree of h, and that Φ_r is irreducible, it follows (since h is non-constant) that h is a scalar multiple of both G_1 and G_2. However, by comparing the monomials in G_1, G_2, it follows that they are not constant multiples of each other. Indeed, if z_2 is known, for $z \in \{x, y\}$, then $\deg_{z_1} G_1 = 3$ while $\deg_{z_1} G_2 = 2$; the other case involves analysing the system arising from the coefficients of F_1, F_2. Hence we have a contradiction and so the resultant is non-zero. □

8.2 Main Results

Theorem 8.2. *Consider Isogeny-HNP where the oracle $\mathcal{O}$ outputs one component of $j(E') \in \mathbb{F}_{p^2}$ in its representation over $\mathbb{F}_p$. Then there is an algorithm that makes two queries to $\mathcal{O}$ and outputs a list that contains the "hidden" j-invariant $j(E_s)$. The size of the list is bounded by 18 if both components are coefficients of 1, by 12 if both components are coefficients of θ and by 15 otherwise.*

Proof. Let E_s be the unknown elliptic curve in Isogeny-HNP. The query $\mathcal{O}(1)$ gives one component of $j(E_s)$ and the query $\mathcal{O}(2)$ gives one component of $j(E')$, where E' is 2-isogenous to E_s. From Lemma 2.2, we get $\Phi_2\left(j(E_s), j(E')\right) = 0$. Let $j = j(E_s) = j_1 + j_2\theta$, $j' = j(E') = j'_1 + j'_2\theta$ and $\Phi_2 = F_1 + F_2\theta$ for two polynomials $F_1, F_2 \in \mathbb{F}_p[x_1, x_2, y_1, y_2]$. We have

$$\Phi_2(j, j') = F_1(j_1, j_2, j'_1, j'_2) + F_2(j_1, j_2, j'_1, j'_2)\theta = 0\,,$$

and so $F_1(j_1, j_2, j'_1, j'_2) \equiv F_2(j_1, j_2, j'_1, j'_2) \equiv 0 \pmod{p}$.

The oracle answers provide the values $x_{3-k} = j_{3-k}$ and $y_{3-l} = j'_{3-l}$ for $k, l \in \{1, 2\}$. Plugging these values into the polynomials F_i, we construct two bivariate polynomials G_i in variables x_k, y_l where the highest degree of each variable is at most 3. By taking the resultant of these polynomials with respect to y_l we get a univariate polynomial in x_k of degree at most 18. The resultant is not the constant zero as shown in Lemma 8.1. One can then factor this polynomial to get at most 18 roots over $\mathbb{F}_p$. One of the roots is j_k. As we have j_k and j_{3-k}, we can construct $j(E_s)$. Hence, ranging over all of these roots one can produce a list of at most 18 items, one of which is the unknown j-invariant of E_s.

Finally, notice that $\deg_{x_1} F_1 = \deg_{y_1} F_1 = 3$, $\deg_{x_2} F_1 = \deg_{y_2} F_1 \leq 3$, $\deg_{x_1} F_2 = \deg_{y_1} F_2 \leq 2$ and $\deg_{x_2} F_2 = \deg_{y_2} F_2 \leq 3$. Computing the resultant's degree in all cases gives

$$\deg_{x_k} \mathrm{Res}(G_1, G_2, y_l) \leq \begin{cases} 12 & \text{if } k = l = 1, \\ 18 & \text{if } k = l = 2, \\ 15 & \text{otherwise .} \end{cases}$$

□

8.2.1 Hardcore Bits for Supersingular Isogeny Diffie–Hellman

Theorem 8.2, along with the explanation of the source of Isogeny-HNP, gives the following bit security result for supersingular isogeny Diffie–Hellman key exchange.

Theorem 8.3. *Consider supersingular isogeny Diffie–Hellman key exchange. Given an efficient algorithm that computes any component of $j(E_{AB})$ from E_A, E_B, $\phi_A(P_B)$, $\phi_A(Q_B)$, $\phi_B(P_A)$, $\phi_B(Q_A)$, there exists a deterministic polynomial-time algorithm that computes $j(E_{AB})$.*

Unreliable Oracles

Theorem 8.2, and so also Theorem 8.3, assumes the oracle always gives the correct answer, i.e. a component of $j(E')$. We show how to achieve a similar result for an unreliable oracle.

When the oracle's probability in producing the correct result is non-negligible, then the list of candidates for the shared key is polynomial in $\log(p)$.

The main idea is to consider the isogeny graph around the curve E_s. We restrict only to the 2-isogeny graph, though one can consider other degrees, separately or simultaneously. Observe that in order to recover the j-invariant of any curve in the graph, using the method above, one needs to obtain a single component of the j-invariant of two curves of distance one (2-isogenous curves). This is the equivalent to the oracle queries $\mathcal{O}(1), \mathcal{O}(2)$ above.

We explain how to travel along the graph to recover one j-invariant, then travel back to recover $j(E_s)$. Using the oracle's success probability and a birthday paradox approach, one computes how many oracle queries are needed in order to have with a given probability the correct components of the j-invariant of two neighbouring curves (of distance one). We do not assume to know where in the graph this holds.

Start at E_s to get one component of $j(E_s)$, potentially wrong. Then query the oracle on each of its three neighbours (recall that there are three distinct 2-isogenies over $\mathbb{F}_{p^2}$) For each of them, query on the two untraveled neighbours, and so on. It is important to exhaust all curves of distance d from E_s before travelling to curves of distance $d+1$. Then, after T queries the maximal distance from E_s is $O(\log(T))$.

Fix some desired success probability to recover $j(E_s)$. Suppose one needs T' queries to have with this probability the correct component of the j-invariants of two neighbouring curves. There are (at most) $T'-1$ neighbours on this path on the graph. Taking the oracle answers on each of them and applying the technique above results in at most $18(T'-1)$ possibilities for the j-invariants of all the curves in the travelled path on the graph. We suppose one of them is the correct j-invariant of some curve. We do not assume to know which one it is.

We now use the modular polynomials to travel back to E_s. For each curve E_a of distance $d+1$ (to E_s) in this path, denote by E_b the neighbouring curve of distance d (in the unlikely event of more than one E_b, choose one of them). Take a candidate $j(E_a)$ and solve $\Phi_2\left(j(E_a), y\right) = 0$. We get at most 3 candidates y for $j(E_b)$. Proceed recursively until reaching to distance zero, i.e. to E_s. Since the maximal distance to E_s on the path is $\log(T')$, each E_a gives rise to at most $3^{\log(T')}$ candidates for $j(E_s)$, so overall we have $18(T'-1)3^{\log(T')}$ candidates for $j(E_s)$ (one can get a better bound as there are at most $T'/2$ curves of distance $\log(T')$, at most $T'/4$ curves of distance $\log(T')-1$ and so on). Finally, if the oracle's success probability is non-negligible, then T' is bounded by some polynomial in $\log(p)$ and so the list of candidates for E_s is polynomial in $\log(p)$

Remark 8.4. *Some approaches can be taken to reduce the size of the final list. For example, querying the oracle on more curves may result in expected ℓ different correct j-*

invariants in the graph. Then $j(E_s)$ will appear in ℓ different lists. One can also consider paths on other r-isogenous curves to keep a small distance from E_s.

Other Partial Information Combining the Weil descent approach with the ability to travel on the isogeny graph can lead to bit security results when other types of partial information is given. However, this in general leads to very weak results. A more precise description can be found in [35, Section 5.1.2], where the partial information is most significant bits of the two components of $j(E_s)$.

Ordinary Elliptic Curves The results presented here for supersingular isogeny Diffie–Hellman also hold for the scheme on ordinary elliptic curves [72, 89], with the necessary changes.

Chapter 9

Bit Security of CDH Under DDH

This chapter studies the hardness of computing single bits of Diffie–Hellman keys under the decisional Diffie–Hellman assumption. Under this stronger assumption we prove the bit security of individual outer bits of Diffie–Hellman keys, and of any block of 4 consecutive inner bits. The result extends for (most significant) bit fractions, using the approximations function APPR, and holds for any oracle that predicts this partial information with any non-negligible advantage over a guess.

9.1 Exposition

We rephrase DDH in the following way: For an element $g \in G$ of order T, the decisional Diffie–Hellman problem is to distinguish between the distributions (g^a, g^b, g^{ab}) and (g^a, g^b, g^c) where a, b, c are chosen independently and uniformly at random in the interval $[0, T-1]$. The decisional Diffie–Hellman assumption states that no such efficient distinguisher exists. This assumption is of great value to contemporary cryptography. We refer to [14] for a very good survey on DDH and its applications.

A folklore theorem is that computing partial information about the Diffie–Hellman key would solve DDH. Let us state the argument. Suppose one can compute partial information about g^{ab}. Given a DDH triple (g^a, g^b, g^c), where either $c \equiv ab \pmod{T}$ or c is uniformly chosen, one can take g^a, g^b, compute this partial information about g^{ab} and compare it to the same value of g^c; if the values are not the same then $g^c \neq g^{ab}$. For example, suppose there exists an algorithm $\mathcal{A}$ that computes $\mathrm{MSB}_k(g^{ab})$. Invoke $\mathcal{A}$ on g, g^a, g^b and test $\mathcal{A}(g, g^a, g^b) \stackrel{?}{=} \mathrm{MSB}_k(g^c)$. If $c \equiv ab \pmod{T}$ then the equality always holds, however if $c \not\equiv ab \pmod{T}$ then roughly speaking this equality holds $1/2^k$ of the time. This informal argument shows that computing the k most significant bits of Diffie–Hellman keys gives an advantage in solving DDH.

Surprisingly, amplifying the probability to achieve an algorithm that succeeds with

overwhelming probability could not have been achieved for single bits. A rigorous study of this problem was initiated by Blake and Garefalakis [7], where they gave a result for the 2 most significant modular bits (which can be as large as $k = 3$ in the example above), for $G = \mathbb{F}_p^*$ and $T \geq p^{1/3+\epsilon}$. Together with Shparlinski [8] they improved this result for $T \geq p^{\epsilon}$, showed that one can also consider fractions of 2 bits, and gave a similar result for $G = E(\mathbb{F}_p)$.

The earlier work also speculates if a one-bit result is achievable, where they suggest to the negative, mentioning the Legendre symbol of $g^{ab} \in \mathbb{Z}_p^*$, which is easily computable. As $|\mathbb{Z}_p^*| = p-1$ and $g^{p-1} = 1$ for every $g \in \mathbb{Z}_p^*$, the arithmetic in the exponent takes place mod $p-1$, which is an even number. Therefore, if either a or b are even integers, then $ab \mod p-1$ is also even. Given g^a, and assuming g is a generator, testing the parity of a is easily done by raising g^a to the power $(p-1)/2$ (as shown in Chapter 4); if the result is -1 then g^a is a generator and a is odd, otherwise a is even.

This example also shows one of the difficulties in constructing an algorithm to solve DDH. To start with g is not taken to be a generator (as then we can apply the Legendre symbol test), and so the multipliers – as we show later – are not uniformly distributed in $\mathbb{Z}_p^*$. This fact makes this problem less trivial than it may seem at first sight.

9.2 Representing Bits by Approximations

As this section is concerned with single bits, we emphasise that the difference between classical bits and modular bits or the more relaxed approximations should not be overlooked. As already mentioned, in some cases the more relaxed definitions for k bits, actually implies $k+1$ classical bits. Specifically, in some cases the previous results hold given 3 most significant (classical) bits. Therefore, when working with small amount of bits, like 1 or 2, it is more desirable to consider the classical bits definition, when possible.

Nevertheless, we remark that it is not necessary to restrict to classical bits. The Legendre symbol gives a good example of other partial information that one can derive. Therefore, while it is very natural to consider classical bits, other types of partial information, like the ones above, may also be of interest. In particular, we show that these types of bits are useful in the study of (classical) inner bits. However, as we now explain, one should be very careful about the results he derives when working with other types of bits. We give two examples.

Example 9.1. *Consider $\mathbb{Z}_p$ and denote $x = \lceil p/4 \rceil$ and $y = p-1$. Then $MSB_1(x) = 0$ and $MSB_1(y) = 1$. Notice that for $u := \lfloor p/8 \rceil$ we have $u = APPR_1(x)$, and u is thought to share the same most significant bit as x. We also have $u = APPR_1(y)$, where u is thought to share the same most significant bit as y. A careless conclusion is that also x*

and y share the most significant bit. This example shows that while the classical definition is transitive, the approximations are not.

This issue, that any two values in $\mathbb{Z}_p^*$ can give the same APPR_1, is essentially what prevented the previous works from achieving a single-bit result. We will show how to overcome it. A more serious caution should be taken for arguments as in the following example.

Example 9.2. *Let g be a generator of $\mathbb{Z}_p^*$ and consider an oracle $\mathcal{O}$ that outputs $APPR_1$ of Diffie–Hellman keys. That is $\mathcal{O}(g^a, g^b) = APPR_1(g^{ab})$. Writing $APPR_1(x) = x - e$, the value e is assumed to be chosen independently and uniformly at random from $[-p/4, p/4]$. Making the oracle query $\mathcal{O}(g^a, g^b)$ gives $APPR_1(g^{ab}) = g^{ab} - e$. Noticing that $-g^b = g^{b+(p-1)/2}$, the oracle query $\mathcal{O}(g^a, -g^b)$ gives $APPR_1(g^{ab}g^{a(p-1)/2}) = APPR_1(-g^{ab}) = -g^{ab} - e'$ if a is odd. We then have $V := APPR_1(g^{ab}) + APPR_1(-g^{ab}) = e + e'$ where $|e|, |e'| \leq p/4$. If $|V| = |e + e'| \leq p/4$ then we cannot derive a better bound on the size of e and e'. On the other hand, if $|V| = |e + e'| > p/4$ then e, e' are both either positive or negative (which is reflected by the sign of V).*

This shows that we can combine the two queries to get more bits of g^{ab}, indeed suppose $V > 0$ then $0 \leq e \leq p/4$ and $APPR_1(g^{ab}) - p/8 = g^{ab} - (e - p/8)$ and $-p/8 \leq e - p/8 \leq p/8$, so $APPR_1(g^{ab}) - p/8 = APPR_2(g^{ab})$; also $APPR_1(-g^{ab}) - p/8 = APPR_2(-g^{ab})$. Since e, e' are chosen independently and uniformly at random the probability for $|V| = |e + e'| > p/4$ is high. Moreover, following the same arguments we can also have that $|V| = |e + e'| \geq p/O(\log(p))$ with non-negligible probability. This will allow us to combine only 2 samples, both for a single most significant bit related to g^{ab} and receive $O(\log\log(p))$ most significant bits about g^{ab}.

Notice that if we work with MSMB instead then we always get that $MSMB_k(g^{ab}) + MSMB_k(-g^{ab}) = 2^k - 1$, so no further information can be learnt on g^{ab} from the extra query. With MSB very little information can be learnt from this extra query, and for the specific case of a Mersenne prime $p = 2^n - 1$, we always get that $\varepsilon_i(x) = 1 - \varepsilon_i(-x)$ where ε_i is the i-th bit of x.

9.3 Distribution of g^x

Let $g \in \mathbb{F}_p$ of multiplicative order T. The distribution of g^x in $\mathbb{F}_p$ is well studied. Of special interest is the following. Given an element $\lambda \in \mathbb{F}_p^*$ and positive integers r, h denote by $N_\lambda(r, h)$ the number of solutions x to the congruence $\lambda g^x \equiv v \pmod{p}$ where $0 \leq x \leq T - 1$ and $r + 1 \leq v \leq r + h$. Exponential sums are a useful tool in the understanding of the distribution of g^x where x ranges in some interval. Theorem 3.4 of

[53] implies

$$\max_{gcd(\lambda,p)=1} \left| \sum_{x=0}^{T-1} \exp(2\pi i \lambda g^x/p) \right| \ll \begin{cases} p^{1/2} & \text{if } T \geq p^{2/3}; \\ p^{1/4}T^{3/8} & \text{if } p^{1/2} \leq T \leq p^{2/3}; \\ p^{1/8}T^{5/8} & \text{if } p^{1/3} \leq T \leq p^{1/2}. \end{cases}$$

Informally speaking, exponential sums give us measure of the linear structure. If the values g^x were biased, that is they lie in one interval much more than expected, then the exponential sums would be much greater. This is also true if they form an arithmetic progression of difference d, as we can choose $\lambda = d^{-1}$.

This result allows to show that the number $N_\lambda(r,h)$ is close to its expected value $\frac{Th}{p}$. That is, among all T values λg^x, we expect that Th/p of them will lie in any interval of size h. We have the following result (see [41, Lemma 2.1]).

Lemma 9.3. *For any $\epsilon > 0$ and any $g \in \mathbb{F}_p$ of multiplicative order $T \geq p^{1/3+\epsilon}$ the bound*

$$\max_{0 \leq r,h \leq p-1} \max_{gcd(\lambda,p)=1} \left| N_\lambda(r,h) - \frac{Th}{p} \right| = O(T^{1-\delta})$$

holds, where $\delta \in \begin{cases} [1/4, 1/2] & \text{if } T \geq p^{2/3}; \\ [1/8, 1/4] & \text{if } p^{1/2} \leq T \leq p^{2/3}; \\ [9\epsilon/(8+24\epsilon), 1/8] & \text{if } p^{1/3} \leq T \leq p^{1/2}. \end{cases}$

We will also use the following stronger bound, which immediately follows from [20] (see [8, Lemma 1] and reference within).

Lemma 9.4. *For any $\epsilon > 0$ there exists $\delta > 0$ such that for any element $g \in \mathbb{F}_p$ of multiplicative order $T \geq p^\epsilon$ the bound*

$$\max_{0 \leq r,h \leq p-1} \max_{gcd(\lambda,p)=1} \left| N_\lambda(r,h) - \frac{Th}{p} \right| = O(T^{1-\delta})$$

holds.

9.4 Main Results

We give the following results. Theorems 9.5 shows that computing the most significant classical bit of the Diffie–Hellman key leads to an algorithm that solves the decisional Diffie–Hellman problem. For this result to hold (in polynomial time) one needs $p = 2^n + r$ for $r \geq p/poly(\log(p))$, so that the information given by this bit is meaningful,

as explained in Chapter 4. Theorem 9.6 gives a similar result for fractions of the most significant bit. In Section 9.4.3 we consider other bits of the Diffie–Hellman key, where Theorem 9.8 gives a similar result for the least significant bit and for small windows of any consecutive bits. Lastly, Proposition 9.7 shows that all of these results hold even if one only has some advantage in predicting the bits.

All the operations in this section are in $\mathbb{Z}_p^*$ unless otherwise stated. Our results use the following procedure, called DDH-Test, which on a triple (g^a, g^b, g^c) and an oracle $\mathcal{O}(g^a, g^b) = F(g^{ab})$ for some function F proceeds as follows.

Procedure DDH-Test

1: Choose $r \in [0, T-1]$ uniformly and independently at random.
2: Compute $g^{a+r} = g^a g^r$ and $g^{br} = (g^b)^r$.
3: Query $\mathcal{O}(g^{a+r}, g^b)$ to get $F(g^{ab} g^{br})$.
4: **if** $\mathcal{O}(g^{a+r}, g^b) = F(g^c g^{br})$ **then**
return 1
else
return 0

If $c \equiv ab \pmod{T}$ it is always the case that $\mathcal{O}(g^{a+r}, g^b) = F(g^c g^{br})$. Therefore we will have to analyze the probability that the equality $\mathcal{O}(g^{a+r}, g^b) = F(g^c g^{br})$ does not hold on random choices of g^{br} when $c \not\equiv ab \pmod{T}$. Notice that when g is a generator $\mathbb{F}_p^*$ then it is immediate that g^{br} distributes uniformly in $\mathbb{F}_p^*$. However, we already showed that in this case one can compute the Legendre symbol of g^{ab} and so the DDH assumption does not hold. For our results below we are mostly interested in the case where T is prime.

9.4.1 The Most Significant Bit

Theorem 9.5. *Let $\epsilon > 0$, let $p = 2^n + \eta$, for $0 < \eta < 2^n$, be a prime and let $g \in \mathbb{Z}_p^*$ be an element of prime order $T \geq p^\epsilon$. Suppose $\mathcal{O}$ is an oracle which takes $g^a, g^b \in \mathbb{Z}_p^*$ and outputs $MSB_1(g^{ab})$. Then there exists a probabilistic polynomial-time algorithm that takes $g^a, g^b, g^c \in \mathbb{Z}_p^*$, makes d calls to $\mathcal{O}$, and decides if $c \equiv ab \pmod{T}$ with error probability $\big(1 + o(1)\big)\left(1 - \frac{\eta+1}{2^n+\eta}\right)^d$.*

Proof. First notice that $\big|\{x \in \mathbb{Z}_p^* \mid \mathrm{MSB}_1(x) = 0\}\big| = 2^n - 1$ and $\big|\{x \in \mathbb{Z}_p^* \mid \mathrm{MSB}_1(x) = 1\}\big| = \eta$. Calling DDH-Test, if $c \equiv ab \pmod{T}$ we always have $\mathcal{O}(g^{a+r}, g^b) = \mathrm{MSB}_1(g^c g^{br})$, then we always receive 1.

On the other hand, if $c \not\equiv ab \pmod{T}$ we show that the equality $\mathcal{O}(g^{a+r}, g^b) = \mathrm{MSB}_1(g^c g^{br})$ holds with probability at most $\frac{2^n-1}{p} + O(1/T^\delta)$ for some δ that only depends on ϵ. Write $s = g^{ab}$, $w = g^c$ and $t = g^{br}$, we have $u := s - w \neq 0$. We want to know for how many different $t \in \langle g \rangle$ it holds that st and $wt = st - ut$ have the same most significant

bit. Let $V_t := \{v \in \mathbb{Z}_p^* \mid \mathrm{MSB}_1(st) = \mathrm{MSB}_1(st - v)\}$. It is clear that V_t is an interval in $\mathbb{Z}_p^*$ of size at most $2^n - 1$. We can rephrase the problem as determining the probability for $ut \in V_t$. It is clear that if t was to be distributed uniformly in $\mathbb{Z}_p^*$, then also ut, and the claim would have hold. For g as in the claim, since $|\langle g \rangle| = T$ is prime, also g^b has order T so t distributes uniformly in $\langle g \rangle$. From Lemma 9.4 we know that ut distributes in $\mathbb{Z}_p^*$ as expected. That is, the number of values for which $ut \in V_t$ is at most $\frac{T|V_t|}{p} + O(T^{1-\delta})$ for some $\delta > 0$, and therefore the probability that $\mathcal{O}(g^{a+r}, g^b) = \mathrm{MSB}_1(g^c g^{br})$ is at most $\frac{2^n - 1}{p} + O(1/T^\delta)$.

We call DDH-Test (at most) d times, and the algorithm decides that $c \equiv ab \pmod{T}$ if and only if DDH-Test returns 1 for every query. The algorithm fails with probability at most

$$\left(\frac{2^n - 1}{p}\right)^d \left(1 + O\left(\left(\frac{p}{(2^n - 1)T^\delta}\right)^d\right)\right) < \left(1 - \frac{\eta + 1}{2^n + \eta}\right)^d \left(1 + O\left(1/p^{\epsilon\delta d}\right)\right),$$

since

$$\frac{2^n - 1}{2^n + \eta} = \frac{2^n + \eta}{2^n + \eta} - \frac{\eta + 1}{2^n + \eta} = 1 - \frac{\eta + 1}{2^n + \eta}.$$

□

As we explain above modular bits and approximations allow us to consider more types of partial knowledge which give less information than a single bit. To get a more general result we work with the approximation APPR, but an analogous result can be stated for the modular bits MSMB as well.

Theorem 9.6. *Let $\epsilon > 0$, let $0 \leq i < n$ and let $g \in \mathbb{Z}_p^*$ be an element of prime order $T \geq p^\epsilon$. Suppose $\mathcal{O}$ is an oracle which takes $g^a, g^b \in \mathbb{Z}_p^*$ and outputs $APPR_k(g^{ab})$ for $k > 0$. Then there exists a probabilistic polynomial-time algorithm that takes $g^a, g^b, g^c \in \mathbb{Z}_p^*$, makes d calls to $\mathcal{O}$, and decides if $c \equiv ab \pmod{T}$ with error probability $(1 + o(1))/2^{d/k}$.*

Proof. The proof is similar to the proof of Theorem 9.5, where we repeat the algorithmic process and use the same notation as there. It is clear that if $c \equiv ab \pmod{T}$ then we always have $\left|\mathcal{O}(g^{a+r}, g^b) - g^c g^{br}\right| \leq p/2^{k+1}$. In other words $\mathcal{O}(g^{a+r}, g^b) = \mathrm{APPR}_k(g^c g^{br})$.

Suppose $c \not\equiv ab \pmod{T}$, and let $V_t := \{wt + e \mid |e| \leq \frac{p}{2^{k+1}}\}$. We want to analyze the probability that $\mathcal{O}(g^{a+r}, g^b) = \mathrm{APPR}_k(st)$ lies in V_t. Notice that for $k \leq 1$ for any $x \in \mathbb{Z}_p^*$ there exists $v \in V_t$ such that $v = \mathrm{APPR}_k(x)$. A key observation is that once g is fixed the oracle $\mathcal{O}$ only receives g^{a+r} and g^b. Therefore, even if the oracle is adversarial, it does not know g^c and therefore not the center point wt of V_t or the distance between st to V_t. Since $\mathrm{APPR}_k(st)$ is a function of st it is sufficient to show that st distributes

(almost) uniformly among all intervals of size $|V_t|$, as already mentioned that with respect to $\mathcal{O}(g^{a+r}, g^b)$ the set V_t is independent.

Applying Lemma 9.4, the number of values for which st lies in any interval of size $|V_t|$ is at most $\frac{T|V_t|}{p} + O(T^{1-\delta}) = \frac{T(1+p/2^k)}{p} + O(T^{1-\delta})$ for some $\delta > 0$, and therefore the probability that $\mathcal{O}(g^{a+r}, g^b) = \mathrm{APPR}_k(g^c g^{br})$ is at most $\frac{1}{2^k} + O(p^{-1} + p^{-\epsilon\delta})$. Repeating d times gives the desired result. □

9.4.2 Predicting the Bits

Suppose the oracle $\mathcal{O}$ is unreliable. That is, it does not always give the correct output $F(g^{ab})$. We show that as long as it has a non-negligible advantage over the guessing strategy in predicting $F(g^{ab})$, our results still hold. Here, the guessing strategy is taken with respect to expected value of solutions $N_\lambda(r, h)$.

Proposition 9.7. *In the previous theorems, suppose the oracle $\mathcal{O}$ has non-negligible advantage over $\frac{1}{2^k} + O(T^{-\delta})$ in predicting $F(g^{ab})$. Then, there exists a probabilistic polynomial-time algorithm that takes $g^a, g^b, g^c \in \mathbb{Z}_p^*$, and decides if $c \equiv ab \pmod{T}$ with overwhelming probability.*

Proof sketch. The rule of determining whether $c \equiv ab \pmod{T}$ needs to be changed as it is no longer guaranteed that if $c \equiv ab \pmod{T}$ DDH-Test always outputs 1. However, we can use the distribution of outputs given by DDH-Test to solve the DDH problem. Denote by D_{ab} the distribution of DDH-Test outputs when $c \equiv ab \pmod{T}$ and by D_c the distribution of DDH-Test outputs when $c \not\equiv ab \pmod{T}$. Since $\mathcal{O}$ has a non-negligible advantage, the two distributions are non-negligibly far. We can therefore call DDH-Test sufficiently many times (derived from the Chernoff–Hoeffding bound) and determine if the outputs follow D_{ab}. □

9.4.3 Other Bits

Applying the techniques in Section 5.1.1 immediately gives that any four consecutive bits of Diffie–Hellman keys are as hard to compute as solving the DDH problem. Moreover, a similar statement holds for the least significant bit.

Theorem 9.8. *Let $\epsilon > 0$, let $0 < i \leq n$ and let $g \in \mathbb{Z}_p^*$ be an element of prime order $T \geq p^\epsilon$. Suppose $\mathcal{O}$ is an oracle which takes $g^a, g^b \in \mathbb{Z}_p^*$ and outputs $Bits_{i,i+3}(g^{ab})$. Then there exists a probabilistic polynomial-time algorithm that takes $g^a, g^b, g^c \in \mathbb{Z}_p^*$, makes d calls to $\mathcal{O}$, and decides if $c \equiv ab \pmod{T}$ with error probability $(1 + o(1))/2^{d/0.4}$. The same result holds if $\mathcal{O}$ outputs $LSB_1(g^{ab})$.*

Proof. Given $\mathrm{Bits}_{i,i+3}(g^{ab}) = (\varepsilon_{i+3}, \ldots, \varepsilon_i)$ we compute $\alpha = \sum_{j=0}^{3} \varepsilon_{i+j} 2^{i+j}$ so that $g^{ab} = 2^{i+4}\beta + 2^i\alpha + \gamma$ for some integers $0 \le \beta \le p/2^{i+4}$ and $0 \le \gamma < 2^i$. As explained in Section 5.1.1 above, using the convergents from the continued fraction expansion of w/p, for $w = \left((2^{i+4})^{-1} \mod p\right)$ we obtain $\lambda \in \mathbb{Z}_p^*$ such that $|\lambda| < p/2^{i+2}$ and $0 < \lambda 2^{i+4} \le 2^{i+2}$. Then $\lambda 2^i\alpha + p/8 = \lambda g^{ab} - (\lambda 2^{i+4}\beta - p/8 + \lambda\gamma)$, and

$$\left|\lambda 2^{i+4}\beta - \frac{p}{8} + \lambda\gamma\right| \le \left|\lambda 2^{i+4}\beta - \frac{p}{8}\right| + |\lambda|\gamma < \left|\frac{p2^{i+2}}{2^{i+4}} - p/8\right| + \frac{p2^i}{2^{i+2}} = \frac{p}{8} + \frac{p}{4} = \frac{p}{2^{2+\log(2/3)}} < \frac{p}{2^{1.4}}\,.$$

Thus, $\lambda 2^i\alpha + p/8 = \mathrm{APPR}_{0.4}(\lambda g^{ab})$. Notice that since λ only depends on i and p, we in fact get a deterministic representation of elements in $\mathbb{Z}_p$.

If $\mathcal{O}(g^a, g^b) = \mathrm{Bit}_0(g^{ab})$, then $\alpha := \mathrm{Bit}_0(g^{ab}) = g^{ab} - 2\beta$ for $0 \le \beta \le p/2$. We take $\lambda = (p+1)/2 = 2^{-1}$ so $\lambda\alpha + \lfloor p/4 \rceil = \lambda g^{ab} - (\beta - \lfloor p/4 \rceil) = \mathrm{APPR}_1(\lambda g^{ab})$.

In both cases, apply Theorem 9.6 with $g^a, g^b, \lambda g^c$. □

We now consider the case of an oracle that computes a single bit of the Diffie–Hellman key, that is $\mathcal{O}(g^a, g^b) = \mathrm{Bit}_i(g^{ab})$ for $0 < i < n$. We need to show that when $c \not\equiv ab \pmod T$ the probability that DDH-Test outputs 0 is non-negligible. The most challenging part in approaching this case is the analysis of the distribution of g^x. We use Lemma 9.3 for the case $T > p^{1/3+\epsilon}$ to show that with non-negligible probability g^x does not lie in some highly linearly structured set.

We use the notation in Theorem 9.5, that is we define $u = g^{ab} - g^c$ and suppose $u \ne 0$. Let $V_t := \{v \in \mathbb{Z}_p^* \mid \mathrm{Bit}_i(g^{ab}t) = \mathrm{Bit}_i(g^{ab}t - v)\}$. We analyze the probability that $ut \in V_t$. Notice that V_t is not an interval, however it is still highly structured – it consists of intervals of similar size (except for at most one interval), spaced according to an arithmetic progression of difference 2^{i+1}.

Let ℓ be the number of intervals. We can reduce the problem to ℓ independent questions about the number of solutions $N_{\lambda_j}(l_j, h_j)$ for $1 \le j \le \ell$, where $\sum h_j \approx p/2$. We get that the number of values $ut \in V_t$ is $\sum_{j=1}^{\ell} N_{\lambda_j}(l_j, h_j)$ which satisfies

$$\sum_{j=1}^{\ell} \max_{gcd(\lambda_j, p)=1} \left| N_{\lambda_j}(l_j, h_j) - \frac{Th_j}{p} \right| = \sum_{j=1}^{\ell} O(T^{1-\delta}) \le \ell O(T^{1-\delta})\,.$$

This implies that the probability that $ut \in V_t$ is $\frac{1}{2} + \ell O(T^{-\delta})$, where δ is given in Lemma 9.3. We therefore need $\ell O(T^{-\delta})$ to be non-negligibly far from $1/2$. Lastly, notice that without loss of generality we can assume that $\ell \le p^{1/2}/2$ as if $\ell > p^{1/2}$ (as happens for the bottom half of the bits; notice $h_j \le p^{1/2}/2$) then taking $\lambda = 2^{-i-1}$ permutes V_t to contain at most $\ell \le p^{1/2}/2$ (then $h_j \ge p^{1/2}$). We get, for example, that in the case $T = (p-1)/2$ all single bits, except of the middle $O(\log\log(p))$ bits, are as hard to

compute as solving the DDH problem.

Special Cases

We consider three special cases where one can prove that computing $\mathrm{Bit}_i(g^{ab})$ for every $0 \leq i \leq n$ will solve the DDH problem.

Primes of a Special Form Fix $0 < i < n$, and suppose that p is of the form $p = \lambda 2^{i+2} + r$ for some $0 < r < 2^i$. Then $|\lambda 2^{i+2}| \equiv |r| < 2^i \pmod{p}$ and $\lambda = (p-r)/2^{i+2} < p/2^{i+2}$. In this special case, applying the technique from Theorem 9.8 gives $\mathrm{APPR}_1(\lambda g^{ab})$ which we can solve using Theorem 9.6. Notice that this λ permutes all the subintervals in V_t into one interval, as discussed in the previous section.

Modified DDH As noted above, if the values g^{br} were to be distributed uniformly in $\mathbb{Z}_p^*$, then proving that with non-negligible probability $ut \notin V_t$ would be very easy. Consider the following variant of the DDH problem: $p = 2q + 1$ is a safe prime and g generates $\mathbb{Z}_p^*$. Now, instead of taking a, b, c uniformly at random in $[0, p-1]$, the challenger takes b (or a) to be odd, computes the Legendre symbol of g^{ab}, and then let g^c to have the same Legendre symbol. That is, if a (or b) is even, then c is chosen uniformly at random in $2[0, q]$, and otherwise it is chosen from $2[1, q] - 1$. Then, one cannot use the Legendre symbol test to solve DDH, and since g^b is a generator then the values g^{br} distribute uniformly in $\mathbb{Z}_p^*$. Under this variant of the DDH problem, we get that computing any single bit of the Diffie–Hellman key is as hard as solving this DDH problem.

Stronger Oracle If instead of oracles with a fixed base point g that only take as input g^a, g^b, one considers oracles that also take the base point as input, then one can "get out" of the subgroup generated by g, and achieve uniform multipliers in $\mathbb{Z}_p^*$. We consider the safe-prime case $p = 2q+1$ for a prime q, but this can be generalised. Suppose a generator h for $\mathbb{Z}_p^*$ has been found. Then there exists l such that $g = h^{2l}$, and so $g^a = h^{2la}$ and $g^b = h^{2lb}$. Hence the Diffie–Hellman key $g^{ab} = h^{4l^2ab}$.

Notice that $g^b h = h^{2lb+1}$ is also a generator. We have $g^a h^r = h^{2la+r}$ and $g^b h = h^{2lb+1}$. Therefore, querying $\mathcal{O}(h, g^a h^r, g^b h)$ we receive a single bit of $h^{(2la+r)(2lb+1)} = h^{4l^2ab+2la+(2lb+1)r} = (g^{ab} g^a)(g^b h)^r$. Denote $t := (g^b h)^r$, and notice that t distributes uniformly in $\mathbb{Z}_p^*$. Lastly, given g^c, we use DDH-Test on $g^c g^a$ to test if it equals to $g^{ab} g^a$.

Similar to the previous case, having uniform multipliers in $\mathbb{Z}_p^*$ makes the proof easy. We get that under this oracle model, computing any single bit of the Diffie–Hellman key is as hard as solving the DDH problem.

Chapter 10

Concluding Remarks

This thesis presented a study of the hidden number problem and its applications to the bit security of Diffie–Hellman key exchange. The hidden number problem in finite fields has received the most attention in the literature and has been extensively studied. This study explored other Diffie–Hellman groups, where first bit security results were given for prime-field elliptic curves and algebraic tori, as well as for supersingular isogeny Diffie–Hellman. These results were accomplished by reducing the computational Diffie–Hellman problem to the problem of computing certain bits of the Diffie–Hellman keys. In a similar fashion, the study improved the bit security results that rely on decisional Diffie-Hellman. In the spirit of the times the study examined a Fourier analysis approach to the hidden number problem and possible applications to the bit security of Diffie–Hellman schemes. Furthermore, some results on Fourier concentrated functions were given.

The overall results in this field are far from being satisfactory. The fact that we only have proofs that $\sqrt{\log(p)}$ bits of finite-field Diffie–Hellman keys or 5/6-th of all bits of elliptic curve Diffie–Hellman keys are as hard to compute as the entire key does not mean that it is possible to compute smaller portions of bits. In fact, it is believed that if the Diffie–Hellman key exchange scheme takes place in a group of prime order, then no partial information can be computed about the Diffie–Hellman key. Whether this lack of reductions implies stronger security of the scheme is a question on the philosophical side. For example, if one leaks half the bits of his elliptic curve Diffie–Hellman keys and recovering his key remains intractable, should that be taken as an evidence that elliptic curves obfuscate the keys very well and increase our belief in their security? Or should this uncertainty implies less confidence in the security of the entire scheme?

It seems that new directions should be taken to improve the current results, furthermore it is known that the original lattice approach taken on the hidden number problem cannot be used to prove the security of single bits (see the analysis in [71, Section 12.3.3]). We briefly present new approaches to be further investigated.

10.1 Future Directions

In the original hidden number problem some structure can be obtained from the fact that the problem gives rise to an arithmetic progression mod p. Let us explain, recall that one receives pairs of the form (t, h) where

$$h = \mathrm{APPR}_k(st) \equiv st - e \pmod{p},$$

where s is a secret value to be recovered and e is unknown such that $|e| \leq p/2^{k+1}$ (this is $\mathbb{F}_p$-HNP with $f = \mathrm{APPR}_k$). Rearranging this equation we have $s \equiv ht^{-1} + et^{-1} \pmod{p}$. The right-hand side is an arithmetic progression (in $\mathbb{Z}_p$) of difference t^{-1}, where e ranges from $-\lfloor p/2^{k+1} \rfloor$ to $\lfloor p/2^{k+1} \rfloor$. This fact allows us to represent s in the following way

$$s = n_1 x_1 + \cdots + n_z x_z + b,$$

where the x_i are *bounded* variables and $n_1, \ldots, n_z, b$ are known. Notice that the equality is over the integers. Expressions of this form are called generalised arithmetic progressions, or linear sets, and are closely related to the study of Bohr sets.

Moreover, it is the case that $n_1 > n_2 > \cdots > n_z > b$. In fact, this expression can be thought of as a representation of s under the tuple $(n_1, \ldots, n_z, 1)$, as

$$(s \mod n_1) \ldots \mod n_j = n_{j+1} x_{j+1} + \cdots + n_z x_z + b.$$

We introduce below a process that takes (t, h) and produces the value $n_1, \ldots, n_z, b$.

The main obstacle is that the tuple $(n_1, \ldots, n_z)$ is dependent on the multiplier t, and so for different multipliers we get different values. It is not clear how to combine in general this information from the different samples, though we show below that one can combine the information from pairs of samples, to reduce the (explicit) list of possible secrets.

We remark that it should also be possible to combine three samples, but in general this problem is closely related to simultaneous Diophantine approximation and the results of Minkowski in this area. It is interesting to study if this approach of arithmetic progression mod p has a strong relation to the theory of *Beatty sequences*.

We also remark that since the equations hold over the integers, from each sample (t^i, h^i) one can form the modular relation

$$s \equiv n_2^i x_2^i + \cdots + n_{z^i}^i x_{z^i}^i + b^i \pmod{n_1^i}.$$

This can lead to an interesting variant of the Chinese remainder theorem, with bounded

variables, where one tries to compute the smallest value that satisfies all of the modular equations. It is of interest to examine if the theory of quantum computing can be of use for such problems.

Lastly, one can define functions $f^i : \mathbb{Z}_p \to \mathbb{C}$ by

$$f^i = \sum_{x_1^i,\ldots,x_{z^i}^i} \omega_p^{n_1^i x_1^i + \cdots + n_{z^i}^i x_{z^i}^i + b^i},$$

for which s is a heavy coefficient. As a product of exponential sums, these functions are efficiently computable, and so it is interesting to see if tools from Fourier analysis can be used to determine the unique simultaneous heavy coefficient.

In a different aspect, the hidden number problem is closely related to the learning with errors problem. It is of great interest to study this relation in a rigour way. A first approach of this kind was taken in [21], moreover the recent work [36] studies these relations in great detail, using notions from Fourier analysis on finite groups.

Examples and Further Details

Given (t, h) it holds that $h \equiv st - e \pmod{p}$, so over the integers we have $h = st - e - lp$, for some integer l. Let $n_1 := t^{-1} \pmod{p}$, and express $tn_1 = m_1 p + 1$. Then

$$n_1 h = stn_1 - en_1 - lpn_1 = s(m_1 p + 1) - en_1 - lpn_1 = sm_1 p + (s - en_1) - lpn_1 \,.$$

Dividing by p, we get

$$\frac{n_1}{p} h = \frac{1}{p}(sm_1 p + (s - en_1) - lpn_1) = sm_1 + \frac{s}{p} - e\frac{n_1}{p} - ln_1 \,.$$

Express $s = n_1 x_1 + s_1$, where $s_1 = s \mod n_1$ for some integer $0 \leq x_1 \leq p/n_1$ and let $\tilde{e}_1 := -\frac{s}{p} + e\frac{n_1}{p}$. We then have

$$\frac{n_1}{p} h = s_1 m_1 - \tilde{e}_1 - (l - x_1) n_1 \equiv s_1 m_1 - \tilde{e}_1 \pmod{n_1} \,.$$

One can round this equation (this is not necessary; in fact also dividing by p is not necessary, as we can work over the modulus pn_1 with the secret sp), and express this value as $(m_1, \tilde{h}_1)$ which is an instance of the hidden number problem mod n_1. Notice that $s/p < 1$, so the number of bits that we have, which is reflected by $\tilde{e}_1$ has almost not changed, as the concealed information by $e\frac{n_1}{p}$ over the modulus n_1 is proportional to the concealed information by e over the modulus p. However, the secret has changed and it is now $s_1 = s \mod n_1$. One can repeat this process until the noise term vanishes. We

give a brief example

Example 10.1. *Let* $p = 15485863$, $s = 4925145$, $t = 9553044$ *and* $|e| \leq 2^{16}$. *Suppose we have the sample* $(t, 2305377)$, *therefore*

$$h_1 = 2305377 = st_1 - 10034 \mod p\,.$$

Denoting $n_1 := -t^{-1} = 2554491$ *we have* $tn_1 = 9553044 \cdot 2554491 = 1575835p - 1$ *and so* $m_1 = 1575835$. *Thus*

$$\tilde{h}_1 := \lfloor \frac{n_1}{p} h \rceil = \lfloor s_1 m_1 - \frac{s}{p} - e\frac{n_1}{p} \rceil = 380287\,.$$

This gives the pair $(m_1, \tilde{h}_1)$ *over the modulus* n_1, *with secret* $s_1 = s \mod n_1 = 2370654$, *where the noise term* $\tilde{e}_1 = \lfloor \frac{s}{p} + e\frac{n_1}{p} \rceil$ *satisfies* $-10811 \leq \tilde{e}_1 \leq 10811 + 1$.

We repeat where now we denote $n_2 := m_1^{-1} = 158917$ *and so* $m_2 = 98034$, *and*

$$\tilde{h}_2 := \lfloor \frac{n_2}{n_1} \tilde{h}_1 \rceil = \lfloor s_2 m_2 + \frac{s_1}{n_1} - \tilde{e}_1 \frac{n_2}{n_1} \rceil = 23658\,.$$

This gives the pair $(m_2, \tilde{h}_2)$ *over the modulus* n_2, *with secret* $s_2 = s_1 \mod n_2 = 145816$, *where the noise term* $\tilde{e}_2$ *satisfies* $-674 \leq \tilde{e}_1 \leq 673$.

We repeat this procedure to have $n_3 := 11819$, $n_4 := 5270$, $n_5 := 1279$, *and* $n_6 = 154$ *where we have* $(m_6 = 95, \tilde{h}_6 = 23)$ *and the noise term* $\tilde{e}_5$ *is either* 0 *or* -1, *therefore the secret* $s_6 \in \{151, 104\}$. *The following modulus is* $n_7 = m_6^{-1} = 47$, *where in both cases the secret* s_6 *satisfies* $s_6 \mod 47 = 10$.

We can therefore write

$$s = 2554491x_1 + 158917x_2 + 11819x_3 + 5270x_4 + 1279x_5 + 154x_6 + 10\,.$$

Indeed, $(((((s \mod n_1) \mod n_2) \mod n_3) \mod n_4) \mod n_5) \mod n_6 = 10$. *The bounds on the variables are* $x_1 < 7, x_2 < 17, x_3 < 14, x_4 < 3, x_5 < 5, x_6 < 9$.

Notice that this procedure runs in time polynomial in $\log(p)$ – in particular one can guarantee to have $|z| < \log(p)$, as either n_i or $-n_i$ is smaller than $n_{i-1}/2$. We now explain how the theory of continued fractions allows us to combine two samples.

In fact one does not have to take n_1 to be $\pm t^{-1}$, but any desired value. The theory of continued fractions, some of which is presented earlier, helps us to find "good" values n_1 that on the one hand are small, and on the other give small remainders. This would allow us to have shorter tuples (though larger variables x_i). To combine different samples, one would need to simultaneously work with different multipliers. Consider two instances

(t_1, h_1) and (t_2, h_2). Rephrasing the result of Vinogradov that is presented in 5.1.1, given $t_1^{-1}t_2$ and a bound λ, we can find integers x, y such that $0 < x \leq \lambda$, $|xy| < p$ and $t_1^{-1}t_2 x \equiv y \pmod{p}$. Then $t^{-1}x \equiv z \equiv t_2^{-1}y$, and so $t_1 z \equiv x$ and $t_2 z \equiv y$. It seems desirable to take $\lambda = \sqrt{p}$ such that on multiplication by z, both residues are at most $\sqrt{p}$. We then set $n_1 := z$, which is now simultaneous for t_1, t_2. One can repeat this approach for the subsequent values m_1, m_2 over modulus n_1 etc. We give an example.

Example 10.2. *As before, let* $p = 15485863$, $s = 4925145$, $t_1 = 9553044$ *and* $|e| \leq 2^{16}$, *and let* $t_2 = 10282847$. *Suppose we have the samples* $(t_1, 2305377), (t_2, 10772380)$, *therefore*

$$h^1 = 2305377 = st_1 - 10034 \mod p\,, \qquad h^2 = 10772380 = st_1 + 63875 \mod p\,.$$

Using the continued fraction convergents of $t_1^{-1}t_2/p$ *we find that* $t_1^{-1}t_2 1654 \equiv -2269 \pmod{p}$. *Then* $t_1^{-1}1654 \equiv 2512485 \equiv t_2^{-1}(-2269)$. *Therefore, by setting* $n_1 := 2512485$ *we have*

$$\frac{n_1}{p}h^1 = s_1 m_1^1 + \frac{1654s}{p} - e^1\frac{n_1}{p} \mod n_1\,,$$
$$\frac{n_1}{p}h^2 = s_1 m_1^2 - \frac{2269s}{p} - e^2\frac{n_1}{p} \mod n_1\,,$$

where $m_1^1 = 1549922$ *and* $m_1^2 = 1668328$. *We can therefore write* $(m_1^1, 374033), (m_1^2, 1747752)$ *as two samples of the hidden number problem mod* 2512485 *with secret* $s_1 = s \mod 2512485$, *where the noise term* $\tilde{e}_1^1$ *on the first sample satisfies* $-12287 = -1654 - \lfloor 2^{16}\frac{n_1}{p} \rceil \leq \tilde{e}_1^1 \leq \lfloor 2^{16}\frac{n_1}{p} \rceil = 10633$ *and the noise term* $\tilde{e}_1^2$ *on the second sample satisfies* $-10633 = -\lfloor 2^{16}\frac{n_1}{p} \rceil \leq \tilde{e}_1^1 \leq 2269 + \lfloor 2^{16}\frac{n_1}{p} \rceil = 12902$.

We repeat this procedure to have $n_2 := 374371$, $n_3 := 36582$, $n_4 := 13453$ *and* $n_5 := 1634$. *We then have the samples* $(m_5^1 = 1008, \tilde{h}_5^1 = 243), (m_5^2 = 1085, \tilde{h}_5^2 = 1137)$ *mod* 1634 *with secret* $s_5 := ((((s_1 \mod n_1) \mod n_2) \mod n_3) \mod n_4) \mod n_5$, *where the noise term* $\tilde{e}_5^1$ *on the first sample satisfies* $-12 \leq \tilde{e}_5^1 \leq 100$ *and the noise term* $\tilde{e}_5^2$ *on the second sample satisfies is* $-36 \leq \tilde{e}_5^2 \leq 32$. *For each of this samples, we range over all the possible noise terms and list the candidates for* s_5 *(as* m_5^1 *divides* n_5 *we work over* $n_5/2 = 817$ *and lift the solutions to* n_5*). Then by intersecting these lists we find that* $s_5 \in \{1242, 117\}$. *Therefore, taking* $n_6 := 1242 - 117 = 1125$ *and* $s_6 = s_5 \mod 1125$, *we get* $s_6 = 117$. *We can therefore write*

$$s = 2512485x_1 + 374371x_2 + 36582x_3 + 13453x_4 + 1634x_5 + 1125x_6 + 117\,.$$

Indeed, $(((((s \mod n_1) \mod n_2) \mod n_3) \mod n_4) \mod n_5) \mod n_6 = 117$. *The bounds on the variables are* $x_1 < 7, x_2 < 7, x_3 < 11, x_4 < 3, x_5 < 9, x_6 < 2$. *Therefore, this gives*

a total of $7 \cdot 7 \cdot 11 \cdot 3 \cdot 9 \cdot 2 = 29106$ *possible values for* s*. Notice that each single sample gives rise to* $2^{17} + 1 = 131073$*, so by combining the two samples we have reduced the list of possible values by a factor of* ≈ 4.5*.*

References

[1] Akavia, A. (2009) "Solving Hidden Number Problem with One Bit Oracle and Advice," in Halevi, S. (ed.) Advances in Cryptology – CRYPTO 2009. LNCS, vol. 5677, pp. 337–354. Springer, Heidelberg.

[2] Akavia, A., Goldwasser, S., and Safra, S. (2003) "Proving Hard-Core Predicates Using List Decoding," in FOCS 2003, pp. 146–157. IEEE Computer Society, Washington, DC.

[3] Alexi, W., Chor, B., Goldreich, O., and Schnorr, C.P. (1988) "RSA and Rabin Functions: Certain Parts are as Hard as the Whole," in SIAM Journal on Computing, **17**(2), 194–209.

[4] Aranha, D.F., Fouque, P.-A., Gérard B., Kammerer, J.-G., Tibouchi., M., and Zapalowicz, J.-C. (2014) "GLV/GLS Decomposition, Power Analysis, and Attacks on ECDSA Signatures with Single-Bit Nonce Bias," in Sarkar, P., Iwata, T. (eds.) Advances in Cryptology – ASIACRYPT 2014. LNCS, vol. 8873, pp. 262–281. Springer, Heidelberg.

[5] Babai, L. (1986) "On Lovász' Lattice Reduction and the Nearest Lattice Point Problem," in Combinatorica, **6**(1), 1–13.

[6] Ben-Or, M., Chor, B., and Shamir, A. (1983) "On the Cryptographic Security of Single RSA Bits," Johnson, D.S., Fagin, R., Fredman, M.L., Harel, D., Karp, R.M., Lynch, N.A., Papadimitriou, C.H., Rivest, R.L., Ruzzo, W.L., Seiferas, J.I. (eds.) STOC 1983, pp. 421–430. ACM, New York.

[7] Blake, I.F., and Garefalakis, T. (2004) "On the Complexity of the Discrete Logarithm and Diffie–Hellman Problems," in Journal of Complexity, **20**(2-3), 148–170.

[8] Blake, I.F., Garefalakis, T., and Shparlinski, I.E. (2006) "On the Bit Security of the Diffie–Hellman Key," in Applicable Algebra in Engineering, Communication and Computing, **16**(6), 397–404.

[9] Blake, I.F., Seroussi, G., and Smart, N.P. (1999) Elliptic Curves in Cryptography. Cambridge University Press.

[10] Bleichenbacher, D. (2000) "On the Generation of One-time Keys in DL Signature Schemes," Presentation at IEEE P1363 Working Group meeting.

[11] Blackburn, S.R., Gomez-Perez, D., Gutierrez, J, and Shparlinski, I.E. (2003) "Predicting the Inversive Generator," in Paterson, K.G. (ed.) Cryptography and Coding. LNCS, vol. 2898, pp. 264–275. Springer, Heidelberg.

[12] Blackburn, S.R., Gomez-Perez, D., Gutierrez, J, and Shparlinski, I.E. (2005) "Predicting Nonlinear Pseudorandom Number Generators," in Mathematics of Computation, **74**(251), 1471–1494.

[13] Brouwer, A.E., Pellikaan, R., and Verheul, E.R. (1999) "Doing More with Fewer Bits," in Advances in Cryptology – ASIACRYPT '99, 321–332.

[14] Boneh, D. (1998) "The Decision Diffie–Hellman problem," in Buhler, J.P. (ed.) Algorithmic Number Theory – ANTS-III. LNCS, vol. 1423, pp. 48–63. Springer, Heidelberg.

[15] Boneh, D., Halevi, S., and Howgrave-Graham, N. (2001) "The Modular Inversion Hidden Number Problem," in Boyd, C. (ed.) Advances in Cryptology – ASIACRYPT 2001. LNCS, vol. 2248, pp. 36–51. Springer, Heidelberg.

[16] Boneh, D., and Shparlinski, I.E. (2001) "On the Unpredictability of Bits of the Elliptic Curve Diffie–Hellman Scheme," in Kilian, J. (ed.) Advances in Cryptology – CRYPTO 2001. LNCS, vol. 2139, pp. 201–212. Springer, Heidelberg.

[17] Boneh, D., and Venkatesan, R. (1996) "Hardness of Computing the Most Significant Bits of Secret Keys in Diffie–Hellman and Related Schemes," in Koblitz, N. (ed.) Advances in Cryptology – CRYPTO '96. LNCS, vol. 1109, pp. 129–142. Springer, Heidelberg.

[18] Boneh, D., and Venkatesan, R. (1997) "Rounding in Lattices and its Cryptographic Applications," in Saks, M.E. (ed.) SODA 1997, pp. 675–681. ACM/SIAM, Philadelphia.

[19] Bourgain, J., and Konyagin, S.V. (2003) "Estimates for the Number of Sums and Products and for Exponential Sums over Subgroups in Fields of Prime Order," in Comptes Rendus Mathematique, Acad. Sci. Paris, Ser. I **337**(2), 75–80.

[20] Bourgain, J., Glibichuk, A.A., and Konyagin, S.V. (2006) "Estimates for the Number of Sums and Products and for Exponential Sums in Fields of Prime Order," in Journal of the London Mathematical Society, **73**(2), 380–398.

[21] Brakerski, Z., Langlois, A., Peikert, C., Regev, R., and Stehlé, D. (2013) "Classical Hardness of Learning with Errors," in Proc. 45th ACM Symposium on Theory of Computing – STOC 2013, pp. 575–584. ACM, New York.

[22] Childs, A.M., Jao, D., and Soukharev, V. (2014) "Constructing Elliptic Curve Isogenies in Quantum Subexponential Time," in Journal Mathematical Cryptology, **8**(1), 1–29.

[23] Coppersmith, D. (1997) "Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities," in Journal of Cryptology, **10**(4), 233–260.

[24] Couveignes, J.-M. (2006) "Hard Homogeneous Spaces," in Cryptology ePrint Archive, Report 2006/291. http://eprint.iacr.org/2006/291.

[25] Cox, D.A. (1989) Primes of the Form $x^2 + ny^2$. John Wiley & Sons, Inc. New York.

[26] Cox, D.A., Little, J., and O'Shea, D. (2007) Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra. Undergraduate Texts in Mathematics (3nd edition). Springer-Verlag, New York.

[27] De Mulder, E., Hutter, M., Marson, M.E., and Pearson, P. (2013) "Using Bleichenbacher's Solution to the Hidden Number Problem to Attack Nonce Leaks in 384-Bit ECDSA," in Bertoni, G., Coron, J.-S. (eds.), CHES 2013, Springer LNCS 8086, 435–452.

[28] Diffie, W., and Hellman, M.E. (1976) "New directions in cryptography," *IEEE Transactions on Information Theory*, **12**(6), 644–654.

[29] Duc, A., and Jetchev, D. (2012) "Hardness of Computing Individual Bits for One-Way Functions on Elliptic Curves," in Safavi-Naini, R., Canetti, R. (eds.) Advances in Cryptology – CRYPTO 2012. LNCS, vol. 7417, pp. 832–849. Springer, Heidelberg.

[30] Fazio, N., Gennaro, R., Perera I.M., and Skeith, W.E. III (2013) "Hard-Core Predicates for a Diffie–Hellman Problem over Finite Fields," in Canetti, R., Garay, J.A. (eds.) Advances in Cryptology – CRYPTO 2013. LNCS, vol. 8043, pp. 148–165. Springer, Heidelberg.

[31] Galbraith, S.D. (2012) Mathematics of Public Key Cryptography. Cambridge University Press.

[32] Galbraith, S.D., Hopkins, H.J., and Shparlinski, I.E. (2004) "Secure Bilinear Diffie–Hellman Bits," in Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) Information Security and Privacy – ACISP 2004, Proc. 9th Australasian Conference. LNCS, vol. 3108, pp. 370–378. Springer, Heidelberg.

[33] Galbraith, S.D., Laity, J., and Shani, B. (2016) "Finding Significant Fourier Coefficients: Clarifications, Simplifications, Applications and Limitations," under review. arXiv:1607.01842 [cs.CR]. https://arxiv.org/abs/1607.01842.

[34] Galbraith, S.D., and Shani, B. (2015) "The Multivariate Hidden Number Problem," in Lehmann, A., Wolf, S. (eds.) Information Theoretic Security – ICITS 2015, Proc. 8th International Conference on Information-Theoretic Security. LNCS, vol. 9063, pp. 250–268. Springer, Heidelberg.

[35] Galbraith, S.D., Petit, C., Shani, B., and Ti, Y.B. (2016) "On the Security of Supersingular Isogeny Cryptosystems," in Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology – ASIACRYPT 2016. LNCS, vol. 10031, pp. 63–91. Springer, Heidelberg.

[36] Gama, N., Izabachène, M., Nguyen, P.Q., and Xie, X. (2016) "Structural Lattice Reduction: Generalized Worst-Case to Average-Case Reductions and Homomorphic Cryptosystems," in Coron, J.-S., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2016. LNCS, vol. 9666, pp. 528–558. Springer, Heidelberg.

[37] Gilbert, A.C., Indyk, P., Iwen, M., and Schmidt, L. (2014) "Recent Developments in the Sparse Fourier Transform," in IEEE Signal Processing Magazine, **31**(5), 91–100.

[38] Gong, M.I., and Harn, L. (1999) "Public-Key Cryptosystems Based on Cubic Finite Field Extensions," in IEEE Transactions on Information Theory, **45**(7), 2601–2605.

[39] González Vasco, M.I., and Näslund, M. (2001) "A Survey of Hard Core Functions," in Lam, K.-Y., Shparlinski, I., Wang, H., Xing, C. (eds.) Proc. Workshop on Cryptography and Computational Number Theory 1999. Progress in Computer Science and Applied Logic, vol. 20, pp. 227–255. Birkhäuser, Basel.

[40] González Vasco, M.I., Näslund, M., and Shparlinski, I.E. (2004) "New Results on the Hardness of Diffie–Hellman Bits," in Bao, F., Deng, R., Zhou, J. (eds.) Public Key Cryptography – PKC 2004. LNCS, vol. 2947, pp. 159–172. Springer, Heidelberg.

[41] González Vasco, M.I., and Shparlinski, I.E. (2001) "On the Security of Diffie-Hellman Bits," in Lam, K.-Y., Shparlinski, I., Wang, H., Xing, C. (eds.) Proc. Workshop on Cryptography and Computational Number Theory 1999. Progress in Computer Science and Applied Logic, vol. 20, pp. 257–268. Birkhäuser, Basel.

[42] Granger, R., Page, D., and Stam, M. (2004) "A Comparison of CEILIDH and XTR," in Buell, D. (ed.) Algorithmic Number Theory – ANTS-VI. LNCS, Vol. 3076, pp. 235–249. Springer-Verlag.

[43] Granger, R., and Vercauteren, F. (2005) "On the Discrete Logarithm Problem on Algebraic Tori," in Shoup, V. (ed.) Advances in Cryptology – CRYPTO 2005. LNCS, Vol. 3621, pp. 66–85. Springer-Verlag.

[44] Gross, B.H. (1987) "Heegner Points and the Modular Curve of Prime Level," in Journal of the Mathematical Society of Japan, **39**(2), 345–362.

[45] Howgrave-Graham, N., and Smart, N.P. (2001) "Lattice Attacks on Digital Signature Schemes," in Designs, Codes and Cryptography, **23**(3), 283–290.

[46] Håstad, J., and Näslund, M. (2003) "The Security of all RSA and Discrete Log Bits," in Journal of the ACM, **51**(2), 187–230.

[47] Jao, D., and De Feo, L. (2011) "Towards Quantum-Resistant Cryptosystems from Supersingular Elliptic Curve Isogenies," in Yang, B.-Y. (ed.) Post-Quantum Cryptography – PQCrypto 2011. LNCS, vol. 7071, pp. 19–34. Springer, Heidelberg.

[48] Jao, D., Jetchev, D., and Venkatesan, R. (2007) "On the Bits of Elliptic Curve Diffie–Hellman Keys," in Srinathan, K., Pandu Rangan, C., Yung, M. (eds.) Progress in Cryptology – INDOCRYPT 2007. LNCS, vol. 4859, pp. 33–47. Springer, Heidelberg.

[49] Jetchev, D., and Venkatesan, R. (2008) "Bits Security of the Elliptic Curve Diffie–Hellman Secret Keys," in Wagner, D. (ed.) Advances in Cryptology – CRYPTO 2008. LNCS, vol. 5157, pp. 75–92. Springer, Heidelberg.

[50] Joux, A. (2000) "A One Round Protocol for Tripartite Diffie-Hellman," in Bosma, W. (ed.) Algorithmic Number Theory – ANTS-IV. LNCS, Vol. 1838, pp. 385–393. Springer-Verlag.

[51] Kiltz, E. (2001) "A Primitive for Proving the Security of Every Bit and About Universal Hash Functions & Hard Core Predicates," in Freivalds, R. (ed.) Fundamentals of Computation Theory, Proc. 13th International Symposium, FCT 2001, pp. 388–391. Springer-Verlag. Full paper available at http://homepage.ruhr-uni-bochum.de/Eike.Kiltz/papers/hash_full.pdf.

[52] Koblitz, N. (1987) "Elliptic Curve Cryptosystems," in Mathematics of Computation, **48**(177), 203–209.

[53] Konyagin, S.V., and Shparlinski, I.E. (1999) Character Sums with Exponential Functions and Their Applications. Cambridge University Press.

[54] Kushilevitz, E., and Mansour, Y. (1991) "Learning Decision Trees Using the Fourier Sprectrum," in Koutsougeras, C., Vitter, J.S. (eds.) STOC 1991, ACM, 455–464.

[55] Laity, J., and Shani, B. (2016) "On Sets of Large Fourier Transform Under Changes in Domain," under review. arXiv:1610.04330 [math.CA]. https://arxiv.org/abs/1610.04330.

[56] Lenstra, A.K., Lenstra, H.W. Jr., and Lovász, L. (1982) "Factoring Polynomials with Rational Coefficients," in Mathematische Annalen, **261**(4), 515–534.

[57] Lenstra, A.K., and Verheul, E.R. (1999) "The XTR Public Key System," in Bellare, M. (ed.) Advances in Cryptology – CRYPTO 2000. LNCS, vol. 1880, pp. 1–19. Springer, Heidelberg.

[58] Lenstra, A.K., and Verheul, E.R. (2001) "An Overview of the XTR Public Key System," in Alster, K., Urbanowicz, J., Williams H.C. (eds.) Proc. of the Public-Key Cryptography and Computational Number Theory Conference, pp. 151–180. Walter de Gruyter, Berlin.

[59] Li, W.-C.W., Näslund, M., and Shparlinski, I.E. (2002) "Hidden Number Problem with the Trace and Bit Security of XTR and LUC," in Yung, M. (ed.) Advances in Cryptology – CRYPTO 2002. LNCS, vol. 2442, pp. 433–448. Springer, Heidelberg.

[60] Lidl, R., and Niederreiter, H. (1997) Finite Fields. Cambridge University Press.

[61] Ling, S., Shparlinski, I.E., Steinfeld, R., and Wang, H. (2012) "On the Modular Inversion Hidden Number Problem," in Journal of Symbolic Computation, **47**(4), 358–367.

[62] Mansour, Y. (1992) "Randomized Interpolation and Approximation of Sparse Polynomials," in Proceedings of the 19th International Colloquium on Automata, Languages and Programming, 261–272.

[63] Maurer, U.M. (1994) "Towards the Equivalence of Breaking the Diffie-Hellman Protocol and Computing Discrete Logarithms," in Desmedt, Y.G. (ed.) Advances in Cryptology – CRYPTO '94. LNCS, vol. 839, pp. 271–281. Springer, Heidelberg.

[64] Micciancio, D., and Voulgaris, P. (2013) "A Deterministic Single Exponential Time Algorithm for Most Lattice Problems Based on Voronoi Cell Computations," in SIAM Journal on Computing, **42**(3), 1364–1391.

[65] Miller, V.S. (1986) "Use of Elliptic Curves in Cryptography," in Williams, H.C. (ed.) Advances in Cryptology – CRYPTO '85. LNCS, vol. 218, pp. 417–426. Springer, Heidelberg.

[66] Moreno, C.J., and Moreno, O. (1991) "Exponential Sums and Goppa Codes: I," in Proc. Amer. Math. Soc., **111**, 523–531.

[67] Morillo, P., and Ràfols, C. (2009) "The Security of All Bits Using List Decoding," in Jarecki, S., Tsudik, G. (eds.) Public Key Cryptography – PKC 2009: Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography. LNCS, vol. 5443, pp. 15–33. Springer, Heidelberg.

[68] Nguyen, P.Q., and Shparlinski, I.E. (2002) "The Insecurity of the Digital Signature Algorithm with Partially Known Nonces," in Journal of Cryptology, **15**(3), 151–176.

[69] Nguyen, P.Q., and Shparlinski, I.E. (2003) "The Insecurity of the Elliptic Curve Digital Signature Algorithm with Partially Known Nonces," in Journal of Cryptology, **30**(2), 201–217.

[70] Nguyen, P.Q., and Stern, J. (2001) "The Two Faces of Lattices in Cryptology," in Silverman, J.H. (ed.) Cryptography and Lattices 2001. LNCS, vol. 2146, pp. 146–180. Springer, Heidelberg.

[71] Nguyen, P.Q., and Tibouchi, M. (2012) "Lattice-Based Fault Attacks on Signatures," in Fault Analysis in Cryptography. Information Security and Cryptography, pp. 201–220. Springer, Heidelberg.

[72] Rostovtsev, A., and Stolbunov, A. (2006) "Public-Key Cryptosystem Based on Isogenies," in Cryptology ePrint Archive, Report 2006/145. http://eprint.iacr.org/2006/145.

[73] Rubin, R., and Silverberg, A. (2003) "Torus-Based Cryptography," in Boneh, D. (ed.) Advances in Cryptology – CRYPTO 2003. LNCS, vol. 2729, pp. 349–365. Springer, Heidelberg.

[74] Rubin, R., and Silverberg, A. (2004) "Using Primitive Subgroups to Do More with Fewer Bits," in Buell, D. (ed.) Algorithmic Number Theory – ANTS-VI. LNCS, Vol. 3076, pp. 18–41. Springer-Verlag.

[75] Rubin, R., and Silverberg, A. (2008) "Compression in Finite Fields and Torus-Based Cryptography," in SIAM Journal on Computing, **37**(5), 1401–1428.

[76] Schnorr, C.P. (1987) "A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms," in Theoretical Computer Science, **53**(2-3), 201–224.

[77] Shani, B. (2017) "On the Bit Security of Elliptic Curve Diffie–Hellman," in Fehr, S. (ed.) Public Key Cryptography – PKC 2017. LNCS, vol. 10174, pp. 361–387. Springer, Heidelberg.
Full paper appears in Cryptology ePrint Archive, Report 2016/1189. http://eprint.iacr.org/2016/1189.

[78] Shoup, V. (1997) "Lower Bounds for Discrete Logarithms and Related Problems," in Fumy, W. (ed.) Advances in Cryptology – EUROCRYPT '97. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg.

[79] Shparlinski, I.E. (2001) "On the Generalised Hidden Number Problem and Bit Security of XTR," in Boztas, S., Shparlinski, I.E. (eds.) Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. LNCS, vol. 2227, pp. 268–277. Springer, Heidelberg.

[80] Shparlinski, I.E. (2001) "Sparse Polynomial Approximation in Finite Fields," in Proc. 33rd ACM Symposium on Theory of Computing – STOC 2001, pp. 209–215. ACM, New York.

[81] Shparlinski, I.E. (2004) "Security of Polynomial Transformations of the Diffie–Hellman Key," in Finite Fields and Their Applications, **10**(1), pp. 123–131.

[82] Shparlinski, I.E. (2005) "Playing "Hide-and-Seek" with Numbers: The Hidden Number Problem, Lattices and Exponential Sums," in Garrett, P., Lieman, D. (eds.) Public-Key Cryptography; Proceedings of Symposia in Applied Mathematics, vol. 62, AMS, pp. 153–177.

[83] Shparlinski, I.E., and Winterhof, A. (2003) "A Hidden Number Problem in Small Subgroups," in Mathematics of Computation 2005, vol. 74, pp. 2073–2080.

[84] Shparlinski, I.E., and Winterhof, A. (2004) "A Nonuniform Algorithm for the Hidden Number Problem in Subgroups," in Bao, F., Deng, R.H., Zhou, J. (eds.) Public Key Cryptography – PKC 2004. LNCS, vol. 2947, pp. 416–424. Springer, Heidelberg.

[85] Silverman, J.H. (2009) The Arithmetic of Elliptic Curves. Graduate Texts in Mathematics, vol. 106 (2nd edition). Springer-Verlag, New York.

[86] Smith, P., and Skinner, C. (1995) "A Public-Key Cryptosystem and a Digital Signature System Based on the Lucas Function Analogue to Discrete Logarithms," in

Pieprzyk, J., Safavi-Naini, R. (eds.) Advances in Cryptology – ASIACRYPT '94. LNCS, vol. 917, pp. 355–364. Springer, Heidelberg.

[87] Stam, M. (2003) "Speeding up Subgroup Cryptosystems." Ph.D. Thesis, Technische Universiteit Eindhoven.

[88] Stinson, D.R. (2005) Cryptography: Theory and Practice. Discrete Mathematics and Its Applications (3rd edition). CRC Press.

[89] Stolbunov, A. (2010) "Constructing Public-key Cryptographic Schemes Based on Class Group Action on a Set of Isogenous Elliptic Curves," in Advances in Mathematics of Communications, **4**(2), 215–235.

[90] Terras, A. (1999) Fourier Analysis on Finite Groups and Applications. London Mathematical Society Student Texts (No. 43), Cambridge University Press. Cambridge.

[91] Verheul, E.R. (2000) "Certificates of Recoverability with Scalable Recovery Agent Security," in Imai, H., Zheng, Y. (eds.) Public Key Cryptography – PKC 2000. LNCS, vol. 1751, pp. 258–275. Springer, Heidelberg.

[92] Vinogradov, J.M. (1927) "On a General Theorem Concerning the Distribution of the Residues and Non-Residues of Powers," in Transactions of the American Mathematical Society, **29**(1), 209–217.

[93] Wang, Y., and Lv, K. (2016) "The Security of Polynomial Information of Diffie–Hellman Key," in Information and Communications Security ICICS 2016, pp. 71–81.

[94] Wang, M., Zhan, T., and Zhang, H. (2016) "Bit Security of the CDH Problems over Finite Fields," in Selected Areas in Cryptography – SAC 2015, pp. 441–461.

[95] Xu, J., Hu, L., Huang, Z., and Peng, L. (2014) "Finding Small Solutions of a Class of Simultaneous Modular Equations and Applications to Modular Inversion Hidden Number Problem and Inversive Congruential Generators," in Cryptology ePrint Archive, Report 2014/843. http://eprint.iacr.org/2014/843.

[96] Zhang, Fangguo (2015) "Bit Security of the Hyperelliptic Curves Diffie–Hellman Problem," in Cryptology ePrint Archive, Report 2015/614. http://eprint.iacr.org/2015/614.