On General Multi-Quadratic Function Field Extensions in the GHS Attack

Ahmad Lavasani

A Thesis

in the Department of

Mathematics and Statistics

Presented in Partial Fulfillment of

the Requirements for the Degree of

Master of Science(Mathematics)

at

Concordia University

Montreal, Quebec, Canada

July 2008

Your file Votre référence
ISBN: 978-0-494-45468-8
Our file Notre référence
ISBN: 978-0-494-45468-8

Abstract

On General Multi-Quadratic Function Field Extensions in the GHS Attack

Ahmad Lavasani

To date, elliptic curves offer the most efficient cryptographic solution. Particularly efficient among elliptic curves, are those defined over binary composite finite fields, such as $GF((2^r)^n)$. These curves were no longer considered secure when, in 1998, Gerhard Frey innovated a concept which paved the road for the GHS attack. The idea behind the GHS attack is to map the Discrete Logarithm Problem (DLP) over such a curve to an equivalent DLP over the jacobian of another curve, defined over the smaller field $GF(2^r)$.

In this thesis, we study the theoretical structure of the GHS attack for elliptic curves defined over fields of arbitrary characteristics. We study the GHS attack using general quadratic extensions for elliptic curves defined over composite fields of even characteristic and we estimate the genus of resulting function field. We also implement the GHS attack and present some computational results.

Keywords: GHS Attack, Elliptic Curve Cryptography, Function Fields.

Résumè

Sur les extensions multi-quadratiques des corps de fonctions dans l'attaque GHS

Ahmad Lavasani

Jusqu'à present, les courbes elliptiques offrent la solution cryptographique la plus efficace. En particulier, les courbes elliptiques qui sont définies sur des corps binaires comme $GF((2^r)^n)$ sont encore plus efficace. Ces courbes ne sont plus considérées securitaires après l'innovation de l'idée de l'attaque GHS par Gerhard Frey en 1998. L'idée principale de l'attaque GHS est de transférer le problème du logarithme discret (PLD) sur une courbe elliptique de cette famille à un PLD équivalent sur la jacobienne d'une autre courbe qui est définie sur le corps le plus petit $GF(2^r)$.

Dans ce mémoire, nous étudions la structure théorique de l'attaque GHS pour les courbes elliptiques définies sur les corps de n'importe quelle caractéristique. Nous étudions l'attaque GHS en utilisant des extensions quadratiques générales pour les courbes sur les corps de caractéristique paire et nous estimons le genre de la courbe. Nous présentons aussi le résultat de nos implémentations de l'algorithme de GHS.

Mots Cles: Attaque GHS, Cryptographie sur les Courbes Elliptiques , Corps de Fonctions.

To

My mother, FSM and Pickle

Preface

In 2005 I was a research assistant of Prof. Taraneh Eghlidos in Cipher Lab of Sharif ERC. I was assigned the task of comparing the efficiency of the arithmetic of different elliptic curves for processors with limited resources. The result of my implementation was unanimously in favor of elliptic curves over finite field extensions with composite degree. I wrote a report and I recommended them for their efficiency [ELS06]. A week later, after submitting my report, I leafed through a guide for choosing elliptic curves for cryptography. It was suggested in the guide, that those finite fields are not recommended for cryptographic purposes, as they are subject to Weil Descent attack. It is not a good feeling to submit something and then find evidence against it. I felt a mix of sadness and curiosity. I was determined to know what this attack was, that ruined the pleasure of using those nice curves. I immediately submitted a research proposal to Dr. Eghlidos to study the Weil Descent attack. I should thank her for accepting it and for her support in that research [EGL07]. In the short period of time that I had, I got acquainted with the attack, but I had still a long way to study it.

When I began my Master's in Concordia, I didn't feel like leaving my research half unfinished. I talked to my supervisor, Prof. Chantal David, about my personal attachment to GHS attack. She was very kind to accept my proposal, even though she had another proposal for me and the GHS attack was not her main interest. I would like to thank her, not only for accepting my proposal, but also for her continuous support; for reading papers with me line-by-line and for going through the proofs together, for her help in solving my financial problems, and for reading and correcting my messy

thesis drafts line by line. Above all, I should thank her for thinking with me about the problems I faced, which I think is very important point.

I also should thank Ann-Marie Agnew, our kind graduate assistant of the Math Department of Concordia University, for her continuous support during my two years of Master's study.

I am thankful to the other members of the thesis committee: Prof. David Ford, Prof. Hershy Kisilevsky and Prof. Francisco Thaine, who agreed to evaluate my work under time pressure.

On the path to preparing this work I am indebted to several people who, either listened to my problems and/or suggested solutions, including: Ferenc Balogh, Marco Bertola, Noam Elkies, Florian Hess, Jorge Jimenez Urroz and Nicolas Thériault.

I should especially thank my linguist friend Natalie Kershaw who, by helping me with the grammar of this thesis, also proved her theory in the disability of the human brain in learning new language after losing its neuro-plasticity. I should also thank Valérie Hudon for helping me with the french abstract.

I would like also to thank my uncle, Abbas Beheshti and my aunt for helping me financially in the application and visa processes.

Last but not least, I would like to express my appreciation to Pickle who almost constantly kept me company while I was writing my thesis, even though occasionally I had to bribe him with pieces of carrot to keep him interested.

Contents

List of Algorithms

List of Diagrams

CHAPTER 1

Introduction

1.1 History

1.1.1 Algebraic geometry

In the ancient past, it was too difficult for the Greeks to look at algebra and geometry as two different fields of mathematics since they were thinking about algebra "geometrically". Rather than "calculating" with numbers they were "measuring" magnitude and searching for relations. Multiplying two values meant computing the area that two "lengths" make. Therefore the Greeks considered area as a magnitude of a different type. This made the notion of polynomials not very meaningful for them [DD85].

However, this geometric view of algebra helped them to excel in algebraic geometry in that very early age. The intensive usage of the geometric methods can be found in "Konika" of Apollonius of Perga (262-190 BCE) in his thorough study of conic sections. Half a millennium later, Diophantus of Alexandria (about 200- about 284) wrote his series of *Arithmetica*. He was aware of the group operation on the points of elliptic curves and used this operation to find new rational points on these curves. This was probably the first time in history that elliptic curves appeared in the literature [vdW83].

Omar Kayyam (1048-1122) was the first one who made a clear distinction between "Numerical Algebra" and "Geometrical Algebra" and tried to prove some of Euclid's theorems using purely numerical quantities. Nevertheless, his algebra was mainly geo-

metric; for example, he solved cubic equations by intersecting conics [DD85]. Following Kayyam, algebra and geometry started to grow in different directions. Mathematicians started avoiding geometric methods when studying algebraic problems. In this way, the reunification of algebra and geometry was postponed until the 19th century.

In the late 18th and early 19th century, prominent mathematicians such as Leonhard Euler (1707-1783), Joseph Lagrange (1736-1813) and especially Adrien-Marie Legendre (1752-1833) tried unsuccessfully to find solutions for indefinite elliptic integrals. Their failed efforts did, however, lead to the theory of "elliptic functions", which are double periodic functions on complex plane. These functions were the first instance of "abelian varieties" later named as such by Solomon Lefschetz(1884-1972). Niels Abel (1802-1829) and Carl Jacobi (1804-1851) independently generalized this theory for higher periods. These works led to the invention of another class of abelian varieties called "jacobian of hyperelliptic curves" which are one of main objects studied in our work.

During the 19th century, algebraic methods in geometry made considerable progress, mainly due to the work of Bernhard Riemann (1826-1866) and his successors [DD85]. Algebraic geometry was reborn as mathematicians started to use geometric methods to study "abelian functions" again. It was Weil in the 1940s who gave abelian varieties their modern foundations in the language of algebraic geometry.

Abelian varieties are geometric objects, such as curves and surfaces, which are algebraic groups at the same time. At the end of the 20th century, abelian varieties became the intersection point between an ancient branch and a recent branch of mathematics: algebraic geometry and cryptography.

1.1.2 Cryptography

Less than 3 decades after the birth of cryptography as a science through Claude Shannon's seminal paper [Sha48] on information theory, cryptography reached an inflection

point. Martin Hellman discovered that a public-key cryptosystem would be possible if there exists a problem which is hard in one way and easy in other way. A Public-key cryptosystem is a system which allows everybody to send an encrypted message to Alice that no one else can decrypt but Alice. It is sometimes called asymmetric cryptography. Ralph Merkle, suggested to Hellman that the "Discrete Logarithm Problem (DLP)" in $\mathbb{Z}/p\mathbb{Z}$ is a good choice [DH76].

Since then, the DLP has become a popular problem in designing public key cryptosystems. The popularity of the DLP in cryptography motivates both cryptographers and cryptanalysts. While cryptographers have been searching for groups in which solving the the DLP is hard, cryptanalyzers, on the other hand, have been trying to find faster algorithms to solve the DLP on those groups. It was there that algebraic geometry and cryptography were tied together.

As abelian varieties provide a rich supply of algebraic groups, they attracted the attention of the cryptographers in their search for harder instances of the DLP. Neal Koblitz [Kob87] and Victor Miller [Mil86] (independently) suggested using the group of points of an elliptic curve over a finite field. The contribution of algebraic geometry to cryptography, however, is wide-reaching and did not stop at that point. Koblitz later also proposed the use of the jacobian of a hyperelliptic curve for the same purpose [Kob89]. Other examples are torus-based cryptography proposed by Alice Silverberg and Karl Rubin [RS03] or the usage of a generalized jacobian by Isabelle Déchène [Dec05].

On the other side of the race, cryptanalysts were trying to develop algorithms to solve the DLP over different groups. Victor Shoup proved [CFA06] that a generic DLP solver (according to the definition of generic DLP solver in Theorem 19.2 [CFA06]) can not be less complex than $O(\sqrt{\text{Group Size}})$. To solve the DLP faster, cryptanalysts started to look for algorithms which exploit the structure of the underlying group. At

the end of the 70's, the index calculus algorithm was introduced for solving the DLP over the multiplicative group of $\mathbb{Z}/p\mathbb{Z}$ in subexponential time, based on the idea of A. E. Western and J. P. C. Miller. This idea can be generalized to the multiplicative group of any finite field. Later in 1999, Leonard Adleman, Jonathan DeMarrais and Ming-Deh Huang designed the index calculus algorithm for jacobians of hyperelliptic curves [ADH94]. To date, index calculus algorithms are the strongest algorithms for solving the DLP in many groups.

In spite of the successful application of the index calculus attack to many groups, there are still some groups on which this attack is not applicable. The failure to find a fast algorithm for solving the DLP on these groups has been the motivation for studying isomorphism attacks. These attacks try to find an efficiently computable map from a group with no known fast DLP solver to another group which admits a fast DLP solver such as index calculus. In the presence of such a map, one can solve the DLP in the range group to solve the DLP in the domain.

The group of points on a general elliptic curve is a prominent example of groups for which no successful subexponential DLP solver has been found yet. As a result, in order to solve the DLP problem on elliptic curves, mathematicians try the option of the isomorphism attack. Two examples of an isomorphism attack which are applicable to special groups of elliptic curves, are [MVO91] and [Sma99], however neither of these is applicable to a significant number of curves.

1.1.3 The GHS Attack

The difficulty of the elliptic curve DLP (ECDLP) made elliptic curves very interesting objects for cryptography. Among all kinds of elliptic curves, the cryptographers were particularly interested in curves defined over binary finite fields with a composite extension degree , i.e. $GF((2^r)^n)$, because of the fast algorithms existing for the arithmetic on

these fields. Examples of practical interest in these curves can be found in the internet protocol standard RFC4212 [Orm98] and also in [PSR97], [GP97], [PFR98], [AMV93] and [PFSR99].

As the elliptic curves over composite finite fields were gaining popularity, Gerhard Frey took mathematical cryptography to a higher level of sophistication by opening a new trend of isomorphism attacks for elliptic curves over composite fields [Kob07]. He proposed a method to map the DLP over elliptic curves defined over composite fields to the DLP over a higher-dimensional abelian variety, which is called the Weil restriction of the curve. In this way solving the DLP over the resulting variety became equivalent to solving the DLP over the curve [Fre98].

In [GS99], Nigel Smart and Steve Galbraith studied different aspects of the method in the case of characteristic 2. They proposed a way to find an algebraic curve over the Weil restriction and a way to transform the DLP from the elliptic curve to the new curve. They experimentally discovered that the resulting curve is usually hyperelliptic.

Later Pierrick Gaudry, Florian Hess and Smart used the idea of [GS99] but worked with the function fields of the curves instead of the curves themselves. The basic idea in [GHS02b] was to extend the function field of the elliptic curve iteratively to reach a function field corresponding to a curve suitable for an index calculus attack. This became known as the GHS attack. The function field is extended with a series of Artin-Schreier extensions and the authors were able to prove that the function field resulting from the extension process belongs to a hyperelliptic curve. As the index calculus for the jacobian of an hyperelliptic curve is a well studied problem [CFA06], the GHS attack provides a way to solve the DLP over such elliptic curves with exactly computable complexity.

The method of [GHS02b], although it was applicable to elliptic curves defined over the majority of composite fields, resulted in an easier DLP only for a small proportion

of these curves. This difficulty arises because most of the time, the genus of the extended function field is very large, which means that the jacobian of its defining curve would be large. In such situations, the subexponential index calculus algorithm for this jacobian cannot out-perform the exponential algorithm over the original elliptic curve. In [GHS02a] Galbrith, Hess and Smart devised a method to solve this problem to some extent. If the elliptic curve under attack results in an unreasonably large genus curve, the method tries to find an isogenous elliptic curve which results in a lower genus under the GHS method.

Hess in [Hes04] showed that using more general forms of Artin-Schreier extensions can improve the performance of the attack by finding a function field of smaller genus. However, when one uses this general form of the GHS attack, there is no guarantee that the resulting curve is hyperelliptic. As the complexity of the index calculus attack for a general (non-hyperelliptic) curve is not a well-studied subject, the exact complexity of the attack proposed in [Hes04] cannot be determined.

1.2 Outline

In the generalized GHS attack of [Hes04], Hess used Artin-Schreier extensions to build the extended function field. However, by choosing other extensions for special elliptic curves, we might end up at a less complex DLP using the GHS construction.

In this work, our main goals are to explain the GHS attack as it is introduced in [Hes04] and [BSS05] with more emphasise on studying the function field structure of the attack. We want also to study the GHS attack when general quadratic extensions are used to construct function fields. We also propose a way to rank the elements of a finite field according to their security against GHS attack.

In Chapter 2, we present the jacobian of an algebraic curve and the DLP problem over it. We also present the index calculus algorithm for the jacobian of curves. Finally,

we discuss the applicability and complexity of this algorithm to the jacobian of different curves to rank the difficulty of the DLP over them.

In Chapter 3, we explain the general algorithm of the GHS attack as it is explained in [Hes04], [Die03] and [BSS05]. We discuss each step of the attack separately and we explain the algebraic properties of the function fields we encounter in the attack. We also discuss conditions under which the attack can be carried out theoretically (independent of its efficiency).

In Chapter 4, we limit ourselves to elliptic curves over fields of even characteristic. In this chapter we want to evaluate the security of different field extensions against the GHS attack. However, unlike [Hes04], we do not limit ourselves to the Artin-Schreier extensions and we generalize some of results related to the complexity of GHS attack proved in [Hes04], [BSS05], [MQ01] and [MT06] for more general quadratic extensions. Using this information, we study the security of elements of finite fields with prime extension degree. We also propose an algorithm to find weak elements of a field against GHS attack.

In Chapter 5, we include and explain our implementation both for the attack and for the weak element finder. We present some examples of running these implementations and we discuss their results. We also present the evaluation of security of finite extensions of prime degree of even characteristics finite fields.

In Chapter 6, we discuss the result and the impact of the GHS attack on the cryptographic world. We also discuss future work that can be done to improve the GHS attack in more general situations.

The beauty of the GHS attack, independent of its application to the real world, is in its deep interaction between two exciting fields of mathematics, namely algebraic geometry and cryptography.

CHAPTER 2

Discrete Logarithm Problem over Jacobian of Curves

The problem of solving the logarithm can be discussed in any group. However, the difficulty of solving discrete logarithm vs. the efficiency of cryptographic algorithms guides cryptographers to choose one group over another.

Miller and Koblitz independently proposed the idea of using the group of points on an elliptic curve in cryptography [Mil86][Kob87]. Unfortunately, the fantastic property that the points on elliptic curves form a group, does not hold for other kind of curves. Nonetheless we can assign to any curve a group called the jacobian. In all the following, a "curve" will be a non-singular complete, projective curve as defined in Definition V.10.3 [Lor96].

The index calculus attack provides a subexponential algorithm to solve the DLP for the jacobian of curves of large genus in several cases. The idea behind the GHS attack is to map the elliptic curve discrete logarithm problem(ECDLP) to the jacobian of another curve and try to break the new DLP using index calculus attack.

In this chapter, we define and study the jacobian group and we study the DLP on the jacobian. We explain the index calculus attack and the different ways that index calculus strategies can be applied to different types of jacobians.

2.1 Divisor class group and jacobian

The set of points on an elliptic curve form an abelian group. The group structure is based on the fact that every straight line meets the curve in 3 (not necessarily distinct) points P_1, P_2, P_3 in the projective plane. Looking at each line and defining $P_1+P_2+P_3 = 0$, the addition of two points can be defined as $P_1 + P_2 = -P_3$. This fact comes from the famous Bézout theorem in algebraic geometry:

Theorem 2.1.1 (Corollary I.7.8 in [Har97]) Bézout theorem *Let C_1 and C_2 be two distinct curves in $\mathbb{P}^2$, having degrees d_1, d_2 respectively and let $C_1 \cap C_2 = \{P_1, \ldots, P_n\}$. Then*

$$\sum_{j=1}^{n} i(C_1, C_2 : P_j) = d_1 d_2$$

where $i(C_1, C_2 : P_j)$ is the multiplicity of intersection of C_1 and C_2 at P_j.

For curves of higher genus (and then higher degree), instead of considering each point as an element of the group, we consider the formal sum of points as an element of the group.

Therefore any formal sum of points would be an element of our group. Each instance of such formal sum of points is called a divisor:

Definition 2.1.2 Divisor (Over a smooth curve): *The divisor group of a curve C, denoted by $Div(C)$ is the free abelian group generated by the points of C. Thus if $P_1, ..., P_n$ be points on curve C and $n_i \in \mathbb{Z}$, then $D := \sum_{i=1}^{n} n_i P_i$ is called a divisor over C.*

Similar to the case of elliptic curve where the lines that intersect the curve were defining the relations on the group, we need to defines the same kind of relations over this group to obtain a group which is useful for our purpose. However, we need to

consider more general objects than just straight lines to have a well-defined group. These general objects are rational functions defined in the function field of the curve.

Definition 2.1.3 ([Sil94]) Function field and rational functions *Suppose that K is a field and C is a curve defined by irreducible equation $p(x,y) \in K[x,y]$. The integral domain $K[x,y]/p(x,y)$, denoted by $K[C]$, is called the coordinate ring of C. The field of fractions of $K[C]$, denoted by $K(C)$, is called the function field of C. Elements of $K(C)$ are called rational functions.*

In this thesis we only consider the case that K is a finite field. This assumption holds for all cryptographic applications.

We consider the intersection points of each rational function in the projective plane with the curve as a defining relation for the group. In this way we can define the divisor class group as the quotient of $Div(C)$ by one of its subgroup (defined below).

Definition 2.1.4 (Definition I.4.3 [Sti93]) subgroup of principal divisors *To each rational function $f(x,y)$ in $K(C)$, we associate a divisor*

$$div(f) = \sum_{P_i \in C} v_f(P_i) P_i$$

where $v_i(P_i)$ is the signed multiplicity of P_i over f (either as a pole or zero). Such a divisor is called a principal divisor. The set of principal divisors forms a group which we denote by $\mathcal{P}_{K(C)}$.

Definition 2.1.5 Divisor Class Group *The divisor class group of C is*

$$Cl(C) := Div(C)/\mathcal{P}_{K(C)}.$$

We denote the divisor class represented by divisor D, by $[D]$.

Definition 2.1.6 (I.3 [Sil94]) Degree of a Divisor *The degree of a divisor* $D = \sum_{P_i \in C} v_i P_i$ *is*

$$\deg(D) := \sum_{P_i \in C} v_i$$

The degree of a principal divisor is 0 (See II.6.10 [Har97]) and therefore the degree depends only on $[D]$ and not on the representing divisor D. This mean that the degree is well-defined over $Cl(C)$ as well.

Definition 2.1.7 Jacobian of C *For curve* C *we define following set as "The degree 0 part of divisor class group" as follows:*

$$Cl^0(C) := \{[D] \in Cl(C) | \deg([D]) = 0\}$$

Till now, to define a divisor we have considered points on C with coordinates defined in $\bar{K}$, the algebraic closure of the constant field of C. For cryptographic purpose, however, it is important to confine ourselves for the divisor which are defined over the constant field.

Definition 2.1.8 Divisor defined over K *We say that divisor* $D := \sum_{i=1}^{n} n_i P_i$ *is defined over* K *if*

$$D^\sigma = \sum_{i=1}^{n} n_i P_i^\sigma = D \text{ for all } \sigma \in Gal(\bar{K}/K)$$

In this way we define the jacobian of C defined over K as follows:

Definition 2.1.9 *The jacobian of curve* C *defined over* K *denoted by* $Cl^0(K(C))$*, is a subgroup of* $Cl^0(C)$ *which contains all divisor class* $[D]$ *such that* $[D]$ *is equivalent to a divisor class* $[D_0]$ *such that* D_0 *is defined over* K*.*

It can be shown that $Cl^0(K(C))$ is a finite subgroup of $Cl(C)$ and therefore one can study the DLP on this group.As the jacobian of curve C defined over K is the finite

group that we are always interested in, from now on we simply call it the jacobian of C.

2.2 Complexity of the DLP over the jacobian of a curve

The complexity of the general attack to solve a DLP over a general cyclic group depend only on the size of the group. Indeed, the best general attack known for solving the DLP over a general cyclic group is Pollard's ρ algorithm [MVO96]. The complexity of the attack is $O(\sqrt{n})$ in which n is the order of the group. This is better than the $O(n)$ exhaustive search, but it is still is exponential in the bit-length of n. As the complexity of the general attack only depends on the group size it is important to estimate the size of the group whose security is to be analyzed.

As we said, estimating the size of the jacobian of a curve is the first step in evaluating its security. Corollary VIII.6.3 [Lor96] gives an estimate for the size of $Cl^0(K(C))$.

Theorem 2.2.1 *Suppose C is a curve of genus g, defined over a finite field K of size q. Then:*

$$(1 - \sqrt{q})^{2g} \leq |Cl^0(K(C))| \leq (1 + \sqrt{q})^{2g}$$

Using the above theorem we conclude that the size of the jacobian of a curve is of order q^g. Consequently, the complexity of the Pollard's ρ attack for the jacobian of curve C is $O(q^{g/2})$. This means that the best known general attack to the DLP over $Cl^0(K(C))$ is exponential in term of its genus. Actually this was one of the advantages of hyperelliptic curve cryptography over elliptic curve cryptography [CFA06].

By studying the size of the group $Cl^0(K(C))$, we dealt with assessing the resistance of the DLP over this group against general attacks. However, the strongest attack in many practical cryptographic intersting situations is the index calculus attack. For

this reason in the remainder of this section, we study the index calculus attack and its applicability to the $Cl^0(K(C))$ for different curves C.

2.3 Index calculus attack

The family of index calculus algorithms are the most powerful algorithms known to solve the DLP over different groups. In many groups, including the multiplicative group of finite fields, the index calculus method gives a subexponential algorithm to solve any instance of the DLP.

Despite of all successes of the index calculus attack during recent years, there is still no index calculus attack to solve the DLP over a general elliptic curve in subexponential time. In the next sections, we describe the index calculus algorithm in general and for specific groups.

2.3.1 The algorithm

Let $G =< P >$ be an additive cyclic group and suppose that $Q \in G$ is given. The DLP is to find n such that

$$Q = nP. \tag{2.1}$$

As we mentioned before, index calculus is not applicable to a generic cyclic group. The group needs a special property which we call the factor base property.

2.3.1.1 Factor base property

There is a formal technical definition for the factor base, which is beyond the scope of this work. We refer the interested reader to [CFA06]. However, we give a more intuitive definition here parallel to the idea of [HMV03].

Definition 2.3.1 Factor base *Let G be an abelian group. A factor base $B = \{P_1, \ldots, P_m\} \subset G$ for the index calculus algorithm is a (finite) subset of G such that a randomly chosen element $Q \in G$ can be efficiently expressed in terms of element of B as*

$$Q = \sum_{i=1}^{m} n_i P_i \tag{2.2}$$

with a significant probability.

The elements in the factor base should be chosen in a way to increase the efficiency mentioned in Definition 2.3.1. The word "efficiently" in the above definition could be defined precisely. However, we shall only give an intuitive idea of what efficiency means here. For a given factor base, and for several random Q's, we want to find m values of n_i's such that $Q = \sum_{i=1}^{m} n_i P_i$. Therefore, the elements of the factor base should be special elements such that this operation is "efficiently" possible and it should be possible for a "significant" number of randomly chosen Q's.

The elements of the factor base are called *primes* as the prime elements of $\mathbb{Z}$ were used as factor base elements to solve the DLP in $(\mathbb{Z}/p\mathbb{Z})^*$ when index calculus was applied for the first time. An element $Q \in G$ is called B-smooth if it is expressible in terms of elements of factor base B as in (2.2).

We say that a group G has the factor base property if there is a fast algorithm to find an *efficient* factor base for the DPL problem in G.

The efficiency of a factor base B depends on its "size" and the "probability of finding a B-smooth element". There is a natural trade-off between the size and the probability that a random element is B-smooth. The greater the factor base, the more likely that a random element is B-smooth. On the other hand, the index calculus needs to gather more relations (as defined in following section) to solve the DLP and this decreases

the efficiency of the algorithm. However, the smaller the factor base, the smaller the probability that a randomly chosen Q be a B-smooth. If this probability is low, the index calculus algorithm wastes time on choosing a lot of useless elements until it finds enough B-smooth elements. Therefore the definition of an efficient factor base is group-dependent or even implementation-dependent.

The problem of finding an efficient factor base for a specific group is non-trivial. In other words for each group, we need a specific method to find an efficient factor base. For the case $(\mathbb{Z}/p\mathbb{Z})^*$, it is easy to prove that the primes form an efficient factor base. On the other hand, in the case of the group of points on a general elliptic curve, the question has still not been solved.

Under the assumption of having an efficient factor base, the index calculus algorithm can be applied to any group without difficulty. Algorithm 2.1 describes the index calculus for a general additive group when the factor base is found:

Algorithm 2.1 INDEX CALCULUS - GIVEN THE FACTOR BASE

Require: : P, Q and $B := \{P_1, \ldots, P_m\}$ an efficient factor base.
Ensure: : Find n such that $Q = nP$.

1: **for** i in $\{1, \ldots, m+N\}$ such that N is a small integer **do**
2: **repeat**
3: Generate (Regenerate) random b_i
4: **until** b_iP is $B-$smooth
5: $Q_i \leftarrow b_iP$
6: Compute $e_{i,j}$ for $1 \leq j \leq m$ such that $Q_i = b_iP = \sum_{j=1}^{m} e_{i,j}P_j$
7: **end for**
8: $A \leftarrow (e_{i,j})$
9: $b \leftarrow (b_i)$
10: Solve the linear system $Ax = b$ (Now we have x_j such that $P_j = x_jP$.)
11: **repeat**
12: Generate random r
13: **until** rQ is $B-$smooth
14: Compute r_j for $1 \leq j \leq m$ such that $rQ = \sum_{j=1}^{m} r_jP_j = \sum_{j=1}^{m} r_jx_jP$
15: $n \leftarrow r^{-1}(\sum_{j=1}^{m} r_jx_j)$

The N extra B-smooth elements are chosen in the loop in Line 1 of Algorithm 2.1 to make sure that the linear system is solvable with high probability. The Line 15 of Algorithm 2.1, is valid because $Q = nP$, so for a random r, we have $rQ = rnP = (\sum_{j=1}^{m} r_j x_j)P$.

The basic idea of the algorithm is as follows: First one can solve the DLP for the factor base quite easily and then using that information one can solve the DLP for the specific element Q. As the index calculus is applicable as soon as a factor base is found for a specific group, the question of solvibility of the DLP in a specific group using index calculus boils down to the question of finding a suitable factor base in that group. In the following sections, we discuss the problem of finding a suitable factor base in the groups that will be involved in the GHS attack.

2.4 Index calculus for the DLP over the jacobian of curves

In order to find a suitable factor base we need to study the structure of $Cl^0(K(C))$. In this section we first discuss the problem of finding the factor base for a special family of curves called "hyperelliptic curves" [CFA06]. Then we explain how to find the factor base on a general curve. At the end, we discuss the case where a factor base cannot be found with current knowledge.

2.4.1 Jacobian of hyperelliptic curves

In the case of hyperelliptic curves we have a nice representation for the elements of divisor class group. This representation called the *Mumford representation*, can be defined according to Proposition 2.4.1:

Proposition 2.4.1 (VII.1 [BSS05]) *Let C be a hyperelliptic curve of genus g_C defined over a field K with equation $Y^2 + H(X)Y = F(X)$. Then the elements of the*

jacobian of C that are defined over K are in one-to-one correspondence with the pairs of polynomials $(a(X), b(X))$ with coefficients in K, such that $deg(b) < deg(a) < g_C$ and such that the polynomial a is monic and a divides $b^2 + bH - F$.

We denote the divisor class that corresponds to a pair $(a(X), b(X))$ by $div(a, b)$. The factorization of the polynomial $a(X)$ is compatible with the divisor group operation and this leads to the idea of defining a prime element in the divisor class group of a hyperelliptic curve

Definition 2.4.2 *Let $[D]$ be the divisor class corresponding to $div(a, b)$ by Proposition 2.4.1. $[D]$ is said to be prime if the polynomial a is irreducible over K.*

Under this condition an efficient factor base can be defined for $Cl^0(K(C))$

Definition 2.4.3 *Let B be an integer. The set of all divisors $div(a, b)$ such that a is irreducible in K and is of degree $\leq B$, make a factor base for index calculus on $Cl^0(K(C))$. A divisor $D \in Cl^0(K(C))$ is B-smooth if it can be written in terms of element of this factor base.*

In [BSS05] it is shown that if the genus of the hyperellptic curve C is large enough (compared to the size of K), using above factor base, index calculus can provide a subexponential attack.

2.4.2 Jacobian of a general curve

As it is mentioned in Definition 2.1.2, a divisor in $Cl^0(K(C))$ can be represented as the sum of finite many points. This idea is parallel to the idea of primes in $\mathbb{Z}$. In $\mathbb{Z}$ every integer can be uniquely written as a product of primes. In the same way, we can write each divisor as a sum of closed points or prime divisors [Die08]. In the index calculus algorithm, we expect that the elements of factor base belong to the group

under attack. However, for a prime divisor class $[P]$ we may have $\deg(P) \neq 0$ and therefore $[P] \notin Cl^0(K(C))$. To cure this problem, we fix a divisor class $[D_0]$ of degree 1 and whenever we face such a problem we consider divisor class $[P] - \deg(P)[D_0]$ in the factor base instead.

Using this factor base, it is shown in [Die08] that the index calculus algorithm is subexponential in the genus of curve C as the genus of C approaches infinity.

2.4.3 Infeasibility of index calculus method for elliptic curves

Despite of several attempts, there is still no practical factor base for the DLP on elliptic curves.

The factor base of Definition 2.4.3 is useless when one looks at an elliptic curve as a hyperelliptic curve of genus 1. With the Mumford representation, all divisors in the class group of an elliptic curve have $a(X)$ polynomial (in the $\mathrm{div}(a, b)$ representation of the divisor) of degree $\leq g = 1$. This means that they all are irreducible polynomials by definition. Therefore all the divisors in the class group are prime and should be employed in the factor base which is absurd.

Looking at elliptic curves as general curves also does not help. As stated in Theorem III.3.4 [Sil94], for elliptic curve E, we have $E \cong Cl^0(E)$. Therefore, all elements of $Cl^0(E)$ are primes, which is the same problem as the case of hyperelliptic factor base. Another approach is to lift the elliptic curve defined over a finite field to an elliptic curve defined over $\mathbb{Q}$ and search for the factor base elements there. The Xedni-Calculus attack mentioned in [Sil00] is an example of such an effort.

However, in contrast to the fact that lifting an element of $(\mathbb{Z}/p\mathbb{Z})^*$ to $\mathbb{Q}$ is a trivial fact, it is not the case for points on an elliptic curve. It is claimed in [HMV03] that this approach fails because there is no efficient algorithm to lift a point from an elliptic curve defined over a finite field to $\mathbb{Q}$. In addition we are only able to lift those points

to $\mathbb{Q}$ efficiently whose canonical height over lifted curve (over $\mathbb{Q}$) is small. However it is proved that the number of such points on a general elliptic curve is negligible.

Another possible approach is lifting the curve to a local field as well [Gau04].

However, [Gau04] introduced a factor base for the same family of curves that we study in this thesis. The method is based upon Weil Descent attack which is also a basis for the GHS attack. Although the method is less efficient than the GHS attack, it can be applied in more general cases.

The lack of a general index calculus attack for elliptic curves was the primary motivation for mapping the group of points of elliptic curves to other groups were the index calculus is feasible. The GHS attack which we discuss in next chapter is an example of such attack.

CHAPTER 3

GHS Attack and Its Function Field Structure

In previous section, we studied the DLP over different curves. We also studied the attacks which are applicable to them. The difference in the complexity of the DLP solvers over different groups, encourages studying the so-called "isomorphism attacks" [HMV03]. The isomorphism attacks try to find efficiently computable maps between a group with a "hard" instance of the DLP and an "easier" one. Obviously, the underlying instance of the DLP should stay invariant under these maps. Weil [MVO91] and Tate [FR94] pairing attacks are some examples of this category.

The purpose of this work is to study another instance of this category of attacks, called the *GHS Weil Descent attack*. The primary motivation for designing the GHS attack was the absence of subexponential algorithm to solve the DLP over the group of points of an elliptic curve. In this attack, we try to find a homomorphism from the group of points on an elliptic curve to the jacobian of another curve such that:

- The solution of the DLP remains the same under the homomorphism. This means that we can use the solution of the DLP in the jacobian of the new curve to solve the DLP over the elliptic curve.
- The DLP in the new group, is easier to solve (usually provided by an index calculus attack).

The GHS attack, first proposed in [GHS02b], exploits the relationship between function fields of curves to find a map to transform the DLP from the elliptic curve group to the jacobian of a new curve. In this section, we first study the general process of the GHS attack for finding the DLP transformer map using the relation between the function fields. After that we focus on each function field involved in the attack and we study its properties. Finally we describe the homomorphisms which transform the DLP to an easier instance.

It is worthy of mention that the GHS attack can be applied to solve the DLP on the jacobian of hyperelliptic curves [Die03], [Hes04]. In this work, we restrict ourselves to the case of elliptic curves. In this case, the group of the points of the elliptic curve E and the jacobian of the elliptic curve $Cl^0(E)$ are isomorphic (See for example Proposition III.3.4 [Sil94]).We will always use the notation $Cl^0(E)$ for the group of the points of E.

3.1 The GHS attack in nutshell

The ultimate goal of the GHS attack is to find a homomorphism between the jacobians of two curves. This map should preserve the DLP. Suppose we want to attack curve E using curve C. The original DLP is to find n such that:

$$[D_2] = n[D_1] \text{ where } [D_2], [D_1] \in Cl^0(E) \simeq E$$

We want to find a homomorphism ϕ such that:

$$\phi : Cl^0(E) \rightarrow Cl^0(C)$$

The general process of finding such a homomorphism ϕ, known as the GHS attack ([GHS02b] and [Hes04]), is to find it by extending the function field of the curve under

attack. Let $K(E)$ be the function field of the curve under attack. In the GHS attack the field $K(E)$ is extended to F. The resulting field is a function field of another curve C (according to Proposition 3.3.5), so we have $F = K(C)$. The function field extension $K(C)$ needs to satisfy a series of criteria in order to transfer the DLP successfully. These criteria are discussed in Section 3.2. The GHS attack consists of an algorithm to extend $K(E)$ and construct $K(C)$. The extension process involves choosing a series of irreducible polynomials and adding their roots to $K(E)$. The process of choosing such polynomials and constructing $K(C)$ are discussed in Section 3.4.

After finding $K(C)$, we need a map to transfer the DLP to C. Since $K(E) \subset K(C)$, the injection between these function fields induces a map between the ideal groups of their coordinate rings and consequently their jacobians. In this situation, any ideal in the coordinate ring of E can be regarded as an ideal in the coordinate ring of C, parallel to that, any divisor class in $Cl^0(K(E))$ can be regarded as a divisor class in $Cl^0(K(C))$. In this way we can map the DLP to $Cl^0(K(C))$.This map is called the "Conorm map" in the GHS literature (for example in [GHS02b] and also in [Che51]). We will study the map in more detail in Section 3.6.

The fact that the genus of $K(C)$ is larger than the genus of $K(E)$ makes the index calculus algorithm available to solve the DLP over $Cl^0(K(C))$. However, as $K(C)$ and $K(E)$ has the same constant fields, while the function field $K(C)$ is larger than $K(E)$, solving the DLP over $Cl^0(K(C))$, is harder than the original DLP, despite using index calculus.

The whole scenario would be then useless, unless we can find a smaller constant field $k \subset K$ on which C is defined. This is achieved by using a special automorphism σ over $K(C)$. We set $F' := K(C)^{<\sigma>}$ in which by $K(C)^{<\sigma>}$ we mean the fixed field of σ. In Section 3.3.4 we see that indeed F' has a smaller constant field. If $k \subsetneq K$ is the constant field of F' and C_0 is the curve for which $F' = k(C_0)$, we can solve the DLP in

$Cl^0(k(C_0))$.

At this stage, we reach a smaller function field with a curve with higher genus. The last step is to find another map $\psi : Cl^0(K(C)) \rightarrow Cl^0(k(C_0))$ to transfer the DLP from $Cl^0(K(C))$ to $Cl^0(k(C_0))$. For this part of the process we use the norm function from the large extentended field $K(C)$ to the smaller subfield $k(C_0)$ fixed by σ. The norm function induces a homomorphism between jacobian of C and jacobian of C_0. We will discuss this function in Sections 3.6.3.

In following sections, we study each step of the GHS attack in more detail. The most important step in the GHS attack is to find a suitable function field extension $K(C)$. In next section we describe the criteria of function field $K(C)$. In Section 3.3.1, we define the algebraic objects that we encounter in the GHS attack.

In Section 3.3, we assume that we have the whole structure needed for the attack, and we study the properties of this structure. This would help us to prove that the GHS attack can successfully transform the DLP into an easier problem.

In Section 3.4, we describe how to generally construct the extension $K(C)$ which is needed for the attack.

In Section 3.5, We discuss the genus of extended function fields $K(C)$ and $k(C_0)$.

In Section 3.6, We describe the homomorphism which transform the DLP from $Cl^0(K(E))$ to $Cl^0(K(C))$ and from $Cl^0(K(C))$ to $Cl^0(k(C_0))$.

3.2 Criteria in choosing the extended function field $K(C)$

As we mentioned in previous section, the GHS attack consists of extending the function field of the elliptic curve to the function field of another curve C and then finding a subfield of $K(C)$ on which the DLP is easier to solve.

There are three main concerns in choosing $K(C)$:

- **Faithfulness of the DLP mapping** As we discuss in Section 3.6, we always need to check that the homomorphism we use to transfer the DLP, does not have a large kernel. So, we should not use an extension of $K(E)$ in which the induced map from $Cl^0(K(E)) \rightarrow Cl^0(K(C))$ has a large kernel. However, it is shown in Section 3.6.2 that this map seldom has a large kernel in a practical cryptographic case. Therefore, this concern does not really put a limitation on choosing the extension $K(C)$.

- **Existence of a suitable field automorphism** After constructing $K(C)$, we use an automorphism σ such that $k(C_0) = K(C)^{<\sigma>}$, for some curve C_0 defined over k.

  In order for σ to be effective in lowering the constant field of $K(C)$, σ should not fix K, otherwise, $K \subseteq K(C)^{<\sigma>}$, therefore $K(K(C)^{<\sigma>}) = K(C)^{<\sigma>}$ which means that K is the constant field $K(C)^{<\sigma>}$. This means there $k = K^{\sigma}$ is a subfield of K. The GHS attack is then ineffective against curves defined over the prime field.

  Choosing σ also defines the norm function of $K(C)/k(C_0)$. The norm function induces the homomorphism between the jacobians of the fields. We choose σ to ensure that we get a homomorphism with desirable small kernel.

- **The lowest genus possible** The jacobian of curve C_0 is the group on which we aim to run the index calculus attack. As we will see in Section 3.5 the genus of C_0 is equal to the genus of C. As all index calculus algorithms known for solving the DLP over jacobians, are at best subexponential in the genus of the underlying curve, we want to find the extension with the smallest genus possible to increase the efficiency of the attack. On the other hand, we do not want that the genus of C be very small (1, 2 or 3) as there is no efficient index calculus algorithm for

the DLP over these curves.

3.3 The general field theoretic structure of the attack

In this section we discuss the properties of all fields involved in the GHS attack and their relationship to each other, i.e. the extensions in detail. We establish the notations and the definitions related to the GHS attack in Section 3.3.1. The new notation we develop in this section remains valid in the remainder of the thesis.

The discussion in this section is more theoretical than the discussion in Section 3.4, which shows how algorithmically we can build the $K(C)$.

3.3.1 Notations and assumptions on structure of the GHS attack

Assumption 3.3.1 Algebraic Structure of the GHS attack

- *E: the curve that we want to attack. Form now on we suppose that E is an elliptic curve.*
- *K: a finite field of characteristics p, K is the constant field of $K(E)$ and the field of definition of E.*
- *$F = K(C)$: is a regular finite extension of function field $K(E)$ of degree $d_{C/E} := [K(C) : K(E)]$. We fix a defining curve of this function field and we call it C. Also we require that $K(C)$ has automorphism σ as defined below.*
- *σ is an automorphism of $K(C)$ that does not fix K. The existence of such σ is necessary for the GHS attack.*
- $n := ord(\sigma)$.
- *$k \subset K$ is a subfield of K which is the constant field of $K(C)^{<\sigma>}$.*

- $F' = k(C_0) = K(C)^{<\sigma>}$ *is the subfield of function field* $K(C)$ *that is fixed by* σ.

3.3.2 Fields K, k and extension K/k

Let k be the subfield of K fixed by σ (Assumption 3.3.1). By the fundamental theorem of Galois theory [Mor96], $[K : k] = n$. By theory of finite fields [Mor96], we know that k has p^r elements for some integer r. Consequently $|K| = (p^r)^n$. By the Corollary II.6.7 of [Mor96], $\mathrm{Gal}(K/k)$ is cyclic of order n generated by the Frobenius automorphism of K/k. Therefore, by possibly replacing σ by one of its powers, we can assume that $\sigma|_K$ is the Frobenius automorphism of K over k. From now on, we denote the Frobenius automorphism of K/k with σ as well, whenever it does not cause ambiguity. By the above discussion we know that:

$$\forall a \in K, \sigma(a) = a^{p^r}.$$

The extension K/k is depicted in Diagram 3.1.

$$\begin{array}{l} K := \mathbb{F}_{q^n} \\ \quad\quad n \Big| \, Gal(K/k)=<\sigma> \\ k := \mathbb{F}_{q:=p^r} \end{array}$$

Diagram 3.1: Extension K/k

3.3.3 Extensions $K(z)/k(z)$, $K(E)/K(z)$ and $K(E)/k(z)$

The GHS attack proposes a special way to choose the extended function field $K(C)$ as well as constructing the σ automorphism. Although it is not the only way that one can achieve a suitable $K(C)$ such that it satisfies the criteria of Section 3.2, it is a concrete way to do so.

The first step in building $K(C)$ is to choose a rational subfield of $K(C)$ [BSS05]. This means to choose any function $z \in K(E)$ and write the new equation of the curve in term of this function. This results in a new defining equation of the curve $g(z, y)$. Choosing this element determines $K(C)$, the whole algebraic structure and consequently the complexity of the attack. Therefore choosing the function z is considered as the most important step in implementing the attack on a special curve.

In [GHS02b] and [Hes04], z is chosen such that $K(E)/K(z)$ is an Artin-Schreier extension (of degree 2). We discuss this case in more detail in Section 4.5. In [Die03] and [The03], z is chosen such that $K(E)/K(z)$ is a Kummer extension. As we want to study the general quadratic extensions in this thesis, we assume $[K(E) : K(z)] = 2$.

After choosing z as above, we are able to define $k(z)$ as well as $K(z)$ as follows.

Definition 3.3.2 Function z in the GHS attack *Element $z \in K(E)$ is a transcendental element over K such that $K(E)$ is a quadratic elliptic extension of $K(z)$ where $K(z)$ is the rational function field in z over K.*

Let $k(z)$ be the rational function field in z over k. It is easy to prove that $K(z)/k(z)$ is a finite Galois extension of degree n with cyclic Galois group generated by an extension of the Frobenius automorphism of K/k such that it fixes z. Also we also have $K(E) = K(z)[y]/g(z, y)$, which means $K(E)/K(z)$ is an algebraic extension with degree $d := [K(E) : K(z)] = \deg_{K(z)[y]}(g(z, y))$. As we limit ourselves to quadratic extensions, we know that $d = 2$. Therefore $K(E)/k(z)$ is an algebraic extension and we have $[K[E] : k(z)] = 2n$.

For the GHS attack, we also need to put an extra condition to assure that our attack would be successful.

Assumption 3.3.3 The GHS Strong Assumption *Let $n = [K : k] = ord(\sigma)$, and*

$2 := [K(E) : K(z)]$. *We assume that:*

$$(2, n) = 1$$

Remark. We close this section by emphasizing that the size of the orbit of y when σ acts on Galois closure of $K(E)/k(z)$ is of most importance in the GHS attack as you will be seen in Section 3.3.3.1.

3.3.3.1 Function field $K(C)$ and automorphism σ

In the GHS attack we want to define the function field $K(C)$ as an extension of $K(E)$. However, the definition of a function field in 2.1.3 is based on a defined curve. In the case of $K(C)$ we do not build $K(C)$ using an existing curve C. In contrast, we define it as an extension of $K(E)$. Therefore we need a more general definition of a function field, which does not depend on the defining curve.

Definition 3.3.4 (Definition I.1.1. [Sti93]) Function field of one variable *An algebraic function field F/K of one variable over K is an extension field $F \supset K$, such that F is a finite algebraic extension of $K(x)$ for some element $x \in F$ which is transcendental over K.*

As in this thesis, we just deal with algebraic function fields of one variable, here after we simply use the term "function fields" to refer to them. Although these two definition of function fields seems different, in the case that K is a prefect field it is easy to prove that they are equivalent.

Proposition 3.3.5 *Let F be a function field over K in the sense of Definition 3.3.4, there is a curve C such that $F = K(C)$.*

Proof. Given function field F, one can construct C as follows. As $F/K(x)$ is a finite extension and K is prefect, Corollary I.5.7 [Mor96], assures that F is a simple extension

of $K(x)$, i.e. there exists $y \in F$ such that such that $F = K(x, y)$. Let $p(X, Y) \in K[X, Y]$ be the irreducible polynomial of y over $K(x)$. Let $K(C)$ be the function field of the curve C defined by $p(X, Y)$. Now writing each element $f \in F$ as $f = \sum_{i=0}^{d} f_i(x)y^i \in K(C)$ define an isomorphism between F and $K(C)$, and therefore $K(C) \cong F$.

□

Proposition 3.3.5 allows us to talk about an algebraic extension of $K(E)$ and call it $K(C)$ without defining curve C a priori. In the GHS attack we define $K(C)$ as follows.

Definition 3.3.6 *$K(C)$ is defined to be the splitting field of $K(E)/k(z)$.*

We can now define σ easily. As $K(C)$ is the splitting field of some extension, by definition $K(C)/k(z)$ is normal. On the other hand $K(z)/k(z)$ and $K(E)/K(z)$ are separable and by Proposition I.4.21 [Mor96] $K(E)/k(z)$ is separable. By definition of separability, this means $K(C)/k(z)$, the splitting field of $K(E)/k(z)$ is separable. Therefore $K(C)/k(z)$ is Galois. So we can look at $\mathrm{Gal}(K(C)/k(z))$. We need to choose σ such that it satisfies following condition:

- $\mathrm{ord}(\sigma) = n$
- $\sigma|_{K(z)} =$ Frobenius automorphism of $K(z)/k(z)$.

We need to make sure that we are always able to find such σ. Consider following exact sequence:

$$1 \to \mathrm{Gal}(K(C)/K(z)) \to \mathrm{Gal}(K(C)/k(z)) \to \mathrm{Gal}(K(z)/k(z)) \to 1 \tag{3.1}$$

Let us examine the case when this sequence splits. According to Theorem VI.1.18 [Hun03], we know that the above sequence splits if and only if

$$\mathrm{Gal}(K(C)/k(z)) \simeq \mathrm{Gal}(K(C)/K(z)) \times \mathrm{Gal}(K(z)/k(z))$$

In that case we can simply choose $\sigma \in \mathrm{Gal}(K(C)/k(z) = 1_{K(C)/K(z)} \times \sigma_{K(z)/k(z)}$ which has order n. Therefore the fact that this sequence splits is sufficient for us to carry out the attack. For that it is sufficient that the elements of $\mathrm{Gal}(K(C)/K(z))$ and $\mathrm{Gal}(K(z)/k(z))$ have different orders.

$$
\begin{array}{c}
K(C) \\
\Big| \; d'=|\mathrm{Gal}(K(C)/K(E))| \\
K(E) \\
\Big| \; 2 \\
K(z) \\
\Big| \; n \\
k(z)
\end{array}
$$

Diagram 3.2: The series of function field extensions to reach $K(C)$

According to Diagram 3.2, the sequence splits when $(2d', n) = 1$. By Assumption 3.3.3 we knew that $(2, n) = 1$. Proposition 3.4.2 assures that using the GHS construction the above sequence splits.

3.3.4 Extension $K(C)/k(C_0)$

Since curve C has higher genus than E, the index calculus attack is applicable to $Cl^0(K(C))$. However, no matter what the genus of C is, if we run the index calculus attack on $Cl^0(K(C))$, we always expect worse running time than running generic attacks on $Cl^0(K(E))$ such as Pollard's ρ attack. This is because $K(C)$ and $K(E)$ both have the same constant field K. Although the index calculus attack is subexponential in genus of the curve under attack, it is fully exponential in the size of its constant field, i.e. $O(q)$ if q is the size of the constant field. However, Pollard's ρ method takes only $O(\sqrt{q}) < O(q)$. This means that if we increase the genus while keeping the same constant field, we end up with a worse time for the attack. The whole point of the GHS

attack is to decrease the constant field size at the cost of not extending the genus too much. For this reason, we still need to transform the DLP to another curve whose genus is higher than E and at the same time its constant field is smaller than the constant field of E.

To find this final function field to attack, we need to use the automorphism σ we found in last section. Having σ, we simply set $F' := K(C)^{<\sigma>}$. However to assure that index calculus is efficient on F' we need to prove:

Proposition 3.3.7 *$K(C)$ is a constant field extension of F'.*

Proof. Suppose $f \in F'$ is a rational function. Since $f^\sigma = f$ means that f is defined over k, and therefore F' is defined over $k(z)$. This means that there exists a curve C_0 defined over k, such that $F' = k(C_0)$. Now we want to look at the function field $Kk(C_0)$. We know that the extending polynomial of K/k is irreducible over $k(C_0)$ as its roots are not fixed by σ. However, these roots are in $K \subset Kk(C_0)$. Therefore $[Kk(C_0) : k(C_0)] \geq n$. Now considering the definition of $k(C_0) = K(C)^{<\sigma>}$ and the fact that $\mathrm{ord}(\sigma) = n$, we know $[K(C) : k(C_0)] = n$, so $K(C) = Kk(C_0) = K(C_0)$. Therefore $K(C)$ is a constant field extension of $k(C_0)$.

□

As the constant field of F' is k, we can find a curve C_0 defined over k such that $F' = k(C_0)$. From now on we denote this function field as $k(C_0)$. Using Proposition 3.3.7, we know that $K(C) = Kk(C_0) = K(C_0)$.

3.4 Algorithmic construction of $K(C)$ and the automorphism σ

The Frobenius automorphism of K/k is the heart of the GHS attack. This is the fundamental reason that there is no obvious way to generalize the GHS attack to prime fields as in that situation there is no Frobenius automorphism. We use this automorphism to find $k(C_0)$, a subfield of $K(C)$.

To emphasize the importance of the extended Frobenius automorphism, we remark that when this automorphism is determined (or equivalently z is chosen), the whole structure of $k(C_0)$ can be determined, including the genus of C which determines the difficulty (including the success/failure) of the attack.

In Chapter 4, we discuss the problem of choosing a suitable z for the even characterstic case. However here, we assume that we have chosen z as it is discussed in Section 3.3.3, and therefore we can compute the Frobenius automorphism $\sigma_{K(z)/k(z)}$. Therefore the goal of this section is to describe the algorithm which constructs the extension $K(C)$ given z.

To understand the process of constructing $K(C)$ in [GHS02b] and [Hes04], one can look at the construction of $K(E)$. Since $y_0 := y$ is algebraic over $K(z)$ of degree 2, we have the quadratic equation $f(y_0) = 0$ such that $f(Y) \in K(z)[Y]$. $K(E)$ is simply $K(E) := K(z)[Y]/f(Y)$. Now to extend $K(E)$ we need another polynomial.

Definition 3.4.1 *Let* $f(Y) = \sum_{i=0}^{\ell} a_i Y^i \in K[Y]$ *be a polynomial and* σ *be an automorphism of* K. *We denote the action of* σ *on* $f(Y)$ *by*

$$f^{\sigma}(Y) = \sum_{i=0}^{\ell} \sigma(a_i) Y^i.$$

Let $f(Y) \in K[Y]$ be a quadratic polynomial with root $y_0 \in K(E)$ as defined above,

and let $K_0 := K(E)$. Let $f_1(Y) := f^{\sigma}(Y)$. If $f_1(Y)$ is irreducible over K_0, then we add its root, y_1, to K_0 and we construct $K_1 := K_0(y_1)$, otherwise $K_1 := K_0$. Continuing this procedure we can extend $K(E)$ to $K(C)$ with the desired properties. As σ has order n, the procedure stops after at most n iterations. Algorithm 3.1 is the complete procedure of constructing $K(C)$.

Algorithm 3.1 CONSTRUCTING $K(C)$

Require: $f(Y) \in K(z)[Y]$ be the defining equation of elliptic curve E such that $f(y_0) = 0$, σ be the Frobenius automorphism of K/k.
Ensure: $K_n = K(C)$

1: $K_0 \leftarrow K(E)$
2: **for all** i such that $1 \leq i \leq n$ **do**
3: $f_i(Y) \leftarrow f^{\sigma^i}(Y) \in K(z)[Y]$
4: **if** $f_i(Y)$ is irreducible over K_{i-1} **then**
5: $K_i \leftarrow K_{i-1}(y_i)$ such that $f_i(y_i) = 0$.
6: **else**
7: $K_i \leftarrow K_{i-1}$
8: **end if**
9: **end for**
10: **return** K_n

After generating $K(C)$ using Algorithm 3.1 or equally using the theoretical procedure discussed in Section 3.3.3.1, we need an element $\sigma \in \mathrm{Gal}(K(C)/k(z))$ such that σ has order n over $K(C)$. This is because we want that $[K(C) : K(C)^{<\sigma>}] = n$. According to Proposition 3.4.2 such an element exists.

Proposition 3.4.2 *If n is odd and $K(C)$ is constructed by Algorithm 3.1, there exists σ which satisfies the conditions stated in Section 3.3.3.1.*

Proof. Using field theory we know:

$$[K(C) : K(E)] = \prod_{i=1}^{n-1} [K_i : K_{i-1}].$$

As $\deg(f^{\sigma^i}) = 2$ for all i, $[K_i : K_{i-1}]$ is either 2 or 1. Therefore $d' = [K(C) : K(E)] = 2^m$

for some integer m. By Assumption 3.3.3 we know that $(2, n) = 1$, so $(d', n) = (2^m, n) = 1$. This means that the sequence of (3.1) splits, which is sufficient for existence of σ.

□

3.4.1 The magic number "m"

In Step 2 of Algorithm 3.1, each time we face the question: is $f^{\sigma^i}(Y)$ irreducible over K_{i-1} or not? The number of times that we answer positively to this question, determines the difficulty of the GHS attack to E using the chosen subfield $K(z)$. For that reason, we refer to it as the magic number associated to extension $K(E)/K(z)$ and denote it by m as it is traditional in the GHS literature. Algorithm 3.2 computes the value of m along constructing $K(C)$.

Algorithm 3.2 CONSTRUCTING $K(C)$ AND COMPUTING m

Require: $f(Y) \in K(z)[Y]$ be the defining equation of elliptic curve E such that $f(y_0) = 0$
Ensure: $K_m = K(C)$

1: $K_0 \leftarrow K(E)$
2: $m \leftarrow 0$
3: **for all** i such that $1 \leq i \leq n$ **do**
4: $f_i(Y) \leftarrow (f)^{\sigma^i}(Y) \in K(z)[Y]$
5: **if** $f_i(Y)$ is irreducible over K_{i-1} **then**
6: $m \leftarrow m + 1$
7: $K_m \leftarrow K_{m-1}(y_m)$ such that $f_i(y_m) = 0$.
8: **end if**
9: **end for**
10: **return** K_m

Diagram 3.3 shows the procedure of generating $K(C)$ using Algorithm 3.2. Finally we close this subsection by proving that Algorithm 3.2 construct $K(C)$ according to its theoretical definition.

Proposition 3.4.3 *The field generated by adding $y_i, i = 0, \ldots, m$ to $K(z)$ resulted from Algorithm 3.2 and the field $K(C)$ defined in Definition 3.3.6 are isomorphic.*

$$
\begin{array}{c}
K_n = K(E)(y_0, y_1, y_2, ..., y_m) \\
\vdots \\
K_2 = K(z)(y_0, y_1, y_2) \\
| \\
K_1 = K(z)(y_0, y_1) \\
| \\
K_0 = K(E) = K(z)(y_0) \\
| \\
K(z)
\end{array}
$$

Diagram 3.3: The series of function field extensions generated by algorithm 3.2 to reach $K(C)$

Proof. First we want to show that $K(C)$ contains the splitting field of $m_{k(z)}(y)$, i.e. all of its roots. As P_E is not defined over any subfield of $K(z)$, we know that $P_E^{\sigma^i}(y)$ are different for $0 \leq i \leq n-1$. We define

$$M(Y) := \prod_{i=0}^{n-1} P_E^{\sigma^i}(Y)$$

As this polynomial is invariant under action of σ, it is defined over $k(z)$. On the other hand $m_{k(z)}(Y) \mid M(Y)$ because y is a root of $M(Y)$. Using the GHS algorithm we know that $M(Y)$ has all of its roots in $K(C)$, because the algorithm makes sure that all $P_E^{\sigma^i}(Y)$ (which are quadratic) are reducible before stopping. Now $M(Y)$ is a polynomial of degree $2n$ because it consists of n quadratic factors, therefore $M(Y) | m_{k(z)}(Y)$. This means that $M(Y) = m_{k(z)}(Y)$ and therefore $K(C)$ is the splitting field of extension $K(E)/k(z)$.

□

As the result of Proposition 3.4.3 we name both of these fields as $K(C)$. Diagram 3.4 shows the relation of all function fields involved in the GHS attack:

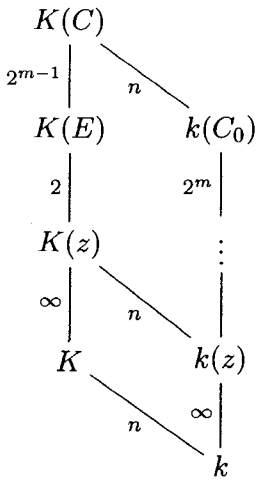

Diagram 3.4: The relation of function field extensions in the GHS attack

3.5 The Genus of resulting curves

The genus plays an important role in the complexity of attack. The final complexity of attack depend on two major facts,

- The size of $Cl^0(k(C_0))$.
- The effectiveness of index calculus attack over $Cl^0(k(C_0))$.

Both of these parameters depends on the genus of the curve C_0. As it is mentioned in Theorem 2.2.1, the size of $Cl^0(k(C_0))$ is of $O(q^g)$ where g is the genus of C_0. Therefore is exponentially related to the genus of C_0. Also the type of index calculus attack to be chosen depend on the genus of C_0.

As we generate $K(C)$ explicitly and defined $k(C_0)$ as the fixed field of $K(C)$ by σ it is easier from a computational point of view to investigate the genus of $K(C)$. The following proposition from [Ros02] shows that we merely need to study the genus of $K(C)$ as it is equal to the genus of $k(C_0)$:

Proposition 3.5.1 *Let k be a prefect field and let K/k be a finite extension. Suppose*

that $k(C_0)$ is a function field whose constant field is k. Let $K(C) = Kk(C_0)$ be a constant field extension of $k(C_0)$. Then the genus of $K(C)$ considered as function field over K is equal to the genus of $k(C_0)$.

Therefore we only need to compute the genus of the curve C which we obtained through Algorithm 3.2.

The genus of the extended function field, depends on the method of extending the function field of elliptic curve E. The genus can be computed in special cases when we extend $K(E)$ using specific polynomials. Two specific methods are discussed for even and odd characteristics in [GHS02b][Die03]. However, in more general cases as dealt in [Hes04], we can merely find upper and lower bounds on the genus of C.

Finding the genus of general extension in the GHS attack is still an open problem. In next chapter we study the situation when an Artin-Schreier extension is used to find the descend. In that situation the genus is computed in [GHS02b], [Hes04].

In next chapter we find an upper bound for the case $p = 2$.

3.6 Mapping the discrete logarithm problem

The focus of this work is on studying the suitable function field extensions for the attack. Nevertheless, the final goal is to solve the DLP which is responsible for the security of the cryptographic system. All the tedious search to find a suitable function fields would be useless, if we can not find a suitable map to transfer the DLP to a group where it is easier to be solved. In this regard, we devote this section to go through the mapping process.

The function field structure is built in order to help us to find the suitable map which can transfer the discrete logarithm over the Elliptic Curve E (or equivalently over the divisor class group, $Cl^0(K(E))$) to the discrete logarithm problem over $Cl^0(k(C_0))$. We

use the extension/subfield relations between $K(E), K(C)$ and $k(C_0)$ to construct that map. This process consists of two steps:

- Mapping a divisor in $Cl^0(K(E))$ (or equivalently a point on E) to a divisor over $Cl^0(K(C))$
- Mapping a divisor in $Cl^0(K(C))$ to a divisor over $Cl^0(k(C_0))$.

Both of these maps are derived using the relationship between function fields we built in this chapter. In following two sections we deal with these maps separately.

3.6.1 Mapping the DLP faithfully

Suppose we want to attack curve E using curve C. The original DLP is to find n such that $[D_2] = n[D_1] : [D_2], [D_1] \in Cl^0(K(E)) \simeq E$. Suppose we have found a homomorphism ϕ such that

$$\phi : Cl^0(K(E)) \rightarrow Cl^0(K(C)).$$

Now we have

$$\begin{aligned} [D_2] - n[D_1] &= 0 \\ \Rightarrow \phi([D_2] - n[D_1]) &= 0 \\ \Rightarrow \phi([D_2]) - n\phi([D_1]) &= 0 \end{aligned}$$

In presence of such homomorphism, suppose that we have a feasible algorithm to solve the DLP $\phi([D_2]) = m\phi([D_1])$ over the $Cl^0(K(C))$. By running that algorithm over

$Cl^0(K(C))$ we obtain m such that:

$$\begin{aligned} \phi([D_2]) - m\phi([D_1]) &= 0_{Cl^0(K(C))} \\ \Rightarrow \phi([D_2] - m[D_1]) &= 0_{Cl^0(K(C))} \\ \Rightarrow [D_2] - m[D_1] &\in Ker(\phi) \end{aligned}$$

As ϕ is a homomorphism, its kernel is a group. If we have $g = |Ker(\phi)|$, the order of this group, by lagrange theorem in group theory [Mor96] then we have

$$g([D_2] - m[D_1]) = g[D_2] - gm[D_1] = 0_{Cl^0(K(E))}.$$

On the other hand the DLP over $Cl^0(K(E))$ says that

$$[D_2] - n[D_1] = 0_{Cl^0(K(E))} \Rightarrow g([D_2] - n[D_1]) = g[D_2] - gn[D_1] = 0_{Cl^0(K(E))}$$

Subtracting these two equations we get

$$gn[D_1] - gm[D_1] = g(n-m)[D_1] = 0_{Cl^0(K(E))} \Rightarrow \mathrm{ord}_{Cl^0(K(E))}([D_1]) \mid g(n-m) \quad (3.2)$$

However, if $\mathrm{ord}([D_1])$ as the generator of the DLP group is not a prime, the Pohlig-Hellman attack is applicable [MVO96] to the DLP on E, which break down the discrete logarithm in the Sylow subgroups of $< [D_1] >$ efficiently then deduce the DLP on E. As solving the DLP over each of those Sylow subgroups, actually consists of solving the DLPs over subgroups of prime order, the complexity of the problem reduces to the difficulty of solving the DLP over a group whose number of elements is equal to the largest prime which divide $| < [D_1] > |$. Therefore, it does not make sense to choose $\mathrm{ord}([D_1])$ to be composite. In the other words if we choose $\mathrm{ord}([D_1])$ to be composite number we

increase the complexity of encryption/decryption process, however the security of the system remains as low as the time that we choose the group order as the largest prime dividing $\text{ord}([D_1])$. In this regard, we can safely assume that $p = \text{ord}([D_1])$ is a large prime number.

Now if p is a prime number, equation 3.2 results in $p \mid g$ or $p \mid n - m$. in the second case it means $m \equiv n(\mod p)$ and it means that we have solved the original DLP. On the other hand, if $p \mid g$. This means that ϕ has a large kernel, because p should be large, (of order $O(2^{160})$), to ensure the security of the system. So in search of such homomorphism to attack the $Cl^0(K(E))$ we should keep away from homomorphisms with large kernel.

In a nutshell, if we can find a map between the jacobians of two curves, under the condition that the map does not have large kernel, solving the DLP over the jacobian of the new curve, solves the original DLP as well. In this regard the goal of the attack is to find an algorithm to compute such a map.

3.6.2 Mapping $Cl^0(K(E))$ to $Cl^0(K(C))$

Using the theory of function fields we have following theorem:

Proposition 3.6.1 *Let E/K and C/K be curves as defined in this chapter, such that $K(E) \subset K(C)$ and $\lambda : K(E) \hookrightarrow K(C)$ be the embedding map. Then there exists a morphism of curves $\phi : C \to E$ which induce the group homomorphism $\phi^* : Cl^0(K(E)) \to Cl^0(K(C))$.*

Proof. First we need to build a map between curves using the embedding of the function fields. According to Theorem II.2.4 of [Sil94], let $\lambda(x)$ and $\lambda(y)$ be images of $x, y \in K(E)$ in $K(C)$ respectively, in this regard we define $\phi := (\lambda(x), \lambda(y))$. We claim that ϕ is a morphism of curves. Suppose that $f(X, Y) \in K[X, Y]$ is the defining equation of curve E, therefore we have $f(x, y) \equiv 0 \in K(E) \Rightarrow \lambda(f(x, y)) = f(\lambda(x), \lambda(y)) \equiv 0 \in K(C)$.

Now $\forall(x,y) \in C \Rightarrow \phi(x,y) = (\lambda(x), \lambda(y)) \Rightarrow f(\lambda(x), \lambda(y)) = 0 \Rightarrow (\lambda(x), \lambda(y)) \in E$.

According to [Sil94][P.33], the morphism ϕ, induces a group morphism ϕ^* such that:

$$\phi^* : Div(E) \rightarrow Div(C)$$

$$\phi(Q) \rightarrow \sigma_{P \in \phi^{-1}(Q)} e_\phi(P)(P)$$

Proposition II.3.6 [Sil94] says that $deg(\phi^*(D)) = deg(\phi)deg(D)$. Therefore ϕ^* maps degree zero divisors to degree zero divisors. So it is induce a map on $Cl^0(K(E)) \rightarrow Cl^0(K(C))$.

□

According to Proposition 3.6.1 We have following commutative diagram

$$\begin{array}{ccccccccc} 0 & \longrightarrow & K(E) & \xrightarrow{div} & Div_0(K(E)) & \longmapsto & Cl^0(K(E)) & \longrightarrow & 0 \\ & & \downarrow{\scriptstyle \phi^*} & & \downarrow{\scriptstyle \phi^*} & & \downarrow{\scriptstyle \phi^*} & & \\ 0 & \longrightarrow & K(C) & \xrightarrow{div} & Div_0(K(C)) & \longmapsto & Cl^0(K(C)) & \longrightarrow & 0 \end{array}$$

Diagram 3.5: The map ϕ^*

Therefore ϕ^* can serve us as a map to transform the discrete logarithm problem from E to C.

The chosen homomorphism to map the DLP in class groups is called "ϕ^*" in [Sil94], "π^*" in [MWZ96] and "Conorm" in [Che51] and [MTW04]. Moreover the literature on the GHS attack uniformly call it "Conorm" function as well.

3.6.2.1 Preservation of the DLP under conorm map

We found a homomorphism to map the class groups, however we should make sure that the solution of the mapped DLP over $Cl^0(K(C))$ is a solution of the original DLP over

$Cl^0(K(E))$. Suppose we try to solve the DLP for $[P], [Q] \in Cl^0(K(E))$, in which $[P]$ and $[Q]$ are the divisor classes corresponding to points P and Q respectively over curve E. The correspondence provided by the isomorphism between points on the elliptic curves and its jacobian. The DLP means we want to find s such that $[Q] = s[P]$ in $Cl^0(K(E))$. Instead we want to solve $\phi^*([P]), \phi^*([Q])$.

Suppose the DLP under attack, is defined in a subgroup $H < Cl^0(K(E))$ with prime order s.

Therefore we transfer the problem to $\phi^*(H)$ and we want to solve the problem. As ϕ^* is a homomorphism, we know that $ker(\phi^*(H)) < H$ therefore $|ker(\phi^*(H))| \mid |H|$. Now because H has prime order we can deduce that either $ker(\phi^*(H)) = \emptyset$ or $ker(\phi^*(H)) = H$.

In the first case we have $ker(\phi^*(H)) = \emptyset$, it means that ϕ^* is an isomorphism and the DLP is transferred faithfully. So, the solution of the DLP over $Cl^0(K(C))$ is the same as the solution for $Cl^0(K(E))$ and the subgroup $\phi^*(H) \cong H$ and particularly has the same order. In this case the DLP is preserved completely.

In the second case, however, the whole subgroup is mapped to the identity element of $Cl^0(K(C))$. In this case we lose the DLP completely. So the ϕ^* drags the whole subgroup to the kernel and therefore the DLP is not transformable to $Cl^0(K(C))$ using ϕ^*.

Now we want to study under what condition we face the second case and consequently a failure in our attack. To investigate the situation, we apply function, ϕ_* to the image of ϕ^*. So in the undesirable case that $\phi^*([P]) = 0 \Rightarrow \phi_*(\phi^*([P])) = 0$. Now according to following proposition (proposition II.3.6 [Sil94]), $\phi_* \circ \phi^*$ can be simplified:

Proposition 3.6.2 *Let E and C be two curves and $\phi : C \rightarrow E$ be a non-constant map*

from C to E. Then for any divisor class$[D] \in Cl^0(K(E)$, we have:

$$\phi_* \circ \phi^*([D]) = [\deg(\phi)][D]$$

Therefore we have $[deg(\phi)][P] = 0$, As $[P]$ was not originally 0 divisor class (the identity element of the group), we see that P should be a $[deg(\phi)]$-torsion point. By definition $deg(\phi) = [K(C) : \phi^*(K(E))]$. In addition we know that ϕ^* is simply embedding of $K(E)$ in $K(C)$ so it means $deg(\phi) = [K(C) : \phi^*(K(E))] = [K(C) : K(E)]$. In Section 3.3.3.1 we chose $K(C)$ so it would be the splitting field of $K(E)/k(z)$. We used m different copies of $K(E)/K(z)$ to construct $K(C)$. So we have $deg(\phi) = [K(C) : K(E)] \mid [K(E) : k(z)]^{m-1}$. So if the divisor class corresponding to point P get killed by ϕ^* we should have:

$$[\deg(\phi)][P] = 0 \Rightarrow \text{ord}([P]) = s \mid \deg(\phi) \mid [K(E) : k(z)]^{m-1} \Rightarrow$$
$$s \mid [K(E) : k(z)]$$

However, in the case of cryptographic interest s is a very large prime (of $O(2^{160})$) and to have a practical attack we want that the order of $[K(E) : k(z)]$ be very small (~ 2). Therefore, it is impossible in practical situation that ϕ does not transform the DLP to $Cl^0(K(C))$ faithfully.

3.6.3 Mapping $Cl^0(K(C))$ to $Cl^0(k(C_0))$

In this side of tower, as $k(C_0) \subset K(C)$, one may find it natural to use the Proposition 3.6.1 in the same way used in Section 3.6.2 and then use ϕ_* to map the DLP. However, as the underlying function fields are defined over different constant field this time, the proposition is not applicable. Instead, we use simple Norm map of the field extention.

The norm map of extension $K(C)/k(C_0)$, $N_{K(C)/k(C_0)}$, induces a norm map from C/K to C_0/k in following way

$$\forall P \in C/K, N_{K(C)/k(C_0)}(P) = \sum_{i=1}^{n} \sigma^i(P). \tag{3.3}$$

Then one can naturally (additively) extend the norm function to the $Div(K(C))$.

3.6.3.1 Preservation of the DLP under the norm map

In the case of Norm function, it is proved in [Die03] that the kernel of Norm function does not have a subgroup of large prime order if there is no intermediary field L such that $k \subsetneq L \subsetneq K$. If there is such a intermediary field, we can replace n by n', a prime divisor of order n, in order to get rid of such an intermediary field and make sure that the DLP got preserved under this map.

CHAPTER 4

Even Characteristic Function Fields

In this chapter we study the problem of attacking the DLP of elliptic curves defined over a field of even characteristic. This is a critical case because such elliptic curves are of special attention for cryptographers, due to the computational advantage.

Therefore in this chapter we suppose that all fields have characteristic 2 and our prime base finite field is $\mathbb{F}_2$. Parallel to the notations we defined in section 3.3 we have $k = \mathbb{F}_{2^r}$ and $K = \mathbb{F}_{(2^r)^n}$. E is an elliptic curve defined over the field K.

As we mentioned in the previous chapter, to attack a curve we need to accomplish the following two steps:

1. Find a function field extension of $K(E)$, namely $K(C)$.

2. Find an automorphism of $K(C)$, namely σ such that $\sigma|_K = \sigma_{K/k} \in \mathrm{Gal}(K/k)$, and $\mathrm{ord}(\sigma) = n$

These two are not totally independent steps. We cannot expect that for a random general extension $K(C)$, we succeed to find such an automorphism like σ. However, with Hypothesis 3.3.3, that we assume in this thesis, we can guarantee that the Frobenius of K/k can be extended to such an automorphism. In [Hes04] the procedure of the GHS attack is studied in detail for a special set of rational subfields of $K(E)$. We describe this work in this chapter. We also generalize some of the result to more general rational subfields of $K(E)$.

We start this chapter with a concrete example of going through the algorithm. We then define the concept of minimal σ-polynomial and we study the K-module related to it. We discuss the implementation of the GHS attack by using quadratic extensions. This generalizes the case of Artin-Schreier extensions discussed in [GHS02b], [BSS05] and [Hes04]. In this chapter, we also discuss the result of [BSS05] for Artin-Schreier extensions. Finally, we discuss the security of a function field against the GHS attack and we define a field security index to assess it.

4.1 Finding a suitable $K(C)$ as an extension of $K(E)$

To construct $K(C)$, we find $K(z)$ such that equation of E in terms of z has σ-polynomial with the smallest degree possible. We will define the concept of "σ-polynomial" in Section 4.2, but we illustrate the concept by a complete example before formally define it. Let

$$\begin{aligned} k &:= \mathbb{F}_2 \\ n &:= 7 \\ K := k(\theta) \text{ such that } \theta \in \bar{k} \text{ satisfies } &\theta^7 + \theta + 1 = 0 \\ E/K &:= y^2 + xy + x^3 + \theta \end{aligned}$$

We choose here $z = x$. Now we want to extend the field extension $K(E)/K(z)$ using the procedure described in last chapter. Let

$$P(Y) := Y^2 + zY + z^3 + \theta \in K(z)[Y]$$

Let $y_0 = y$. Polynomial $P(Y)$ is reducible over $K(E)$ since y_0 is a root of $P(Y)$. Let σ acts on on $P(Y)$ by acting on its coefficients, Then

$$P^{\sigma}(Y) = Y^2 + zY + z^3 + \theta^2$$

is irreducible over $K(E)$. We can extend $K(E)$ by adding a root of this polynomial, say y_1, to get $K_1 := K(E)(y_1)$. Continuing the procedure, we consider

$$P^{\sigma^2}(Y) = Y^2 + zY_2 + z^3 + \theta^4 \in K(z)[Y].$$

This polynomial is irreducible over $K_1 := K(E)(y_1)$. Let $K_2 := K_1(y_2)$, where y_2 is a root of $P^{\sigma^2}(Y)$. Now we consider

$$P^{\sigma^3}(Y) = Y^2 + zY + z^3 + \theta^8 \in K(z)[Y]$$

which is irreducible over K_2. Let $K_3 = K_2(y_3)$ where y_3 is a root of $P^{\sigma^3}(Y)$.

We now consider

$$P^{\sigma^4}(Y) = Y^2 + zY + z^3 + \theta^{16} \in K(z)[Y].$$

This polynomial is reducible over K_3. Indeed, adding the three equations

$$y_0^2 + zy_0 + x^3 + \theta = 0$$
$$y_2^2 + zy_2 + x^3 + \theta^4 = 0$$
$$y_3^2 + zy_3 + x^3 + \theta^8 = 0$$

and considering the fact that we work in characteristic 2, we get

$$(y + y_2 + y_3)^2 + x(y + y_2 + y_3) + x^3 + (\theta^8 + \theta^4 + \theta) = 0.$$

Since

$$\theta^8 + \theta^4 + \theta = \theta^4 + \theta^2 = (\theta^2 + \theta)^2 = (\theta^8)^2 = \theta^{16} \text{ in } K = \mathbb{F}_2(\theta),$$

we have

$$(y + y_2 + y_3)^2 + x(y + y_2 + y_3) + x^3 + \theta^{16} = 0$$

Then $y+y_2+y_3$ is a root of $P^{\sigma^4}(Y)$, and $P^{\sigma^4}(Y)$ is reducible in $K_3 = K(z)(y_0, y_1, y_2, y_3)$. The good news is that no matter what power of σ more than 4 we apply, we still get reducible polynomial. This will be proved in Theorem 4.3.1.

The above example shows a desirable situation in which $K(C) = K_3$ is of degree $2^3 = 8$ over $K(E)$ and not $2^6 = 32$. In this way, C should have smaller genus and it should be easier to attack.

Obviously one can choose another equation for E and one would not get a reducible polynomial before you extend the function field 7 times. In this regard, it is important to find z such that the defining polynomial of $K(E)/K(z)$ becomes reducible after smaller number of application of σ. In next section we will study the defining equation from this perspective.

4.2 Minimal σ-polynomial

To study the complexity of the attack, we need to formalize the example of the previous section. We stopped the process of extending the function field as soon as we had a

linear combination of conjugates of the curve's equation equal to zero. To formalize the procedure we define a $\mathbb{F}_2[t]$–module over the additive group K^+ of K, following [BSS05].

Proposition 4.2.1 *Let K^+ be the additive group of K and let $\mathbb{F}_2[t]$ act on K^+ by*

$$f(t) * \mathbf{b} := \sum_{i=0}^{\ell} a_i \sigma^i(\mathbf{b}) \text{ for } \mathbf{b} \in K \text{ and } f(t) = \sum_{i=0}^{\ell} a_i t^i \in \mathbb{F}_2[t].$$

This makes K^+ into an $\mathbb{F}_2[t]$-module.

Proof. First we prove the distributivity. We have

$$\begin{aligned} f(t) * (\mathbf{b}_1 + \mathbf{b}_2) &= \sum_{i=0}^{\ell} a_i \sigma^i(\mathbf{b}_1 + \mathbf{b}_2) \\ &= \sum_{i=0}^{\ell} a_i((\sigma^i(\mathbf{b}_1) + \sigma^i(\mathbf{b}_2)) \\ &= \sum_{i=0}^{\ell} a_i \sigma^i(\mathbf{b}_1) + \sum_{i=0}^{\ell} a_i \sigma^i(\mathbf{b}_2) \end{aligned}$$

$$\begin{aligned} &= f(t) * \mathbf{b}_1 + f(t) * \mathbf{b}_2 (f_1(t) + f_2(t)) * (\mathbf{b}) \\ &= \sum_{i=0}^{\ell} (a_{1i} + a_{2i}) \sigma^i(\mathbf{b}) \\ &= \sum_{i=0}^{\ell} a_{1i} \sigma^i(\mathbf{b}) + \sum_{i=0}^{\ell} a_{2i} \sigma^i(\mathbf{b}) \\ &= f_1(t) * \mathbf{b} + f_2(t) * \mathbf{b} \end{aligned}$$

For the associativity, first we prove the result for monomials. Suppose $f = \sum_{i=0}^{\ell} a_i t^i$, then for a general monomial $t^{\ell'}$ we have:

$$t^{\ell'} * (f(t) * \mathbf{b}) = t^{\ell'} * \sum_{i=0}^{\ell} a_i \sigma^i(\mathbf{b}) = \sigma^{\ell'}(\sum_{i=0}^{\ell} a_i \sigma^i(\mathbf{b})) = \sum_{i=0}^{\ell} a_i \sigma^{i+\ell'}(\mathbf{b}) = (t^{\ell'} f(t)) * \mathbf{b}$$

For the general case, we use induction over the degree of f. For degree 0 polynomials $*$ is trivially associative. Suppose for all f' such that $\deg(f') < \ell$ and $f' = f_2' f_1'$ we have for all $\mathbf{b} \in K$

$$f'(t) * \mathbf{b} = f_2'(t) * (f_1'(t) * \mathbf{b}).$$

Suppose that $f(t) = f_2(t) f_1(t)$ such that $\deg(f) = \ell$. We want to prove that $f(t) * \mathbf{b} = f_2(t) * (f_1(t) * \mathbf{b})$. Suppose that $\deg(f_2) = \ell'$ and $f_2(t) = t^{\ell'} + g(t)$

$$(f_2)(f_1) * \mathbf{b} = (t^{\ell'} + g(t))(f_1) * \mathbf{b} = (t^{\ell'})(f_1) * \mathbf{b} + g(t)(f_2) * \mathbf{b}.$$

The first term of the right hand of the equality is equal to $(t^{\ell'} * (f_1) * b)$ as $t^{\ell'}$ is a monomial and we proved the associativity when one of the factors is monomial. And the second one is equal to $g(t) * ((f_1) * \mathbf{b})$ by induction as the degree $g(t)(f_1)$ is strictly less than $\deg(f)$. So we have

$$\begin{aligned} (f_2)(f_1) * \mathbf{b} &= t^{\ell'} * ((f_1) * \mathbf{b}) + g(t) * ((f_1) * \mathbf{b}) \\ &= (t^{\ell'} + g(t)) * ((f_1) * \mathbf{b}) = (f_2) * ((f_1) * \mathbf{b}). \end{aligned}$$

□

Definition 4.2.2 *Let $f(t) \in \mathbb{F}_2[t]$. We define a $\mathbb{F}_2[t]$-linear map, $L_{f(t)}$, on the module defined in Proposition 4.2.1 by*

$$L_{f(t)}(\mathbf{b}) := f(t) * \mathbf{b}$$

The kernel of $L_{f(t)}$ is a submodule of K^+ denoted by S_f.

It is straightforward that

- $K = S_{t^n+1}$.
- $f(t) \mid g(t) \Rightarrow S_f \subseteq S_g$.

The other proprieties that we are interested in are as follows:

Proposition 4.2.3 *Suppose that f, g are two polynomials in $\mathbb{F}_2[t]$ and $[f, g]$ is their least common multiple. We have*

$$S_f \oplus S_g = S_{[f,g]}$$

Proof. Let f', g' be such that, $[f, g] = f'f = g'g$ for some polynomials f', g'.

($\subseteq$) For all $\mathbf{b}_1 \in S_f, \mathbf{b}_2 \in S_g, [f, g] * (\mathbf{b}_1 + \mathbf{b}_2) = [f, g] * (\mathbf{b}_1) + [f, g] * (\mathbf{b}_2) = f'f * \mathbf{b}_1 + g'g * \mathbf{b}_2 = f' * (f * \mathbf{b}_1) + g' * (g * \mathbf{b}_2) = 0 + 0 = 0$ therefore, $\mathbf{b}_1 + \mathbf{b}_2 \in S_{[f,g]}$.

($\supseteq$) First we treat the case that $(f, g) = 1$ this means $[f, g] = fg$. Now for all $\mathbf{b} \in S_{fg}$ we have $g(t)f(t) * \mathbf{b} = 0$, by definition of S_{fg} and associativity of $*$, this means that $f(t) * \mathbf{b} \in S_g$. Similarly we have $g(t) * \mathbf{b} \in S_f$. Now as $(f, g) = 1$, there exist polynomials h_1, h_1 such that $fh_1 + gh_2 = 1$. Additionally we have $f(t) * \mathbf{b} \in S_g$ So $(h_1 f) * \mathbf{b} = (h_1) * (f * \mathbf{b}) \in S_g$ as S_g is a $\mathbb{F}_2-$submodule. Similarly we have $gh_2 * \mathbf{b} \in S_f$. This two tells us that

$$\begin{aligned} fh_1 * \boldsymbol{b} + gh_2 * \boldsymbol{b} &= (fh_1 + gh_2) * \boldsymbol{b} \\ &= 1 * \boldsymbol{b} = \boldsymbol{b} \in S_f \oplus S_g. \end{aligned}$$

Now suppose that $h = (f, g)$, using Lemma 4.2.4, we can write $h = h_1 h_2$ such that $(f/h_1, g/h_2) = 1$ and $[f, g] = [f/h_1, g/h_2] = (f/h_1)(g/h_2)$. Using the co-prime case

we know that $S_{[f,g]} = S_{f/h_1} \oplus S_{g/h_2}$. But we know that $S_{f/h_1} \subset S_f$ and $S_{f/h_2} \subset S_g$, therefore, $S_{[f,g]} \subseteq S_f \oplus S_g$.

□

Lemma 4.2.4 *Suppose that f, g are two elements of a unique factorization domain U, and $h = gcd(f, g)$. We can always decompose $h = h_1 h_2$ such that $(f/h_1, g/h_2) = 1$ and $[f/h_1, g/h_2] = [f, g]$.*

Proof. Suppose that $f = p_1^{\alpha_1} ... p_n^{\alpha_n}, g = p_1^{\beta_1} ... p_n^{\beta_n}$ be the prime decomposition of f, g in U. Then $h = p_1^{\min(\alpha_1,\beta_1)} ... p_n^{\min(\alpha_1,\beta_1)}$. Now set $h_1 := 1$, and follow following algorithm

$$\text{For } 1 \leq i \leq n: \text{ if } \alpha_i \leq \beta_i \text{ such that } h_1 := h_1 \times p_i^{\min(\alpha_i,\beta_i)}$$

Then we set $h_2 := h/h_1$. Now suppose that $p|(f/h_1, g/h_2)$ this means that there exists prime p_i such that $p_i|(f, g)$. Now if $\alpha_i \leq \beta_i \Rightarrow p_i \nmid f/h_1$. Otherwise $\beta_i < \alpha_i$ therefore $p_i^{\beta_i} \mid h_2$ so $p_i \nmid g/h_2$ which is a contradiction.

□

Proposition 4.2.5 *For any $\mathbf{b} \in K$, there is a unique monic polynomial $f(t) \in \mathbb{F}_2[t]$ with minimal degree such that $\mathbf{b} \in S_f$. Furthermore, if $g(t) * \mathbf{b} = 0$ then $f(t)$ divides $g(t)$.*

Proof. Suppose $I = \{g(t) \in \mathbb{F}_2[t] \text{ such that } g(t) * \mathbf{b} = 0\}$ the range of degree function on set I is well-ordered and we can choose $f(t)$ as one of those who has minimum degree. Now suppose that $g(t) * \mathbf{b} = 0$. By minimality of degree, we have, $\deg(g) \geq \deg(f)$, dividing by $f(t)$ we have $g(t) = q(t)f(t) + r(t)$ such that $\deg(r(t)) \leq \deg(f(t))$ therefore $0 = g(t) * \mathbf{b} = q(t)f(t) * \mathbf{b} + r(t) * \mathbf{b}$. But we know that $f(t) * \mathbf{b} = 0$ so we must have $r(t) * \mathbf{b} = 0$ which contradict the minimality condition of f. So $f(t)$ is unique up to multiplication of a unit element. However being monic, make $f(t)$ is unique.

□

Based on Proposition 4.2.5, we define the concept of "minimal σ-polynomial".

Definition 4.2.6 Minimal σ-polynomial *For any element $\mathbf{b} \in K^+$, the minimal σ-polynomial of $\mathbf{b}$ is the unique monic polynomial $f(t) \in \mathbb{F}_2[t]$ with minimal degree such that $f(t) * \mathbf{b} = 0$.*

We can extend the definition of operation $*$ between $f(t) = \sum_{i=0}^{m} a_i t^i$ and a rational function $g(z) \in K(z)$ as follow

$$f(t) * g(z) = \sum_{i=0}^{m} a_i g^{\sigma^i}(z).$$

Definition 4.2.7 *For a rational function $g(z) \in K(z)$, the minimal σ-polynomial of $g(z)$ is monic polynomial $m_g(t)$ with minimal degree such that $m_g(t) * g(z) = 0$.*

Knowing the minimal σ-polynomial of the coefficient of a polynomial $g(z)$, Corollary 4.2.8 to Proposition 4.2.3 helps us to find the the minimal σ-polynomial of $g(z)$.

Corollary 4.2.8 *Let $g(z) = \sum_{j=0}^{\ell} \mathbf{b}_j z^j \in K[z]$ be a polynomial. Then $m_g(t) = [m_{\mathbf{b}_j}$ such that for $0 \leq j \leq \ell]$ and $S_{m_g} = \bigoplus_{j=1}^{\ell} S_{m_{\mathbf{b}_j}}$.*

It is worthy of mention that unlike the case of minimal polynomial for an element of an algebraic extension, $f(t)$ is not necessarily irreducible. For example consider $k = \mathbb{F}_2$ and $K = \mathbb{F}(\theta)$ where $\theta^2 + \theta + 1 = 0$. We know that $\sigma^2(\theta) = \theta^4 = (\theta + 1)^2 = \theta^2 + 1 = \theta \Rightarrow \theta \in S_{t^2+1}$. It is clear that $t^2 + 1$ is not irreducible over $\mathbb{F}_2$, however we claim it is the minimal σ-polynomial of θ. If not, then there exists a $f(t)$ such that $\deg(f(t)) \leq 2$ and $f(t) * \theta = 0$, however by uniqueness condition of Proposition 4.2.5, $\deg(f(t)) \neq 2$. The set of monic polynomials of degree 1 over K is $\{t, t+1\}$. Similarly since $\sigma(\theta)^2 = \theta^2 = \theta + 1 \neq \theta$, $\theta \notin S_{t+1}$. Similarly $\theta^2 \neq 0$ implies that $\theta \notin S_t$. So, the $\sigma-$minimal polynomial of θ is the reducible polynomial $t^2 + 1$.

4.3 General quadratic extensions

Now that we have developed the theory of our submodules, we use it to study the complexity of the GHS attack for different curves. Suppose that E is defined by the equation

$$P_E : Y^2h(z) + Yg(z) + f(z) = 0 \text{ such that } f(z) \in K(z), h(z), g(z) \in k(z) \qquad (4.1)$$

Then $K(E) = K(z)(y_0)$ where y_0 is a root of (4.1). We do not confine ourself to represent the equation of the curve in Weierstrass form or any other special form, because we want to allow the attacker to use the form which gives the opportunity of reaching the "best" extension $K(C)$ for attacking the original curve. The only constrains are:

- we require that the equation of the curve is quadratic in Y. This allows us to use the linearity of square in characteristic 2 to implement a simpler attack.
- We require that $h(z), g(z)$ be in $k(z)$. This also important for the algorithm of the attack.

Also if in (4.1) the rational function, $f(z)$ is defined as

$$f(z) = \frac{p(z)}{q(z)}$$

such that $q(z) \notin k[z]$, we can replace it with

$$f(z) = \frac{\frac{Norm_{K(z)/k(z)}(q(z))}{q(z)}p(z)}{Norm_{K(z)/k(z)}(q(z))}.$$

So we can suppose that all rational functions in (4.1) have denominators in $k[z]$.

We also assume that (4.1) is not defined over any proper subfield of $K(z)$, otherwise one can simply replace $K(z)$ with that subfield.

Let σ be as defined in Chapter 3. Then

$$P_E^\sigma = Y^2h(z)^\sigma + Yg(z)^\sigma + f(z)^\sigma = Y^2h(z) + Yg(z) + f^\sigma(z) = 0.$$

in which

$$f^\sigma(z) = \frac{p^\sigma(z)}{q^\sigma(z)}.$$

If P_E^σ is irreducible over $K(z, y_0)$, we add its root y_1 and we continue by acting on coefficient of P_E^σ. This leads to Algorithm 4.1 for finding $K(C)$. This algorithm guarantees that $[K(C) : K(z)]$ does not exceed 2^m, where m is the degree of $m_f(t)$ the minimal σ-polynomial of $f(z)$ in(4.1).

Algorithm 4.1 FINDING $K(C)$ USING GENERAL EXTENSION

Require: $K(E)$ an elliptic function field.
$K(z)$ a rational subfield of $K(E)$ such that $K(E) = K(z)(y_0)$ is a quadratic extension of $K(z)$.
$P_E(Y) \in K(z)[Y]$ the minimal polynomial of y_0 over $K(z)$, under form(4.1).
Ensure: $K_m = K(C)$ (as defined 3.3.3.1) such that $[K(C) : K(z)] \leq 2^m$.

1: Compute $m_f(t) \in \mathbb{F}_2[t]$ be the minimal σ-polynomial of $f(z)$, where $f(z)$ is given by (4.1).
2: $m \leftarrow \deg(m_f(t))$
3: $K_0 \leftarrow K(z)(y_0) = K(E)$
4: **for all** i such that $1 \leq i \leq m$ **do**
5: $\quad P_E^{\sigma^{i-1}} \leftarrow Y^2h(z) + Yg(z) + (f)^{\sigma^{i-1}}(z)$
6: $\quad K_i \leftarrow K_{i-1}(y_i)$ where y_i is a root of $P_E^{\sigma^i}$
7: **end for**
8: **return** K_m

We now need to prove that using the above algorithm, we find the function field $K(C)$ of the original GHS attack so it has same useful properties which allow us to descend it to $k(C_0)$ and transfer the DLP to $Cl^0(k(C))$.

Theorem 4.3.1 Validity of Algorithm 4.1 for general quadratic extensions *If m_f is irreducible in $K[t]$ then the outputs of Algorithm 3.1 and Algorithm 4.1 are equal.*

Proof. The Algorithm 4.1 is essentially a special case of Algorithm 3.1, beside the fact that it stops after m steps and unlike to algorithm 3.1, it does not loop for n steps.

We argue that if Algorithm 4.1 does not stop (and continue as Algorithm 3.1 does) we would get exactly the same result. In the context of Algorithm 4.1, this means that the polynomials $P_E^{\sigma^i} \in K(z)[Y]$ are reducible over $K_m[Y]$ for $i = m, \ldots, n$. First we prove that if we apply σ m times we get a reducible polynomial over K_m. In the other words we want to show that

$$\sigma^m(Y^2h(z) + Yg(z) + f(z)) = Y^2h(z) + Yg(z) + \sigma^m(f)(z) \tag{4.2}$$

is reducible over K_m.

According to the equation 4.1, $m_f * h(z) = h(z), m_f * g(z) = g(z)$. Therefore after running the loop for m times ($i = 0 \ldots m-1$), we have following relationships which all defined over $K_m = K(z, y_0, ..., y_{m-1})$

$$\begin{aligned} y_0^2h(z) + y_0g(z) + f(z) &= 0 \\ y_1^2h(z) + y_1g(z) + \sigma(f)(z) &= 0 \\ y_3^2h(z) + y_1g(z) + \sigma^2(f)(z) &= 0 \\ &\vdots \\ y_{m-1}^2h(z) + y_{m-1} * g(z) + \sigma^{m-1}(f)(z) &= 0. \end{aligned}$$

Now we know that $m_f * f = 0$. So we have

$$
\begin{aligned}
a_0(y_0^2h(z) + y_0g(z) + f(z)) + \\
a_1(y_1^2h(z) + y_1g(z) + \sigma(f)(z)) + \\
a_2(y_2^2h(z) + y_1g(z) + \sigma^2(f)(z) + \\
\vdots \\
a_{m-1}(y_{m-1}^2h(z) + y_{m-1}g(z) + \sigma^{m-1}(f)(z)) = 0
\end{aligned}
$$

Distributing a_i's we get

$$\sum_{i=0}^{m-1} a_i y_i^2 h(z) + \sum_{i=0}^{m-1} a_i y_i g(z) + \sum_{i=0}^{m-1} a_i f(z)$$

Using the fact that we are in characteristic 2 we can write the sum of squares of $y_i's$ as square of their sum.

$$(\sum_{i=0}^{m-1} a_i y_i)^2 h(z) + (\sum_{i=0}^{m-1} a_i y_i) g(z) + \sum_{i=0}^{m-1} a_i \sigma^i(f)(z) \tag{4.3}$$

Let $p(z) = \sum_{j=0}^{\ell} \mathbf{b}_j z^j$. as the denominator of $q(z) \in k[z]$ we deduce that $m_f(t) = m_p(t)$. By definition of m_f, we know that for

$$
\begin{aligned}
\text{For } 0 \leq j \leq \ell \text{ we have } m_f * \mathbf{b}_j &= \sum_{i=0}^{m} a_i \sigma^i(\mathbf{b}_j) = 0 \\
\Leftrightarrow \sigma^m(\mathbf{b}_j) &= \sum_{i=0}^{m-1} a_i \sigma^i(\mathbf{b}_j) \\
\Rightarrow \sum_{j=0}^{\ell} \sigma^m(\mathbf{b}_j) z^j &= \sum_{j=0}^{\ell} \sum_{i=0}^{m-1} a_i \sigma^i(\mathbf{b}_j) z^j
\end{aligned}
$$

Now because σ is isomorphism we can write the last equality as follows

$$p^{\sigma^m}(z) = \sum_{i=0}^{m-1} a_i p^{\sigma^i}(z)$$

Now dividing both side by $q(z)$ considering the fact that $q^\sigma(z) = q(z)$, we get

$$f^{\sigma^m}(z) = \sum_{i=0}^{m-1} a_i f^{\sigma^i}(z)$$

Substituting this equality in 4.3 we get

$$(\sum_{i=0}^{m-1} a_i y_i)^2 h(z) + (\sum_{i=0}^{m-1} a_i y_i) g(z) + f^{\sigma^m}(z) = 0$$

This means that $\sum_{i=0}^{m-1} a_i y_i$ is a root for $(Y^2 h(z) + Y g(z) + f(z))^{\sigma^m}$ and therefore this equation is not irreducible.

□

4.4 Genus of $K(C)$

The main appeal of Artin-Schreier and Kummer extensions considered in [Hes04] and [Die03], is that it is relatively easy to compute the genus of $K(C)$ in these cases. However, there is no known method to compute the genus of $K(C)$ in the general case of the GHS attack.

In the case of quadratic extensions, however, we can compute an upper bound for the genus of $K(C)$.

Proposition 4.4.1 *Let $K(C)$ be the function field provided by Algorithm 4.1 and let g_C be its genus. Then*

$$g_C \leq m2^{m+1} + 1.$$

Proof. For $i \in \{0, ..., m\}$, let $F_i := K(z, y_i)$ with y_i is defined in Algorithm 4.1. It is easy to see that for $i \in \{1, ..., m\}$, $K_i = K(y_0, \ldots, y_i)$ is equal to $K_{i-1}F_i$, the compositum of K_{i-1} and F_i. We want to find an upper bound for g_i, the genus of K_i. We can use Theorem III.10.3 of [Sti93], which bounds the genus of compositum of K_{i-1} and F_i by their genera and the extension degree of K_i. This gives

$$g_i \leq [K_i : K_{i-1}]g_{i-1} + [K_i : F_i]g_{F_i} + ([K_i : K_{i-1}] - 1)([K_i : F_i] - 1). \tag{4.4}$$

The GHS construction is done by quadratic extensions so $[K_i : K_{i-1}] = 2$. Also we know that $K_i = F_i(y_0, y_1, ..., y_i - 1)$ so $[K_i : F_i] = 2^i$. Moreover, $F_i \simeq K(E)$, a function field of genus 1. Substituting these in (4.4), we get the recursive bound

$$g_i \leq 2g_{i-1} + 2^i + 2^i - 1 = 2g_{i-1} + 2^{i+1} - 1. \tag{4.5}$$

Using induction, we climb the tower of paired function fields shown in Diagram 4.1 to get the explicit bound

$$g_i \leq i2^{i+1} + 1.$$

For $i = 1$, the recursive bound and explicit bound are the same and show that $g_1 \leq 5$. Now assuming the explicit bound for g_{i-1} and using the recursive bound for g_i, we have:

$$\begin{aligned} g_i &\leq 2g_{i-1} + 2^{i+1} - 1 \\ \Rightarrow g_i &\leq 2((i-1)2^{(i-1)+1} - 1)) + 2^{i+1} - 1 \\ &= (i-1)2^{i+1} + 2 + 2^{i+1} - 1 \\ &= i2^{i+1} + 1. \end{aligned}$$

Using $i = m$ we get the result.

□

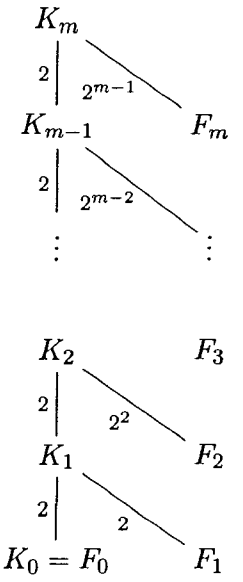

Diagram 4.1: The induction tower used in the proof of Proposition 4.4.1.

4.5 Artin-Schreier extensions

In the last section, we considered the GHS attack for an elliptic curve equation which is written in the form of (4.1). However, if we limit the type of the intermediate extension, i.e. $K_1/K(z), K_2/K_1, ..., K_m/K_{m-1}$, we can prove more precise results. The two special extensions which are studied in the GHS attack literature are the *Artin-Schreier extensions* ([GHS02b], [Hes04]) and the *Kummer extensions* ([Die03], [The03]).

If the elliptic curve is given by (4.1), the extension $K(E)/K(z)$ and consequently K_i's are not Kummer extensions, since the degree of extension and the characteristic of the base field should be co-primes.

Additionally, according to [The03] there is no non-supersingular elliptic curve over characteristic 2 whose function field is a Kummer extension over some rational subfield

$K(z)$.

However, the case of Artin-Schreier extensions for the GHS attack for even characteristic function fields, is well studied and there is a rich literature on this subject for example [GHS02b], [Hes04], [BSS05]. We discuss the main features of this approach, mostly from [BSS05].

Definition 4.5.1 Artin-Schreier extensions for fields of even characteristics *Suppose that F is a function field of characteristic 2, and $g \in F$ is an element such that for all $h \in F$ we have $g \neq h^2 - h$, Therefore the polynomial*

$$Y^2 - Y - u = 0 \tag{4.6}$$

does not have any root in F. Let $F' = F(y)$, where y is a root of (4.6). Then F'/F is called Artin-Schreier extension

If we work in a characteristic $p \neq 2$ then the definition of Artin-Schreier extensions remains the same except that (4.6) would change to $Y^p - Y - u = 0$. These extensions have many nice properties. For example, an Artin-Schreier extension is always Galois. To see that, suppose y is a root of 4.6. Then we have

$$(y+1)^p - (y+1) - u = y^p + 1^p - y - 1 - u = y^p - y - u = 0,$$

and $y + 1$ is a root of that equation as well. Inductively $y + i$ such that $0 \leq i < p$ are roots of this equation. Therefore by adding one root of this equation, it happens that all other roots also belong to the same extension, which means the extension is the splitting field of the equation with p different roots, and is Galois consequently.

Although any separable quadratic extension is Galois, the fact that the 2 roots are y and $y + 1$ is special to Artin-Schreier extensions, and will be important for us. In

characteristic 2 we can change also the general form of Artin-Schreier equation to

$$Y^2 + Y + u = 0 \tag{4.7}$$

4.5.1 Elliptic curve equation in Artin-Schreier form

We want to show that all elliptic curves can be described by an Artin-Schreier equation by means of a simple change of variables. Proposition A.1.1 [Sil94] says that any elliptic curve over a field characteristic two can be defined using either of the following equations depending on the value of $j(E)$

- $Y^2 + a_3Y = x^3 + a_4x + a_6$ if $j(E) = 0$
- $Y^2 + xY = x^3 + a_2x^2 + a_6$ if $j(E) \neq 0$

For the case that $j(E) = 0$ we have

$$Y^2 + a_3Y = x^3 + a_4x + a_6. \tag{4.8}$$

One can replace $Y' = Y/a_3 \Rightarrow Y = a_3Y'$ and the result would be

$$a_3^2Y'^2 + a_3^2Y' = x^3 + a_4x + a_6$$

Dividing both side by a_3^2 we get

$$Y'^2 + Y' + g(x) = 0 \text{ where } g(x) := (x^3 + a_4x + a_6)/a_3, \tag{4.9}$$

which is in the Artin-Schreier form.

On the other hand, if $j(E) \neq 0$, we can describe E with following equation

$$Y^2 + xY = x^3 + a_2x^2 + a_6 \text{ such that } j(E) = 1/a_6. \tag{4.10}$$

In this case, as it is done in [BSS05], we can write $Y = \gamma Y'/z + a_6^{1/2}, x = \gamma/z$. We get

$$\frac{\gamma^2 Y'^2}{z^2} + a_6 + \frac{Y'\gamma^2}{z^2} + \frac{a_6^{1/2}\gamma}{z} = \gamma^3/z^3 + \frac{a_2\gamma^2}{z^2} + a_6.$$

Cancelling a_6 from both side and multiplying both side by $\frac{z^2}{\gamma^2}$ we get

$$Y'^2 + Y' + a_6^{1/2}/\gamma = \gamma/z + a_2,$$

and rearranging the terms we get

$$Y'^2 + Y' + f(z) = 0 \text{ such that } f(z) := \frac{\gamma}{z} + a_6^{1/2}/\gamma z + a_2 \tag{4.11}$$

which is also an Artin-Schreier equation.

Elliptic curves with zero j-invariant are not considered secure in cryptography [HMV03] and for that reason we do not consider that case in following process.

4.5.2 The GHS attack with Artin-Schreier extensions

To implement the GHS attack with Artin-Schreier extension we use the Algorithm 4.1, however assume that E is given by an equation of type 4.11. This means that we make a set of Artin-Schreier equations

$$\begin{aligned} Y^2 + Y + f(z) &= 0 \\ Y^2 + Y + f^{\sigma}(z) &= 0 \\ Y^2 + Y + f^{\sigma^2}(f)(z) &= 0 \\ &\vdots \\ Y^2 + Y + f^{\sigma^{m-1}}(z) &= 0, \end{aligned}$$

and we want to extend $K(E)$ to a field $K_m = K(C)$, such that all these equations has a roots. Using the result of Theorem 4.3.1, we know that the remaining $n-m$ equations

$$\begin{aligned} Y^2 + Y + f^{\sigma^m}(z) &= 0 \\ Y^2 + Y + f^{\sigma^{m+1}}(z) &= 0 \\ &\vdots \\ Y^2 + Y + f^{\sigma^{n-1}}(z) &= 0 \end{aligned}$$

also have roots in K_m. Moreover, for any $d \in K(z)$ we know that the equation $Y^2 + Y + d^2 + d$ has a solution d.

Finally, it is easy to see that if K_m contains roots of different Artin-Schreier equations such as $y_1^2 + y_1 + a = 0$ and $y_2^2 + y_2 + b = 0$ then

$$(y_1 + y_2)^2 + (y_1 + y_2) + a + b = 0$$

therefore it also has roots of $Y^2 + Y + a + b = 0$. Therefore, independently of how we define and construct K_m, we can redefine it as follow: Let Δ be the $\mathbb{F}_2[t]$−submodule generated by all elements $d^2 + d$ for any $d \in K(z)$ and $f(z)$. Then K_m is the smallest field such that for any $\mathbf{b} \in \Delta$, $Y^2 + Y + \mathbf{b} = 0$, has a root in K_m.

The fact that Δ is defined an $\mathbb{F}_2[t]$-submodule means that Δ is closed under operator $*$, therefore $Y^2 + Y + \sigma^i(f)(z)$ has solution in K_m for any i. This shows that the field K_m generated by above procedure contains the $K(C)$ needed for the GHS attack.

The following theorem from [BSS05] and [Hes04], describes completely the situation when the extension $K_m/K(z)$ is formed by Artin-Schreier intermediate extensions.

Theorem 4.5.2 The GHS attack for Artin-Schreier extensions *Suppose E is an elliptic curve defined using Equation 4.11. Let $\beta := \sqrt{a_2}/\gamma$. Let m_γ, m_β and $m_{f(z)}$ be*

the minimal σ-polynomials of γ and β and $f(z)$ respectively. Let C_0 be the curve defined over k (as defined in Section 3.3.1), such that $K(C_0) = K(C) = K_m$, Then we have

- *if there exists $d \in K$ such that $a_2 = d^2 + d$, then $m_{f(z)} = [m_\gamma, m_\beta]$; otherwise $m_{f(z)} = [m_\gamma, m_\beta, t+1]$.*
- *the genus of C_0 is equal to*

$$2^{\deg([m_\gamma,m_\beta])} - 2^{\deg([m_\gamma,m_\beta])-\deg(m_\gamma)} - 2^{\deg([m_\gamma,m_\beta])-\deg(m_\beta)} + 1.$$

- *if $\gamma \in k$ or $\beta \in k$ then C_0 is hyperelliptic.*

4.6 Security evaluation of composite fields

We now want to investigate how secure is a specific even characteristic field with composite degree against the GHS attack. The security of these fields is studied in [JMS01], [BSS05], [Sma01], [MTW04], [MMT01], and [MT06] [MQ01] for the GHS attack using Artin-Schreier extensions. However, here we consider that our curve is defined in the more general form of (4.1) and we do not confine our study to curves with Artin-Schreier equation.

According to Theorem 4.3.1, the final factor that defines the efficiency of attack is the degree of m_f in Algorithm 4.1.

4.6.1 Security of coefficients of $f(z)$

we know that $m_f(t)$ is the minimal degree among the polynomials which annihilate all coefficients of numerator of $f(z)$.

The Corollary 4.2.8, means that a curve with low security (by low security, we mean small m) is a curve which is expressible as (4.1) such that

- All coefficients of $f(z)$ have minimal σ-polynomial of low degree.
- The least common multiplier of those polynomials have small degree.

In following sections we focus on the first condition. The second condition, however, is more complicated and need more research.

4.6.2 On polynomial $t^n + 1$

Clearly, for all $\mathbf{b} \in K$ we have $\sigma^n(\mathbf{b}) + \mathbf{b} = 0$, so $\mathbf{b} \in S_{t^n+1}$ and by Proposition 4.2.5, $m_{\mathbf{b}} \mid t^n + 1$. Therefore the factorization of $t^n + 1$ is related to the size of $m = \deg(m_f)$.

Lemma 4.6.1 (Lemma 2, [MQ01]) *Let n be an odd prime. Let $d = ord_n(2)$ be the order of $2 \in \mathbb{Z}/n\mathbb{Z}$. Write $n = ds + 1$. Then $t^n + 1 = f_0(t)f_1(t)\cdots f_s(t)$ over $\mathbb{F}_2[t]$. where*

- $f_0(t) = t + 1$.
- *For $1 \leq i \leq s, f_i(t)$ is irreducible over $\mathbb{F}_2$.*
- *For $1 \leq i \leq s, \deg(f_i) = d$.*
- *For $i \neq j$ we have $(f_i, f_j) = 1$.*

To categorize the elements of K according to their security level toward the GHS attack, we associate following set to each element.

Definition 4.6.2 *Suppose n is prime, and the f_i's are as defined in Lemma 4.6.1. We define set I_b for any element $b \in K$ by*

$$I_{\mathbf{b}} = \{i \text{ such that } f_i(t) \text{ divides } m_b(t)\}$$

Fixing an odd prime n, the number of elements of I_b is a measure of the degree of $m_b(t)$ which is related to $m = \deg(m_f(t))$. As an attacker, we want m to be small. We now compare the various elements b of a given field K from that point of view. The least secure elements among the elements of K, are those with $I_{\mathbf{b}} = 1$. Another interesting questions, is to consider a specific subset $S \subseteq \{0, \ldots, s\}$, and ask what are the elements $\mathbf{b}$ such that $I_b = S$. To answer these questions we need to study the submodules of the $\mathbb{F}_2$-module defined in Section 4.2.

4.6.3 The structure of K^+ as a $\mathbb{F}_2$-module

As it is mentioned in Section 4.6.2, when n is an odd prime, for any $f(t) \in \mathbb{F}_2[t]$, S_f has one of 2^{s+1} submodules. Therefore we want to study these submodules to understand the security of different elements of the field against the GHS attack.

The following lemma allows us to generalize idea of I_b to submodules of K^+.

Lemma 4.6.3 *Let S be an $\mathbb{F}_2[t]-$submodule of K^+. Suppose that S can be generated by $\mathbf{b}_1, \ldots, \mathbf{b}_\ell$. Then for any $\mathbf{b} \in S$, we have $\deg(m_b(t)) \leq \deg([m_{\mathbf{b}_1}, \ldots, m_{\mathbf{b}_\ell}])$. Moreover, $I_b \subseteq \cup_{i=1}^{\ell} I_{\mathbf{b}_i}$.*

Proof. $\mathbf{b}_i \in S_{m_{\mathbf{b}_i}}$ and as it is a submodule therefore $< \mathbf{b}_i > \subseteq S_{m_{\mathbf{b}_i}}$ therefore $< \mathbf{b}_1, \ldots, \mathbf{b}_n > \subseteq \bigoplus_{i=1}^{\ell} S_{m_{\mathbf{b}_i}}$ by proposition 4.2.3 we have $< \mathbf{b}_1, \ldots, \mathbf{b}_n > \subseteq S_{[m_{\mathbf{b}_1}, \ldots, m_{\mathbf{b}_n}]}$ but $b \in < \mathbf{b}_1, \ldots, \mathbf{b}_n >$ therefore $m_b \mid [m_{\mathbf{b}_1}, \ldots, m_{\mathbf{b}_n}]$. Now the statement about I_b is clear by definition.

□

Definition 4.6.4 *Let S be a $\mathbb{F}_2[t]-$submodule of K^+ generated by $\mathbf{b}_1, \ldots, \mathbf{b}_\ell$ we define*

$$I_S := \cup_{i=1}^{\ell} I_{b_i}$$

If there is another generating set for S, we can conclude with double inclusion that the I_S for the new generating set would be the same which assures that Definition 4.6.4, is well-defined.

The submodules which have $|I_S| = 1$ are the most interesting submodules for us as they contain the least secure elements of the field. To implement an algorithm to find these submodules, we use linear algebra to analyze the Frobenius automorphism which is a linear operator over K as a k vector space. Let A be the matrix representing this Frobenius operator. By the Normal Basis Theorem [Mor96], the minimal polynomial of A is $t^n + 1$.

Algorithm 4.2 FINDING THE GENERATORS OF WEAKEST SUBMODULES

1: Compute the matrix A, the matrix of Frobenius automorphism as a linear operator of K/k.
2: Factorize $t^n + 1 = (t+1)f_1(t) \cdots f_s(t)$
3: **for all** $i \in \{1, \ldots, s\}$ **do**
4: Compute $f_i(A)$.
5: Use Gaussian Elimination to compute a basis for the kernel of $f_i(A)$.
6: Print out the basis elements, which are generators of the unsecure submodule.
7: **end for**

Although the basis elements contains a basis for a subspace of K, we know that the submodule generated by these elements also has $|I_S| = 1$ and is annihilated by $f_i(t)$. Therefore the algorithm print out generator of submodules which contains elements of $|I_S| = 1$.

We implemented this algorithm and applied it to different fields. Those results are presented in 5.3.

4.6.4 Field security evaluator

The method presented above can determine if the defining equation of an elliptic curve is weak against the GHS attack of the DLP. However, this method cannot assure us of the security of the curve, because one can presumably write another equation for the

curve with coefficients which are weak against the GHS. How can we then make sure that a chosen elliptic curve is secure?

A partial answer to above question could be as follow: if the curve is defined over a field in which all the elements are secure, then no matter how we change our equation we get a secure equation again. In language of the GHS attack, this means that all element of the field should have a minimal σ-polynomial of high degree. The weakest elements of the field are those whose minimal σ-polynomial is of degree d as defined in Lemma 4.6.1. Therefore we can evaluate the security of a field against the GHS by considering the parameter d of that field. When $d = n - 1$, we have highest expected security. If we systematically choose n an odd prime such that $\text{ord}_n(2) = n - 1$, then $m_{\mathbf{b}}(t)|(t+1)f_1(t)$ for all $\mathbf{b} \in K$. Since $m_{\mathbf{b}}(t) \neq 1$ if $\mathbf{b} \neq 0$ and $m_{\mathbf{b}}(t) \neq (t+1)$ if $\mathbf{b} \notin k$, this mean $f_1(t) \mid m_{\mathbf{b}}(t)$ for all $\mathbf{b}$. Therefore the $\deg(m_f)$ as defined in Algorithm 4.1, is either n or $n - 1$. Presumably the genus of $K(C)$ would be high in that case. This mean that for a reasonably large n, the GHS attack does not work for the curve.

According to Lemma 4.6.1, we get a fields with $d = n - 1$ whenever 2 has order $n-1$ in $(\mathbb{Z}/n\mathbb{Z})^*$ for a prime n. The question of identifying the odd primes n such that 2 has order $n-1$ is a well-known question in number theory, called the Artin primitive root conjecture. Let $a \in \mathbb{Z}$ be square free and not ± 1. Artin conjectured that there are infinity many primes n such that the order of a is $n - 1$.

Let $N_a(x) = \#\{n \leq x : a \text{ has order } n-1 \text{ in } (\mathbb{Z}/n\mathbb{Z})^*\}$. It is proved in [Hoo67] that under General Riemann Hypothesis we have

$$N_a(x) \sim C(a)\pi(x),$$

in which $C(a)$ is a positive constant and $\pi(x)$ is the number of primes up to x. Unconditional results were first obtained in [GM84], and improved afterwards. To this day, we know that there are at most 2 odd primes for which Artin conjecture fails [HB86].

We have done the numerical analysis for all extension of $\mathbb{F}_2$ whose degree is a prime less than 512 and we represent it in Section 5.3.4.

CHAPTER 5

Implementation and Computational Result

In this section we discuss implementation of the attack and the field evaluator. First we see the implementation the "Evaluator" and we bring the result of execution of the implementation for $n \in [3..512] : n$ is prime. This range of n is of cryptographic interest.

After that we will see the implementation of the general attack. Then we run our attack on some simple example and we see the property of resulting curve C and its function field extension $K(C)$.

5.1 Choosing the computer algebra system

We considered "Sage", "Kash" and "Magma" computer algebra systems for our implementations.

Sage is free software licensed under GNU General License [Fre07] and therefore open source. For that reason many mathematicians, and more specifically number theorist contributed to its development in recent years. We implement our primarily implementation of the attack in Sage. However, Sage has a serious limitation in multiple extension of function field which made the implementation of the complete attack impossible.

Kash is a computer algebra system which is free (but not a free software) and so it is not an open source software. GHS attack as it is described in [GHS02b] was originally

implemented in Kash. Therefore it was natural that we considered Kash as our second option. However, Magma, the third candidate of Computer Algebra System, contains Kash completely and one can run any capability of Kash in Magma. Magma is not free software nor free.

All the implementation is done in Magma which is the most comprehensive computer algebra language for making computation in algebraic function fields and working with curves, As it is described above.

5.2 Security evaluator

Using algorithm 4.2 we can find the unsecure subspaces of K/k. In this sense we can rank finite fields according to their security against GHS attack. This also helps us to find the unsecure subspaces which means if the coefficient of $f(z)$ in Equation 4.1 are all in one of these subspace, we would finish with a unsecure elliptic curve.

We also use the result of this attack to construct a low security curves to be attacked in Section 5.5.

5.2.1 Implementation of the security evaluator

Listing 5.1: 'Magma implementation for finding the generators of the least secure modules'

```
1  /******** WeakElementFinder.m *********
2   * This program is intended to find the elements in a finite fields
3   * for if they are used as coefficients of an elliptic curve
4   * defining polynomial the GHS attack with relatively low genus
5   * would be possible to the curve.
6   *
7   * The algorithm isfor finding unsecure elements in the field. The
8   * trick is to look at Frobenius as a linear operator and find the
9   * kernel of irreducible factors of t^n + 1.
10  */
```

```
11
12 /********Basic Definitions and Initialization********/
13 clear;
14
15 /* As defined in Thesis: example r*n = 155 => r = 15 n = 31*/
16 p := 2;
17 r := 1;
18 n := 31;
19
20 k := FiniteField(p,r);
21
22 IP<Theta> :=  IrreduciblePolynomial ( k, n );
23 kT<T> := Parent(IP);
24
25 print "Basic Parameters: p =",p," , r =",r,", n =",n;
26 print "Extending polynomial IP:", IP;
27
28 K<t> := ext<k | IP>;
29
30 /**** Computing Frobenius Matrix ****/
31 A := Transpose(Matrix(k, n, n, /
32                 [ElementToSequence((t^i)^(2)): i in [0..n-1]]));
33 print "Frobenius Matrix:",A;
34
35 /*Factoring the Frobenius Minimal Polynomial*/
36 if (IsPrime(n)) then
37   ZnZ := FiniteField(n);
38   print "Multiplicative order of 2 in Z/nZ is:", Order(2*ZnZ.1);
39 end if ;
40
41 FroMini := T^n + 1;
42 FroMiniFacts := Factorization(FroMini);
43
44 print "Factorization of", FroMini, ":";
45 print FroMiniFacts;
46
47 for CurFact in FroMiniFacts do
48 print "Current Factor:", CurFact[1];
49   CurMat := Evaluate(CurFact[1], A);
```

```
50    CurKer := Nullspace(CurMat);
51
52    print CurKer;
53
54  end for;
```

5.3 Computational result of field evaluator

To work with a computational sample data. We worked with $K \in \{\mathbb{F}_{2^5}, \mathbb{F}_{2^7}, \mathbb{F}_{2^{31}}\}$. It is worthy of mention that no matter what r is in $k = \mathbb{F}_{p^r}$ the security level of $K = \mathbb{F}_{(p^r)^n}$ just depends on n. This is because we look at decomposition of $t^n + 1$ over $\mathbb{F}_2$. Therefore in the following we assume that we treat the simplest case which is $r = 1$:

5.3.1 Case $n = 5$

In the case $n = 5$, as you can see we get $\min(m) = d = \text{ord}_5(2) = 4$ which make the extension degree $[K(C) : K] = 2^4$ which means that the field is secure:

```
Basic Parameters: p = 2 , r = 1 , n = 5
Extending polynomial IP: T^5 + T^2 + 1
Frobenius Matrix:
[1 0 0 0 1]
[0 0 0 1 0]
[0 1 0 0 1]
[0 0 0 1 1]
[0 0 1 0 0]
Factorization of T^5 + 1:
[
<T + 1, 1>,
<T^4 + T^3 + T^2 + T + 1, 1>
]
Current Factor: T + 1
Vector space of degree 5, dimension 1 over GF(2)
Echelon basis:
(1 0 0 1 0)
Current Factor: T^4 + T^3 + T^2 + T + 1
Vector space of degree 5, dimension 4 over GF(2)
Echelonized basis:
(0 1 0 0 0)
```

```
(0 0 1 0 0)
(0 0 0 1 0)
(0 0 0 0 1)
```

5.3.2 Case $n = 7$

In this case we get $\min(m) = d = \text{ord}_7(2) = 3$ with subspace of dimension 3. This means that if it happens that the coefficients of $f(x)$ in Equation 4.1 are in one of these two subspace, the extension degree of $[K(C) : K] = 2^3$ instead of expected 2^6, which much less secure.

```
Basic Parameters: p = 2, r = 1, n = 7
Extending polynomial IP: T^7 + T + 1
Frobenius Matrix:
[1 0 0 0 0 0 0]
[0 0 0 0 1 0 0]
[0 1 0 0 1 0 0]
[0 0 0 0 0 1 0]
[0 0 1 0 0 1 0]
[0 0 0 0 0 0 1]
[0 0 0 1 0 0 1]
Multiplicative order of 2 in Z/nZ is: 3
Factorization of T^7 + 1:
[
<T + 1, 1>,
<T^3 + T + 1, 1>,
<T^3 + T^2 + 1, 1>
]
Current Factor: T + 1
Vector space of degree 7, dimension 1 over GF(2)
Echelonized basis:
(1 0 0 0 0 0 0)
Current Factor: T^3 + T + 1
Vector space of degree 7, dimension 3 over GF(2)
Echelonized basis:
(0 1 0 0 1 1 0)
(0 0 1 0 1 1 1)
(0 0 0 1 1 1 1)
Current Factor: T^3 + T^2 + 1
Vector space of degree 7, dimension 3 over GF(2)
Echelonized basis:
(0 0 0 1 0 0 0)
(0 0 0 0 0 1 0)
```

```
(0 0 0 0 0 0 1)
```

5.3.3 Case $n = 31$

This case is the most interesting case among our example. one may expect that size of $[K(C) : K(z)] = 2^{31}$ usually, however, we have six, 5-dimensional unsecure subspaces. Even if it happens that the coefficient of $f(z)$ are in two different subspaces still we can attack using $[K(C) : K] = 2^{10}$ which is significantly less than 2^{30}.

```
Basic Parameters: p = 2 , r = 1 , n = 31
Extending polynomial IP: T^31 + T^3 + 1
Frobenius Matrix:
[1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1]
[0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0]
```

```
Multiplicative order of 2 in Z/nZ is: 5
Factorization of T^31 + 1
:
[
<T + 1, 1>,
<T^5 + T^2 + 1, 1>,
<T^5 + T^3 + 1, 1>,
<T^5 + T^3 + T^2 + T + 1, 1>,
<T^5 + T^4 + T^2 + T + 1, 1>,
<T^5 + T^4 + T^3 + T + 1, 1>,
<T^5 + T^4 + T^3 + T^2 + 1, 1>
]
Current Factor: T + 1
Vector space of degree 31, dimension 1 over GF(2)
Echelonized basis:
(1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
Current Factor: T^5 + T^2 + 1
Vector space of degree 31, dimension 5 over GF(2)
Echelonized basis:
(0 1 0 0 0 0 1 1 0 1 0 0 0 1 0 1 1 0 0 1 1 1 1 0 0 0 1 1 1 0 0)
(0 0 1 0 0 0 1 0 0 1 0 0 1 0 1 0 1 0 1 1 1 0 1 0 0 1 0 0 1 1 1)
(0 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 1 1 1 0 1 0 1 0 0 1 1 0)
(0 0 0 0 1 1 0 1 0 0 0 0 1 1 0 1 0 0 1 1 0 1 0 0 1 0 0 1 1 0 0)
(0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 1 0 0 0 1 0 0 1 1 1 1 0 0 1)
Current Factor: T^5 + T^3 + 1
Vector space of degree 31, dimension 5 over GF(2)
Echelonized basis:
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1)
Current Factor: T^5 + T^3 + T^2 + T + 1
Vector space of degree 31, dimension 5 over GF(2)
Echelonized basis:
(0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 0 1 1 0 0 1)
(0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 1 0 1 0 1 0)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 1 1 0 0 0 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 1 0 1)
Current Factor: T^5 + T^4 + T^2 + T + 1
Vector space of degree 31, dimension 5 over GF(2)
Echelonized basis:
(0 0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 1 1 1 1 1 1 0 1 1 1 1)
(0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0)
(0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 1 0 1 0 1 0 1 1 0 0 1 1 1 1 0)
(0 0 0 0 0 0 0 1 0 1 0 1 0 0 1 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 0)
(0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 1 1 1 0 0 1)
```

```
Current Factor: T^5 + T^4 + T^3 + T + 1
Vector space of degree 31, dimension 5 over GF(2)
Echelonized basis:
(0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 1)
(0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 0 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 1 0 1 0 0 1)
(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0 1 0 0 1 1 1)
Current Factor: T^5 + T^4 + T^3 + T^2 + 1
Vector space of degree 31, dimension 5 over GF(2)
Echelonized basis:
(0 0 0 1 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 0 1 0 1 1 0 0 0 0 0 0 0 0)
(0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 1 0 1 1 1 1 0)
(0 0 0 0 0 0 1 0 0 0 1 1 1 0 1 1 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 1)
(0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 0 0 1 0 1 1 1 1 0 0 1 1 1 0 0 1)
(0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 0 1 0 0 1 0 1 1 0 0 1 1 1 1)
```

5.3.4 Field security table

In this section we check all primes in $[3, ..., 512]$ against applicability of GHS attack. The chosen interval is particularly of cryptographic interest. In Table 5.1 one can see the minimum m that a curve can have on the specified field:

5.4 The GHS attack for general function field extensions

Using Algorithm 4.1 we can find $K(C)$ efficiently. However, this algorithm needs to know exact degree of m_f as it is defined in Section 4.3. We can run Algorithm 4.2 for each coefficients of $f(z)$ and then uses Theorem 4.2.3 to find exact degree of m. Therefore in this implementation we suppose that m is given:

5.4.1 Implementation of the attack

Listing 5.2: 'Magma implementation for the function field $K(C)$'

```
1 //FCompositonKash.m
2
3 //Initializing the program
```

Table 5.1: The security of weakest elements of extensions of $\mathbb{F}_2$ with prime degree < 1000. The smallest field extension can be achieved using Algorithm 4.1 is of degree 2^d over $K(z)$. The fields which admit non-trivial factorization of $t^n + 1$ are noted by $*$ in NT column.

n	d	NT	n	d	NT	n	d	NT	n	d	NT
3	2		101	100		229	76	*	373	372	
5	4		103	51	*	233	29	*	379	378	
7	3	*	107	106		239	119	*	383	191	*
11	10		109	36	*	241	24	*	389	388	
13	12		113	28	*	251	50	*	397	44	*
17	8	*	127	7	*	257	16	*	401	200	*
19	18		131	130		263	131	*	409	204	*
23	11	*	137	68	*	269	268		419	418	
29	28		139	138		271	135	*	421	420	
31	5	*	149	148		277	92	*	431	43	*
37	36		151	15	*	281	70	*	433	72	*
41	20	*	157	52	*	283	94	*	439	73	*
43	14	*	163	162		293	292		443	442	
47	23	*	167	83	*	307	102	*	449	224	*
53	52		173	172		311	155	*	457	76	*
59	58		179	178		313	156	*	461	460	
61	60		181	180		317	316		463	231	*
67	66		191	95	*	331	30	*	467	466	
71	35	*	193	96	*	337	21	*	479	239	*
73	9	*	197	196		347	346		487	243	*
79	39	*	199	99	*	349	348		491	490	
83	82		211	210		353	88	*	499	166	*
89	11	*	223	37	*	359	179	*	503	251	*
97	48	*	227	226		367	183	*	509	508	

```
4  clear;
5  // To whether check if the resulting curve is hyperelliptic or not
6  HyperellipticCheck := true;
7
8  //Defining the fields
9  p := 2;
10 r := 1;
11 n := 3;
12
13 m_fDegree := 2;
14
15 //Constructing the Fields and the Homomorphisms between them
16 k := FiniteField(p,r);
17 kx<x> := RationalFunctionField(k);
18 kxT<T> := PolynomialRing(kx);
19
20 IP<Theta> := IrreduciblePolynomial ( k, n );
21 kT<Theta> := Parent(IP);
22 print "Basic Parameters: p =",p," , r =",r,", n =",n;
23
24 EmbkTkxT := hom< kT-> kxT | T>;
25 TIP := EmbkTkxT(IP);
26
27 print "Extending K/k with", TIP;
28 K<t> := ext<k | IP>;
29 Kx<x> := RationalFunctionField(K);
30
31 kxbyb<y> := PolynomialRing(kx);
32 KxbYb<Y> := PolynomialRing(Kx);
33
34 kxy<yp,xp> := PolynomialRing(k, 2);
35 KbXYb<YP,XP> := PolynomialRing(K,2);
36
37 //Defining the curve
38 //No irreducibility check!
39
40 AttackedCurves := {Y^2 + x*Y + t*x^3 + t/*,Y^2 + Y + 1/x + t*x,
41                    Y^2 + x*Y + x^3 + 1*/};
42
```

```
43 for E in AttackedCurves do
44   print "";
45   print "********";
46
47   print "The Original Curve under attack, E is:";
48   print E;
49
50   mCounter := 0; //How many times we extend our basic curve's f.f.
51
52   ExtendingCurve := E;
53   FR := [KxbYb];
54   F := [FunctionField(ExtendingCurve : Check := false)];
55
56   //Extending Frobenius
57   sigmaK := hom<K->F[1]| t^(p^r)>;
58   sigmaT := hom<Kx->F[1]|sigmaK, Kx.1>;
59   sigmaE := [*hom<F[1]->F[1]| sigmaT, F[1].1>*];
60
61   for nCounter in [1..n-1] do
62
63     CurRing<YP> := PolynomialRing(F[mCounter+1]: Global := false);
64     Append(~FR, CurRing);
65
66     AssignNames(~FR[nCounter+1], ["w"*IntegerToString(nCounter)]);
67
68     sigmaEP := hom<FR[nCounter+1]->FR[nCounter+1] | sigmaE[mCounter+1],
69                 FR[nCounter+1].1>;
70     ExtendingCurve := sigmaEP(ExtendingCurve);
71
72     print "Applying sigma^", nCounter, " results in equation ";
73     print ExtendingCurve;
74
75     mCounter := mCounter + 1;
76     Append(~F, FunctionField(ExtendingCurve : Check := false));
77
78     //Extending Frobenius
79     CurK := hom<K->F[mCounter+1]| sigmaK(t)>;
80     CurT := hom<Kx->F[mCounter+1]| CurK, Kx.1>;
81     for i in [1..mCounter] do
```

```
82        CurT := hom<F[i]->F[mCounter+1]|CurT, F[i].1>;
83      end for;
84      Append(~sigmaE, hom<F[mCounter+1]->F[mCounter+1]| CurT,
85              F[mCounter+1].1>);
86
87      if (mCounter eq m_fDegree - 1) then
88          break;
89      end if;
90
91    end for;
92
93    print "Compositon degree [K(C):K(E)] = 2^", mCounter;
94
95    KC := AbsoluteFunctionField(F[mCounter+1]);
96    PC<yp,xp> := DefiningPolynomial(KC);
97
98    print "The genus of the function field K(C) is: ",  Genus(KC);
99
100   //Map PC to a univariate-looking Ring!
101   CanKbXYbKxbYb := hom<KbXYb -> KxbYb | Y, x>;
102   C := CanKbXYbKxbYb(PC);
103   print "C is", C;
104
105   //Redefining KC directly with the defining equation: just for
106   //efficiency.
107   KC<Y> := FunctionField(C : Check := false);
108   KCbWb<W> := PolynomialRing(KC);
109
110   FroKKxbYb := hom<K->FR[1]| sigmaK(t)>;
111   FroKxKxbYb := hom<Kx->FR[1]| FroKKxbYb, Kx.1>;
112   FroKxbYbKxbYb := hom<FR[1]->FR[1]| FroKxKxbYb, FR[1].1>;
113
114   ImgC := FR[1] ! C;
115   ProdC := C;
116   for i in [1..n-1] do
117     ImgC := FroKxbYbKxbYb(ImgC);
118     ProdC *:= ImgC;
119
120   end for;
```

```
121
122     print "Prod_{i = 1..n}Sigma(C(Y)) = ";
123     print ProdC;
124
125     print "Complete Factorization of above Product over KC[W]:";
126     print Factorization(KCbWb ! ProdC);
127
128     if (HyperellipticCheck) then
129       P2<Z,Y,x> := ProjectiveSpace(K,2);
130
131       AI := ideal<KbXYb|[PC]>;
132       KXYZ<Z,Y,x>, hm := Homogenization(AI);
133       PPC := BasisElement(hm(AI),1);
134
135       C := Curve(P2,PPC);
136       hpicity, HC,hmp := IsHyperelliptic(C);
137       if (hpicity) then
138         print "C is hyperelliptic. C is ", HC;
139
140       else
141         print "C is NOT hyperelliptic.";
142
143       end if;
144
145     end if;
146
147 end for;
```

5.5 Computational result of running the attack

In this section investigate some individual examples of curves over different field. For this reason we chose $n \in 3, 7, 31$.

5.5.1 Case n = 3

In this section we attack the curve $Y^2+xY = x^3+tx^2+1$ such that $k = \mathbb{F}_2$ is extended to $K(t)$ such that t^3+t+1. The minimal polynomial of Frobenius is factorized as $t^3+1 = (t+1)(t^2+t+1)$. Now $f(x) = X^3+tx^2+1$. So we have $m_1 = t+1$ and $m_t(t) = t^2+t+1$. This mean $m_f = [t+1, t^2+t+1] = t^3+1$. Therefore $\deg(m_f) = 3$. By running the attack we get following result:

```
Basic Parameters: p = 2 , r = 1 , n = 3
Extending K/k with T^3 + T + 1

---------
The Original Curve under attack E is:
Y^2 + x*Y + x^3 + t
Applying sigma^ 1 results in equation
w1^2 + x*w1 + x^3 + t^2
Applying sigma^ 2 results in equation
w2^2 + x*w2 + x^3 + t^4
Compositon degree [K(C):K(E)] = 2^ 2
The genus of the function field K(C) is: 3
C is Y^8 + (t^2*x^7 + t^4*x^6 + t^6*x^5 + t*x^4)*Y^4 + (t^3*x^7 +
t^5*x^6)*Y^2 + (x^14 + t^2*x^13 + t^4*x^12 + t^6*x^11 + t*x^10 +
t^3*x^9 + t^5*x^8)*Y + t^3*x^20 + t^3*x^18 + t*x^17 + t^6*x^16 +
t*x^15 + t^5*x^14 + t^4*x^13 + x^12 + t^2*x^11 + t^3*x^10 + t^3*x^9
+ x^8 + t^2*x^7 + t^6*x^6 + t*x^5 + t^4*x^4 + t^4
Prod_{i = 1..n}Sigma(C(Y)) =
Y^24 + x^5*Y^20 + (x^7 + x^6)*Y^18 + (x^14 + x^11 + x^9 + x^8)*Y^17
+ (x^20 + x^18 + x^16 + x^13 + x^6)*Y^16 + (x^14 + x^12 + x^11 +
x^10)*Y^14 + (x^16 + x^14 + x^13 + x^12)*Y^13 + (x^27 + x^26 + x^22
+ x^21 + x^20 + x^18 + x^15 + x^13 + x^12 + x^11 + x^9 + x^8 +
x^7 + x^5 + x^4)*Y^12 + (x^20 + x^18 + x^17 + x^16)*Y^11 + (x^28 +
x^25 + x^22 + x^21 + x^20 + x^17 + x^15 + x^14 + x^12 + x^7)*Y^10
+ (x^33 + x^32 + x^28 + x^26 + x^24 + x^23 + x^20 + x^18 + x^16 +
x^15 + x^13 + x^11 + x^10 + x^9)*Y^9 + (x^36 + x^34 + x^33 + x^32
+ x^31 + x^28 + x^27 + x^26 + x^23 + x^22 + x^17 + x^16 + x^15 +
x^13 + x^11 + x^9 + x^7 + x^6 + x^5 + 1)*Y^8 + (x^28 + x^26 + x^25
+ x^24)*Y^7 + (x^32 + x^31 + x^30 + x^28 + x^26 + x^25 + x^24 + x^21
+ x^19 + x^17 + x^13 + x^10)*Y^6 + (x^41 + x^40 + x^36 + x^33 +
x^32 + x^25 + x^22 + x^18 + x^15 + x^12)*Y^5 + (x^47 + x^45 + x^43
+ x^42 + x^40 + x^37 + x^33 + x^32 + x^27 + x^26 + x^25 + x^22 +
x^21 + x^19 + x^17 + x^16 + x^15 + x^14 + x^12 + x^7 + x^6)*Y^4
+ (x^42 + x^39 + x^38 + x^35 + x^34 + x^33 + x^30 + x^29 + x^28 +
x^27 + x^26 + x^25 + x^23 + x^21 + x^19 + x^16)*Y^3 + (x^48 + x^45
+ x^44 + x^43 + x^42 + x^41 + x^40 + x^39 + x^38 + x^37 + x^36 +
```

```
x^35 + x^34 + x^32 + x^30 + x^27 + x^26 + x^25 + x^24 + x^22 + x^21
+ x^20 + x^17 + x^14 + x^13 + x^11 + x^7)*Y^2 + (x^53 + x^51 +
x^50 + x^40 + x^39 + x^38 + x^36 + x^35 + x^34 + x^33 + x^32 + x^30
+ x^27 + x^25 + x^24 + x^20 + x^18 + x^16 + x^15 + x^14 + x^12 +
x^9)*Y + x^60 + x^58 + x^57 + x^55 + x^54 + x^52 + x^49 + x^48
+ x^47 + x^46 + x^45 + x^41 + x^39 + x^38 + x^35 + x^33 + x^30
 + x^29 + x^28 + x^26 + x^23 + x^22 + x^21 + x^20 + x^19 + x^18
 + x^17 + x^15 + x^14 + x^13 + x^12 + x^9 + x^7 + x^4 + 1

Complete Factorization of above Product over KC[W]:
[
<W + Y, 1>,
<W + Y + x^2, 1>,
<W + Y + t*x^2 + t^4*x, 1>,
<W + Y + t^2*x^2 + t^6*x, 1>,
<W + Y + t^3*x^2 + t^4*x, 1>,
<W + Y + t^4*x^2 + t^3*x, 1>,
<W + Y + t^5*x^2 + t^3*x, 1>,
<W + Y + t^6*x^2 + t^6*x, 1>,
<W + (t*x^3 + t^6*x^2 + t^6)/(x^9 + t^5*x^8 + t^2*x^7 + t*x^6 +
t^2*x^5 + t^2*x^4)*Y^4 + (t^3*x^5 + t^3*x^4 + x^3 + t^3*x^2 + t^5*x
+ t^3)/(x^7 + t^5*x^6 + t^2*x^5 + t*x^4 + t^2*x^3 + t^2*x^2)*Y^2
+ (t*x^6 + t^5*x^5 + t^2*x^4 + t^5*x^3 + t^4*x^2 + t^3*x +
t^5)/(x^6 + t^5*x^5 + t^2*x^4 + t*x^3 + t^2*x^2 + t^2*x)*Y +
(t^6*x^12 + x^11 + t*x^10 + t^5*x^9 + t^6*x^8 + t^6*x^7 + t^3*x^6 +
t^3*x^5 + x^4 + x^3 + x^2 + t^5*x + t^3)/(x^8 + t^2*x^6 + t^3*x^5 +
t^4*x^4), 1>,

⋮

<W + (t^3*x^3 + t^3*x^2 + t^6)/(x^9 + x^8 + t^3*x^7 + x^6 + t^5*x^5
+ t^6*x^4)*Y^4 + (t*x^5 + x^4 + t*x^3 + x^2 + t^5*x + t^3)/(x^7
+ x^6 + t^3*x^5 + x^4 + t^5*x^3 + t^6*x^2)*Y^2 + (t*x^6 + t^2*x^5 +
t*x^4 + t^3*x^3 + x + t^5)/(x^6 + x^5 + t^3*x^4 + x^3 +
t^5*x^2 + t^6*x)*Y + (t*x^12 + t^2*x^11 + t^6*x^10 + t^3*x^9 + x^8
+ x^7 + t^4*x^6 + t^6*x^5 + x^4 + t^2*x^3 + t^3*x^2 + t^5*x +
t^3)/(x^8 + t^4*x^7 + t^5*x^6 + t*x^5 + t*x^4), 1>
]

C is hyperelliptic.
```

The product of σ images of C has coefficient in $k(x)$ and is completely factored in $K(C)$. Therefore, $K(C)$ is splitting field of $\prod_{i=0}^{n-1} \sigma(C)$ over $k(x)$ and hence is Galois over it. Therefore it is meaningfull to talk about $K(C)^{\sigma} = k(C)$. The equation of the

curve C is defined over K. However in [GHS02b] and [Hes04], there are algorithm to explicitly compute C such that C is defined over k as well.

As $\deg(m_f) = 3$, the equation of the curve is considered secure against GHS attack. However, there is no guarantee that one can not write $K(E)$ with another equation whose $\deg(m_f) = 2$. But 2 is the absolute lower bound. It is important to notice that the resulting curve is hyperelliptic.

5.5.2 Case n = 7

In this case we attacked Curve $Y^2 + xY + tx^3 + t$. $K = k(t)$ such that $t^7 + t + 1$. The factorization of the minimal polynomial of the Frobenius is not trivial as discussed in Section 5.3.2. Now $f(x) := tx^3 + t$. So the only coefficient is t, so $m_f(t) = m_t(t)$. Due to the result of Section 5.3.2, we know that $m_t(t) = t^3 + t + 1$ and that means that $\deg(m_t(t)) = \deg(m_f(t)) = 3$:

```
Basic Parameters: p = 2 , r = 1 , n = 7
Extending K/k with T^7 + T + 1

---------
The Original Curve under attack E is:
Y^2 + x*Y + t*x^3 + t
Applying sigma^ 1 results in equation
w1^2 + x*w1 + t^2*x^3 + t^2
Applying sigma^ 2 results in equation
w2^2 + x*w2 + t^4*x^3 + t^4
Compositon degree [K(C):K(E)] = 2^ 2
The genus of the function field K(C) is: 7
C is Y^8 + (t^124*x^8 + t^23*x^7 + t^48*x^6 + t^71*x^5 + t^62*x^4)*Y^4
 + (t^5*x^12 + t^84*x^11 + t^61*x^10 + t^42*x^8 + t^48*x^7 +
t^116*x^6)*Y^2 + (t^111*x^14 + t^103*x^13 + t^71*x^12 + t^113*x^11 +
t^91*x^10 + t^73*x^9 + t^113*x^8 + t^123*x^7)*Y + t^98*x^20 +
t^4*x^18 + t^121*x^17 + t^90*x^16 + t^59*x^15 + t^74*x^14 + t^75*x^13
 + t^125*x^12 + t^102*x^11 + t^126*x^10 + t^100*x^9 + t^34*x^7
+ t^85*x^6 + t^31*x^5 + t^120*x^4 + t^47
Prod_{i = 1..n}Sigma(C(Y)) =
Y^56 + (x^8 + x^4)*Y^52 + (x^11 + x^10 + x^8)*Y^50 + (x^14 + x^13 +
x^9 + x^7)*Y^49 + (x^17 + x^16 + x^14 + x^11 + x^9 + x^6 + x^5 +
1)*Y^48 + (x^19 + x^18 + x^16 + x^15 + x^14 + x^10)*Y^46 + (x^21 +
```

```
x^20 + x^19 + x^18 + x^15 + x^13 + x^12 + x^11)*Y^45 + (x^28 +
x^27 + x^26 + x^25 + x^23 + x^21 + x^20 + x^19 + x^16 + x^15 + x^14
+ x^13 + x^12 + x^11 + x^10 + x^9 + x^8 + x^7 + x^6 + x^4)*Y^44 +
$\vdots$
x^53 + x^52 + x^51 + x^50 + x^48 + x^47 + x^44 + x^43 + x^40 + x^37
+ x^35 + x^34 + x^24 + x^19 + x^17 + x^15 + x^14 + x^13 + x^12 +
x^11 + x^10 + x^8 + x^6 + x^5 + x^4 + 1
Complete Factorization of above Product over KC[W]:
[
<W + Y, 1>,
<W + Y + t^5*x^2 + t^77*x, 1>,
<W + Y + t^8*x^2 + t^125*x, 1>,
<W + Y + t^68*x^2 + t^69*x, 1>,
<W + Y + t^81*x^2 + t^102*x, 1>,
<W + Y + t^86*x^2 + t^57*x, 1>,
<W + Y + t^121*x^2 + t^55*x, 1>,
<W + Y + t^123*x^2 + t^19*x, 1>,
<W + (t^4*x^4 + t^57*x^3 + t^126*x^2 + t^27*x + t^91)/(x^10 +
t^119*x^9 + t^87*x^8 + t^2*x^7 + t^107*x^6 + t^89*x^5 + t^2*x^4 +
t^12*x^3)*Y^4 + (t^91*x^6 + t^34*x^5 + x^4 + t^35*x^2 + t^66*x +
t^75)/(x^8 + t^119*x^7 + t^87*x^6 + t^2*x^5 + t^107*x^4 +
t^89*x^3 + t^2*x^2 + t^12*x)*Y^2 + (t^99*x^7 + t^85*x^6 + t^103*x^5
+ t^48*x^4 + t^2*x^3 + t^78*x^2 + t^54*x + t^113)/(x^7 +
t^119*x^6 + t^87*x^5 + t^2*x^4 + t^107*x^3 + t^89*x^2 + t^2*x +
 t^12)*Y + (t^53*x^14 + t^56*x^13 + t^25*x^12 + t^99*x^11 +
t^83*x^10 + t^22*x^9 + t^78*x^8 + t^39*x^7 + t^110*x^6 + t^109*x^5
+ t^64*x^4 + t^96*x^3 + t^9*x^2 + t^114*x + t^51)/(x^10 +
t^119*x^9 + t^87*x^8 + t^2*x^7 + t^107*x^6 + t^89*x^5 + t^2*x^4 +
t^12*x^3), 1>,


⋮


<W + (t^106*x^4 + t^38*x^3 + t^35*x^2 + t^61*x + t^92)/(x^10 +
t^119*x^9 + t^87*x^8 + t^2*x^7 + t^107*x^6 + t^89*x^5 + t^2*x^4 +
t^12*x^3)*Y^4 + (t^21*x^6 + t^123*x^5 + t^84*x^4 + t^114*x^2 +
t^38*x + t^92)/(x^8 + t^119*x^7 + t^87*x^6 + t^2*x^5 + t^107*x^4
+ t^89*x^3 + t^2*x^2 + t^12*x)*Y^2 + (t^40*x^7 + t^126*x^6 +
t^50*x^5 + t^62*x^4 + t^118*x^3 + t^111*x^2 + t^118*x)/(x^7 +
t^119*x^6 + t^87*x^5 + t^2*x^4 + t^107*x^3 + t^89*x^2 + t^2*x
+ t^12)*Y + (t^28*x^14 + t^22*x^13 + t^83*x^12 + t^124*x^11 +
t^50*x^10 + t^26*x^9 + t^67*x^8 + t^57*x^7 + t^121*x^6 + t^58*x^5
+ t^59*x^4 + t^29*x^3 + t^119*x^2 + t^21*x + t^52)/(x^10 +
t^119*x^9 + t^87*x^8 + t^2*x^7 + t^107*x^6 + t^89*x^5 + t^2*x^4 +
t^12*x^3), 1>
]
C is NOT hyperelliptic.
```

The resulting curve is of degree 2^3 instead of secure degree 2^7 and therefore this curve is not secure for cryptographic purpose. $K(C)$ is not hyperelliptic which means that the implementation of index calculus attack against $k(C)$ is not as efficient as the hyperelliptic case.

5.5.3 Case n = 31

We attack the curve $Y^2 + xY + x^3 + t$ defined over $K = k(t)$ such that $t^31 + t^2 + 1 = 0$ so $n = 31$. $f(x) = x^3 + t$. The coefficients are $1, t$ and $m_f(t) = [m_1, m_t]$. $m_1 = t + 1$ as Frobenius fix 1. $m_t(t) == t^5 + t^2 + 1$ using result from Section 5.3.3. Therefoe $m_f(t) == [t + 1, t^5 + t^2 + 1] = (t + 1)(t^5 + t^2 + 1)$ therefore $\deg(m_f) = 6$. Knowing these facts we can run the attack:

```
Basic Parameters: p = 2 , r = 1 , n = 31
Extending K/k with T^31 + T^3 + 1

---------
The Original Curve under attack E is:
Y^2 + x*Y + x^3 + t
Applying sigma^ 1 results in equation
w1^2 + x*w1 + x^3 + t^2
Applying sigma^ 2 results in equation
w2^2 + x*w2 + x^3 + t^4
Applying sigma^ 3 results in equation
w3^2 + x*w3 + x^3 + t^8
Applying sigma^ 4 results in equation
w4^2 + x*w4 + x^3 + t^16
Applying sigma^ 5 results in equation
w5^2 + x*w5 + x^3 + t^4 + t
Compositon degree [K(C):K(E)] = 2^ 5
The genus of the function field K(C) is: 31
C is Y^64 + Y\^{}32((t^30 + t^29 + t^28 + t^27 + t^26 + t^22 + t^21
+ t^20 + t^18 + t^17 + t^15 + t^12 + t^7 + t + 1)*x^64 + (t^29 +
t^26 + t^25 + t^22 + t^21 +

⋮

(t^27 + t^26 + t^24 + t^22 + t^17 + t^16 + t^15 + t^9 + t^8 +
t^7 + t^6 + t^5 + t^2)*x^33 + (t^29 + t^25 + t^24 + t^23 + t^21 +
t^18 + t^14 + t^12 + t^10 + t^7 + t^6 + t^5 + t^4 + t^2 + t + 1)*x^32
```

```
 + t^27 + t^24 + t^23 + t^22 + t^20 + t^19 + t^16 + t^13 + t^7 +
t^6 + t^4 + t
```

The result is a curve of degree $64 = 2^6$, However a secure curve should result in a curve of degree $2^{30} = 1073741824$. Therefore this curve is significantly less secure than it is expected to be.

CHAPTER 6

Conclusion and Further Work

A decade has been passed since introduction of the idea of the GHS method. During this decade, the attack was improved from a mere idea to a standard technique to attack curves defined over composite fields. The progress of the GHS attack also influenced other area of cryptography and algorithmic number theory such as hyperelliptic curve cryptography, algorithmic study of class group of algebraic curves and function fields.

The purpose of this work was to study the GHS attack in more general form, both from theoretical and practical view. We presented the basic foundation of GHS attack, namely solving the discrete logarithm over jacobian of a general curve. Then we described the theoretical aspect of GHS attack to an elliptic curve defined over a finite field of arbitrary characteristics.

We then studied more practical aspects of the GHS attack. In this stage, we restricted ourselves to even characteristic and we considered more general quadratic extensions than [Hes04]. We showed that the same method of σ-polynomial that is used in [Hes04] can be used to analyze the degree of the extension $K(C)$ and we developed an upper bound for the genus of $K(C)$. We also studied the structure of K as a $\mathbb{F}_2[t]-$module and we gave an algorithm to find the least secure elements of a composite field for the GHS attack.

Finally, we brought the implementation of the main algorithms described in this paper along with the example of computational results of these algorithms.

The fundamental idea behind the GHS attack, namely mapping the DLP over the

jacobian of a curve to an easier instance of DLP over jacobian of another curve using the relationship between their function fields, is vast and has the potential of deeper studies.

By studying the inverse image of the factor base of the $Cl^0(C_0)$, it could be possible to find a method to build a factor base for elliptic curve DLP. Similar approach using Weil Descent idea (the geometric parallel of the GHS) is studied in [Gau04].

Because of the GHS attack, it is now standard to avoid composite finite fields for elliptic curve cryptography. It would be interesting to extend some of ideas of the GHS to attack other cases of the DLP. True subfield curves used over a large prime extensions, such as Koblitz curves [HMV03] are still in use and attract considerable interest due to their performance. The method described in this work is not applicable to these curves. However, considering other isogenous curve may lead to an efficient attack against these curves. Partial work on this issue has been done in [DS03].

Optimal Extension Fields (OEFs) [HMV03], are extensions of a prime fields whose characteristics occupies all most one machine words (e.g. 64 bits). These fields are also known to have efficient finite field operation algorithm. The study of the GHS attack against curves defined over these fields is also an interesting problem.

The above examples are a few open problems centered on the study of the GHS attack. The final word is that the GHS attack opened the flood-gates for the use function field arithmetic in cryptography.

Bibliography

[ADH94] Leonard M. Adleman, Jonathan DeMarrais, and Ming-Deh A. Huang, *A subexponential algorithm for discrete logarithms over the rational subgroup of the jacobians of large genus hyperelliptic curves over finite fields*, ANTS-I: Proceedings of the First International Symposium on Algorithmic Number Theory (London, UK), Springer-Verlag, 1994, pp. 28–40.

[AMV93] Gordon B. Agnew, Ronald C. Mullin, and Scott A. Vanstone, *An implementation of elliptic curve cryptosystems over* $f_{2^{155}}$, IEEE Journal on Selected Areas in Communications **11** (1993), no. 5.

[BSS05] Ian Blake, Gadiel Seroussi, and Nigel Smart (eds.), *Advances in elliptic curve cryptography (london mathematical society lecture note series)*, Cambridge University Press, New York, NY, USA, 2005.

[CFA06] Henri Cohen, Gerhard Frey, and Roberto Avanzi, *Handbook of elliptic and hyperelliptic curve cryptography: Theory and practice*, CRC Press, Inc., 2006.

[Che51] Claude C. Chevalley, *Introduction to the theory of algebraic function of one variable*, American Mathematical Society, 1951.

[DD85] Jean Dieudonne and Suzanne Dieudonne, *History of algebraic geometry: An outline of the history and development of algebraic geometry*, CRC Press, Inc., 1985.

[Dec05] Isabelle Dechene, *Generalized jacobians in cryptography*, Ph.D. thesis, McGill University, 2005.

[DH76] Whitfield Diffie and Martin E. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory **IT-22** (1976), no. 6, 644–654.

[Die03] Claus Diem, *The GHS attack in odd characteristic*, Journal of the Ramanujan Mathematical Society **18** (2003), 1–32, URL: `http://www.math.uni-leipzig.de/ diem/preprints`.

[Die08] ———, *On arithmetic and the discrete logarithm problem in class groups of curves*, 2008, Habilitation, University of Leipzig.

[DS03] Claus Diem and Jasper Scholten, *Cover attack*, A report for the AREHCC project, preprint., October 2003.

[EGL07] Taraneh Eghlidos, Amir Ghadermarzi, and Ahmad Lavasani, *Evaluating the security of the elliptic curves based on efficient composite binary fields against weil decent attack*, Tech. report, February 2007.

[ELS06] Taraneh Eghlidos, Ahmad Lavasani, and Laleh Samarbakhsh, *Study of the methods of finding secure and efficient elliptic curves for cryptographic applications*, Tech. report, Electronics Research Center, Sharif University of Technology, February 2006.

[FR94] Gerhard Frey and Hans-Georg Rück, *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Math. Comput. **62** (1994), no. 206, 865–874.

[Fre98] Gerhard Frey, *How to disguise an elliptic curve*, Talk at ECC '98, Waterloo, 1998.

[Fre02] Free Software Foundation Inc., *The gnu free documentation license - version 1.2*, November 2002, http://www.gnu.org/licenses/fdl.html.

[Fre07] Free Software Foundation Inc., *Gnu general public license - version 3.0*, June 2007, http://www.gnu.org/licenses/gpl-3.0.html.

[Gau04] Pierrick Gaudry, *Index calculus for abelian varieties and the elliptic curve discrete logarithm problem*, Cryptology ePrint Archive (2004), Cryptology ePrint Archive: Report 2004/073.

[GHS02a] Steven D. Galbraith, Florian Hess, and Nigel P. Smart, *Extending the GHS weil descent attack*, Theory and Application of Cryptographic Techniques, 2002, pp. 29–44.

[GHS02b] Pierrick Gaudry, Florian Hess, and Nigel P. Smart, *Constructive and destructive facets of Weil descent on elliptic curves*, Journal of Cryptology: the journal of the International Association for Cryptologic Research **15** (2002), no. 1, 19–46.

[GM84] Rajiv Gupta and Ram Murty, *A remark on artin's conjecture*, Invent. Math. **78** (1984), no. 1.

[GP97] Jorge Guajardo and Christof Paar, *Efficient algorithms for elliptic curve cryptosystems*, Lecture Notes in Computer Science **1294** (1997).

[GS99] Steven D. Galbraith and Nigel P. Smart, *A cryptographic application of weil descent*, Proceedings of the 7th IMA International Conference on Cryptography and Coding (London, UK), Springer-Verlag, 1999, pp. 191–200.

[Har97] Robin Hartshorne, *Algebraic geometry*, Graduate Texts in Mathematics, Springer-Verlag, 1997.

[HB86] Roger Heath-Brown, *Artin's conjecture for primitive roots*, Quart. J. Math. Oxford Ser. **(2) 37** (1986), no. 145, 27–38.

[Hes04] Florian Hess, *Generalising the ghs attack on the elliptic curve discrete logarithm problem*, LMS Journal of Computation and Mathematics **7** (2004).

[HMV03] Darrel Hankerson, Alfred J. Menezes, and Scott A. Vanstone, *Guide to elliptic curve cryptography*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.

[Hoo67] Christopher Hooley, *On artin's conjecture*, J. Rein Angew Math. **225** (1967).

[Hun03] Thomas Hungerford, *Algebra*, Graduate Texts in Mathematics, Springer-Verlag, 2003.

[JMS01] Michael Jacobson, Alfred J. Menezes, and Andreas Stein, *Solving elliptic curve discrete logarithm problems using weil descent*, Cryptology ePrint Archive, Report 2001/041, 2001, http://eprint.iacr.org/.

[Kob87] Neal Koblitz, *Elliptic curve cryptosystems*, Mathematics of Computation **48** (1987), 203–209.

[Kob89] ———, *Hyperelliptic cryptosystems*, J. Cryptol. **1** (1989), no. 3, 139–150.

[Kob07] Neal Koblitz, *The uneasy relationship between mathematics and cryptography*, Notices of the AMS **54** (2007), no. 8, 973–979.

[Lor96] Dino Lorenzini, *An invitation to arithmetic geometry*, Graduate Studies in Mathematics, American Mathematical Society, 1996.

[Mil86] Victor S. Miller, *Uses of elliptic curves in cryptography*, 1986, pp. 417–426.

[MMT01] Markus Maurer, Alfred J. Menezes, and Edlyn Teske, *Analysis of the GHS Weil descent attack on the ECDLP over characteristic two finite fields of composite degree*, Lecture Notes in Computer Science **2247** (2001), 195–??

[Mor96] Patrick Morandi, *Field and galois theory*, Graduate Texts in Mathematics, vol. 167, Springer-Verlag, 1996.

[MQ01] Alfred J. Menezes and Minghua Qu, *Analysis of the weil descent attack of gaudry, hess and smart*, CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology (London, UK), Springer-Verlag, 2001, pp. 308–318.

[MT06] Alfred J. Menezes and Edlyn Teske, *Cryptographic implications of hess' generalized ghs attack*, Appl. Algebra Eng., Commun. Comput. **16** (2006), no. 6, 439–460.

[MTW04] Alfred J. Menezes, Edlyn Teske, and Annegret Weng, *Weak fields for ecc*, Topics in Cryptology, ch. Weak Fields for ECC, Springer-Verlag, 2004.

[MVO91] Alfred J. Menezes, Scott A. Vanstone, and Tatsuaki Okamoto, *Reducing elliptic curve logarithms to logarithms in a finite field*, STOC '91: Proceedings of the twenty-third annual ACM symposium on Theory of computing (New York, NY, USA), ACM, 1991, pp. 80–89.

[MVO96] Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot, *Handbook of applied cryptography*, CRC Press, Inc., Boca Raton, FL, USA, 1996.

[MWZ96] Alfred J. Menezes, Yi-Hong Wu, and Robert J. Zuccherato, *An elementary introduction to hyperelliptic curve*, Tech. report, University of Waterloo, 1996.

[Orm98] H. Orman, *The oakley key determination protocol-rfc2412*, Tech. report, The Internet Society, 1998.

[PFR98] Christof Paar, Peter Fleischmann, and Peter Roelse, *Efficient multiplier architectures for galois fields GF(2 4n)*, IEEE Transactions on Computers **47** (1998), no. 2, 162–170.

[PFSR99] Christof Paar, Peter Fleischmann, and Pedro Soria-Rodriguez, *Fast arithmetic for public-key algorithms in galois fields with composite exponents*, IEEE Transactions on Computers **48** (1999), no. 10, 1025–1034.

[PSR97] Christof Paar and Pedro Soria-Rodriguez, *Fast arithmetic architectures for public-key algorithms over Galois fields* GF($(2^n)^m$), Lecture Notes in Computer Science **1233** (1997).

[Ros02] Michael Rosen, *Number theory in function fields*, Springer-Verlag, 2002.

[RS03] Karl Rubin and Alice Silverberg, *Torus-based cryptography*, In Advances in Cryptology (CRYPTO 2003), Springer LNCS, Springer-Verlag, 2003.

[Sha48] C. E. Shannon, *A mathematical theory of communication*, Bell system technical journal **27** (1948).

[Sil94] Joseph H. Silverman, *The arithmetic of elliptic curves*, Graduate Texts in Mathematics, Springer-Verlag, 1994.

[Sil00] ———, *The xedni calculus and the elliptic curve discrete logarithm problem*, Designs, Codes and Cryptography **20** (2000), no. 1, 5–40.

[Sma99] Nigel P. Smart, *The discrete logarithm problem on elliptic curves of trace one*, Journal of Cryptology: the journal of the International Association for Cryptologic Research **12** (1999), no. 3, 193–196.

[Sma01] ———, *How secure are elliptic curves over composite extension fields?*, Lecture Notes in Computer Science **2045** (2001).

[Sti93] H. Stichtenoth, *Algebraic function fields and codes*, Springer-Verlag, 1993.

[The03] Nicolas Theriault, *Weil descent attack for kummer extensions*, Journal of the Ramanujan Mathematical Society **18** (2003), no. 3.

[vdW83] Bartel van der Waerden, *Geometry and algebra in ancient civilizations*, Springer-Verlag, 1983.