Extending the GLS endomorphism to speed up GHS Weil descent using Magma

Jesús-Javier Chi-Domínguez, Francisco Rodríguez-Henríquez, Benjamin Smith

Extending the GLS endomorphism to speed up GHS Weil descent using Magma

Jesús-Javier Chi-Domínguez[b,a,*], Francisco Rodríguez-Henríquez[b,a,1], Benjamin Smith[c,2]

[a] *Technology Innovation Institute (TII), Abu Dhabi, United Arab Emirates*
[b] *Computer Science Department, Center for Research and Advanced Studies of the National Polytechnic Institute of Mexico (Cinvestav - IPN), Mexico City, Mexico*
[c] *Inria and Laboratoire d'Informatique de l'École polytechnique (LIX), Institut Polytechnique de Paris, Palaiseau, France*

Abstract

Let $q = 2^n$, and let $\mathcal{E}/\mathbb{F}_{q^\ell}$ be a generalized Galbraith–Lin–Scott (GLS) binary curve, with $\ell \geq 2$ and $(\ell, n) = 1$. We show that the GLS endomorphism on $\mathcal{E}/\mathbb{F}_{q^\ell}$ induces an efficient endomorphism on the Jacobian $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ of the genus-g hyperelliptic curve $\mathcal{H}$ corresponding to the image of the GHS Weil-descent attack applied to $\mathcal{E}/\mathbb{F}_{q^\ell}$, and that this endomorphism yields a factor-n speedup when using standard index-calculus procedures for solving the Discrete Logarithm Problem (DLP) on $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$. Our analysis is backed up by the explicit computation of a discrete logarithm defined on a prime-order subgroup of a GLS elliptic curve over the field $\mathbb{F}_{2^{5\cdot 31}}$. A Magma implementation of our algorithm finds the aforementioned discrete logarithm in about $1,035$ CPU-days.

Keywords: GHS Weil descent, extended GLS endomorphism, index-calculus algorithm

1. Introduction

Let $\mathbb{G}$ be an additively-written cyclic group of order N. Given an element $\mathsf{P} \in \mathbb{G}$ of order $r \mid N$ and $\mathsf{Q} \in \langle \mathsf{P} \rangle$, the Discrete Logarithm Problem (DLP) in $\mathbb{G}$ is to compute an integer x (if it exists) such that $[x]\mathsf{P} = \mathsf{Q}$. The integer $0 \leq x < r$ is called the discrete logarithm of Q with respect to the base P.

In this work, we are interested in the case where $\mathbb{G} = \mathcal{E}(\mathbb{F}_{q^\ell})$ for an elliptic curve $\mathcal{E}$ over a binary extension field $\mathbb{F}_{q^\ell}$ with $q = 2^n$ and $\ell \geq 2$. We will be equally interested in the case when $\mathbb{G}$ is the Jacobian $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ of a genus-g curve $\mathcal{H}$ over $\mathbb{F}_q$, and P, Q are divisors belonging to $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$. Solving the DLP

*Corresponding author
Email address: jesus.dominguez@tii.ae (Jesús-Javier Chi-Domínguez)
[1] *Email address:* francisco@cs.cinvestav.mx
[2] *Email address:* smith@lix.polytechnique.fr

in the former group appears to be much more difficult than in the latter when the groups are roughly the same size, especially for larger g.

Indeed, Pollard's Rho algorithm is the best known algorithm to solve DLP instances on a generic elliptic curve defined over a characteristic-two field of the form $\mathbb{F}_{q^\ell}$. This algorithm has an exponential computational complexity of $(1 + o(1))O\left(\frac{1}{2}\sqrt{\pi \cdot q^\ell}\right)$ [1, 2]. On the other hand, using an index-calculus strategy one can solve the DLP on the Jacobian of a curve $\mathcal{H}$ over $\mathbb{F}_q$ with a subexponential complexity of $L_{q^g}\left[\frac{1}{2}, \sqrt{2} + o(1)\right]$ (as q and g tend to infinity).[3]

Weil descent. The Weil descent attack was introduced by Frey in 1998 as a means of transferring DLP instances from an elliptic curve $\mathcal{E}$ defined over an extension field $\mathbb{F}_{q^\ell}$ to the Jacobian of a higher-genus curve $\mathcal{H}$ defined over the subfield $\mathbb{F}_q$ [3]. This transfer becomes useful if the DLP in the Jacobian of the curve $\mathcal{H}/\mathbb{F}_q$ is easier than the DLP on $\mathcal{E}/\mathbb{F}_{q^\ell}$, a situation that usually happens if the genus g of $\mathcal{H}$ is neither too large, nor too small (i.e., $g \geq \ell$ and $g \approx \ell$).

Frey's initial construction was refined by Galbraith and Smart in [4]. Gaudry, Hess, and Smart gave an efficient version of the Weil descent technique (GHS) applied to curves defined over binary extension fields [5]. Galbraith, Hess, and Smart extended this attack to a larger class of curves by transferring the DLP to an isogenous elliptic curve vulnerable to the GHS method [6], and Hess generalized the GHS Weil descent attack from hyperelliptic $\mathcal{H}/\mathbb{F}_q$ to possibly non-hyperelliptic $\mathcal{C}/\mathbb{F}_q$ [7, 8].

Our contributions. As explained above, Weil descent allows us to transfer DLP computations from an elliptic curve $\mathcal{E}/\mathbb{F}_{q^\ell}$ into the Jacobian of a genus-g curve $\mathcal{H}/\mathbb{F}_q$. In this paper, we make three main contributions:

0. We show paper that if $\mathcal{E}$ has a GLS endomorphism, then this induces an efficiently-computable endomorphism of $\mathrm{Jac}_{\mathcal{H}}$. We give an explicit description of this endomorphism in §5.2.
1. We show that if $\mathrm{Jac}_{\mathcal{H}}$ has an efficiently endomorphism with an eigenvalue of order n on $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$, then the relation generation stage of the index-calculus algorithm for solving the DLP in $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ can be accelerated by a factor of n. This in turn implies that the size of the factor base is reduced by a factor of n, which accelerates the linear algebra phase by a factor of n^2. We present an algorithmic analysis of the expected speedup for discrete logarithm computations in $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ in §6.
2. To illustrate our techniques, we present a concrete 115-bit discrete logarithm computation attacking a weak GLS elliptic curve defined over the field $\mathbb{F}_{2^{5 \cdot 31}}$ (see §6.2 for a full description of the problem instance). In our experiments, we observed a factor-5 speedup for the index-calculus computation over $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$. In total, our proof-of-concept implementation in

[3] Recall that $L_X[\alpha, c] = \exp\left((c + o(1))(\log X)^\alpha (\log\log X)^{1-\alpha}\right)$ for $0 < \alpha < 1$ and $c > 0$.

the Magma computational algebra system [9] computed the discrete logarithm in just $1,035$ CPU days, which is significantly less than a discrete logarithm computation reported for the same problem in [10] (cf. Table 1).

To the best of our knowledge, the first two observations have not been previously reported in the literature.

Previous work. In 2001, Menezes and Qu showed that the GHS attack cannot be applied efficiently over binary fields with prime extension degree n in the cryptographically interesting range $n \in \{160, \ldots, 600\}$ [11]. Moreover, Jacobson, Menezes, and Stein studied the GHS Weil descent attack on elliptic curves defined over $\mathbb{F}_{q^{31}}$ with $q = 2^5$ [12]. Maurer, Menezes, and Teske analysed the feasibility of the GHS attack on elliptic curves over binary extension fields with composite extension degree n in the interval $n \in \{100, \ldots, 600\}$ [13]. In 2009, Hankerson, Karabina and Menezes showed that binary Galbraith–Lin–Scott (GLS) elliptic curves (see §2) defined over $\mathbb{F}_{2^{2\ell}}$ are secure against the (generalized) GHS attack when ℓ is a prime in $\{80, \ldots, 256\} \setminus \{127\}$ [14]. Finally, Chi and Oliveira presented an efficient algorithm to determine if a given GLS elliptic curve is vulnerable to the GHS attack [15]. In [10], Velichka *et al.* presented an explicit computation of a discrete logarithm problem using the Weil descent attack on a hyperelliptic genus-32 curve over $\mathbb{F}_{2^5}$.

Recently, but tangentially, Galbraith, Granger, Merz, and Petit [16] showed how DLP computations on Koblitz curves can be sped up using carefully-chosen factor bases, taking advantage of the Frobenius endomorphism acting on these curves. Their techniques resemble the ones presented in this paper, since we reduce the factor base under endomorphism orbits defined on Jacobians of hyperelliptic curves. We believe that our factor base reduction can be easily adapted to the elliptic-curve setting from [16], but this time applied to GLS curves.

Organization. This paper is structured as follows. We (briefly) provide mathematical background on hyperelliptic curves and a general description of the (g)GHS Weil descent attack in §2. Generalized GLS binary curves and their endomorphisms are described in §3. In §4, we present a concrete formulation of the GLS endomorphism induced on the Weil restriction. This is followed in §5 by a concrete definition of the GLS endomorphism on $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$, which is the main result of this paper, together with a detailed discussion of the discrete logarithm computation in $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ by means of a standard index-calculus procedure. It is shown that the GLS endomorphism provides a factor-n acceleration, in theory and in practice. Concluding remarks are made in §7.

2. Mathematical background

We begin with some basic definitions and properties of [hyper]elliptic curves, and a general description of the (g)GHS Weil descent attack. For more in-depth details, the interested reader is referred to [17, 18, 19, 20].

2.1. Binary GLS curves

Let $q = 2^n$. A binary elliptic curve is given by the Weierstrass equation

$$\mathcal{E}/\mathbb{F}_{q^\ell} : y^2 + xy = x^3 + ax^2 + b\,.$$

The set of affine solutions $(x, y) \in \mathbb{F}_{q^\ell} \times \mathbb{F}_{q^\ell}$, together with a point at infinity denoted by $\mathcal{O}$, form an abelian group denoted by $\mathcal{E}(\mathbb{F}_{q^\ell})$. A careful selection of the constants a, b, yields a group order $\#\mathcal{E}(\mathbb{F}_{q^\ell}) = c \cdot r$ where r is a large prime, and c a small cofactor. Let $\langle \mathsf{P} \rangle$ be the order-r subgroup of $\mathcal{E}(\mathbb{F}_{q^\ell})$. Given an integer $0 < k < r$, the elliptic curve scalar multiplication operation computes the multiple $\mathsf{Q} = [k]\mathsf{P}$, corresponding to the sum of k copies of P.

GLS curves, introduced in [21], are cryptographically interesting because they come equipped with an efficiently computable endomorphism ψ, which can be used in the Gallant–Lambert–Vanstone (GLV) scalar multiplication technique of [22]. This splits the computation of $\mathsf{Q} = [k]\mathsf{P}$ into two half-sized scalar multiplications such that

$$\mathsf{Q} = [k]\mathsf{P} = [k_1]\mathsf{P} + [k_2]\psi(\mathsf{P})\,,$$

which can be computed using a two-dimensional multiscalar multiplication algorithm. The authors of [14] reported a family of binary GLS curves over quadratic extensions $\mathbb{F}_{q^2}$ with almost-prime group orders of the form $\#\mathcal{E}_{a,b}(\mathbb{F}_{q^2}) = 2r$, where r is a $(2n-1)$-bit prime. The software and hardware implementations of constant-time variable-base-point elliptic curve scalar multiplication using binary GLS curves rank among the fastest at the 128-bit security level [23, 24, 25].

2.2. Basic definitions and properties of hyperelliptic curves

Let $q = 2^n$ and let $\ell > 1$ be an integer prime to n. Throughout this paper, $\mathcal{E}/\mathbb{F}_{q^\ell}$ is an elliptic curve defined by

$$\mathcal{E}/\mathbb{F}_{q^\ell} : y^2 + x \cdot y = x^3 + a \cdot x^2 + b \quad \text{with} \quad a \in \mathbb{F}_{q^\ell} \quad \text{and} \quad b \neq 0 \in \mathbb{F}_{q^\ell}\,, \quad (1)$$

while $\mathcal{H}/\mathbb{F}_{q^\ell}$ is a genus-g hyperelliptic curve defined by

$$\mathcal{H}/\mathbb{F}_{q^\ell} : y^2 + h(x) \cdot y = f(x),$$

where $f, g \in \mathbb{F}_{q^\ell}[x]$ satisfy $\deg f = 2g + 1$ and $\deg h \leq g$.

The set of $\mathbb{F}_{q^\ell}$-rational points of $\mathcal{H}/\mathbb{F}_{q^\ell}$ is

$$\mathcal{H}\left(\mathbb{F}_{q^\ell}\right) = \left\{(x, y) \in \mathbb{F}_{q^\ell} \times \mathbb{F}_{q^\ell} : y^2 + h(x) \cdot y = f(x)\right\} \cup \{\mathcal{O}\}\,,$$

where $\mathcal{O}$ is the point at infinity. The opposite of any point $\mathsf{P} = (x, y) \in \mathcal{H}(\mathbb{F}_{q^\ell}) \setminus \{\mathcal{O}\}$ is defined as $\overline{\mathsf{P}} = (x, y + h(x))$. If $g = 1$ then $\mathcal{H}$ is an elliptic curve, and $\mathcal{H}(\mathbb{F}_{q^\ell})$ has a group law given by the usual chord-and-tangent rules. However, these rules are not well-defined when $g > 1$. Instead, when $g > 1$ we work with the Jacobian $\mathrm{Jac}_{\mathcal{H}}$ of $\mathcal{H}$. The group of points $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$ can be defined in terms of the group of divisors of $\mathcal{H}/\mathbb{F}_{q^\ell}$. A divisor D is a formal sum of points

on the curve, i.e., $\mathsf{D} = \sum_{\mathsf{P}_i \in \mathcal{H}(\mathbb{F}_{q^\ell})} c_i(\mathsf{P}_i)$ where $c_i = 0$ for all but finitely many points $\mathsf{P}_i \in \mathcal{H}(\mathbb{F}_{q^\ell})$. The degree of D is $\deg \mathsf{D} := \sum c_i$. Every nonzero rational function on $\mathcal{H}$ has an associated principal divisor. In the language of divisors, $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_{q^\ell}) = \mathrm{Div}^0_{\mathcal{H}}(\mathbb{F}_{q^\ell})/\mathrm{Prin}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$, where $\mathrm{Div}^0_{\mathcal{H}}(\mathbb{F}_{q^\ell})$ and $\mathrm{Prin}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$ denote the groups of degree-zero and principal divisors on $\mathcal{H}$, respectively.

Algorithmically, it is more convenient to use the Mumford representation for elements of $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$. Each divisor (class) is represented a pair of polynomials $u, v \in \mathbb{F}_{q^\ell}[x]$ such that u is monic with $\deg u \leq g$, and v satisfies $\deg v < \deg u$ and $u \mid (v^2 + vh - f)$. If $\mathsf{D} = \sum_{i=1}^{g} c_i(P_i) - g(\mathcal{O})$, then $x(P_i)$ is a root of u with multiplicity c_i, and $v(x(P_i)) = y(P_i)$. The divisor corresponding to the pair (u, v) is denoted $\mathsf{div}(u, v)$. The group law on divisors in the Mumford representation can be computed using Cantor's algorithm [20].

Mumford's representation allows us to define notions of irreducibility and smoothness for divisors:

1. $\mathsf{div}(u, v)$ is irreducible if u is irreducible, and
2. $\mathsf{div}(u, v)$ is s-smooth if u is s-smooth.

An important and useful fact is that if $u = \prod_i u_i$ then $\mathsf{div}(u, v) = \sum_i \mathsf{div}(u_i, v \bmod u_i)$.

By the Riemann–Roch theorem, every divisor class in $\mathrm{Jac}_{\mathcal{H}}$ can be represented by a sum of divisors in the form $(\mathsf{P}) - (\mathcal{O})$ with $\mathsf{P} \in \mathcal{H}(\overline{\mathbb{F}}_{q^\ell})$. If $\mathsf{P} = (x_{\mathsf{P}}, y_{\mathsf{P}})$, then $(\mathsf{P}) - (\mathcal{O}) = \mathsf{div}\big(x + x_{\mathsf{P}}, y_{\mathsf{P}}\big)$. Consequently, any divisor $\mathsf{div}(u, v) \in \mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$ can be written as $\sum_i c_i \cdot \mathsf{div}\big(x + x_{\mathsf{P}_i}, y_{\mathsf{P}_i}\big)$, where $(x_{\mathsf{P}_i}, y_{\mathsf{P}_i}) \in \mathcal{H}(\overline{\mathbb{F}}_{q^\ell})$, $u = \prod_i (x + x_{\mathsf{P}_i})^{c_i}$, and $v(x_{\mathsf{P}_i}) = y_{\mathsf{P}_i}$.

Remark 1. *The Jacobian of any elliptic curve $\mathcal{E}/\mathbb{F}_{q^\ell}$ is isomorphic to its group of rational points, i.e.,* $\mathrm{Jac}_{\mathcal{E}}(\mathbb{F}_{q^\ell}) \cong \mathcal{E}(\mathbb{F}_{q^\ell})$.

2.3. Computing discrete logarithms on hyperelliptic curves

As we mentioned in the introduction, the (g)GHS Weil descent technique permits to reduce the DLP in $\mathcal{E}(\mathbb{F}_{q^\ell})$ into the $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$, where $\mathcal{H}/\mathbb{F}_q$ is a hyperelliptic genus-g curve defined over $\mathbb{F}_q$ [5, 6, 7, 8]. Suppose, then, that we want to solve a DLP instance $\mathsf{D}' = \lambda\mathsf{D}$ in $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$, where D and $\mathsf{D}' \in \langle \mathsf{D} \rangle$ have prime order r. The most efficient method for solving the DLP on $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_{q^\ell})$ is an index-calculus approach, consisting of the following steps.

Fix a smoothness bound s, and choose a small positive integer ϵ. Let $F(s)$ be the number of irreducible divisors $\mathsf{div}(u, v) \in \mathrm{Jac}_H(\mathbb{F}_q)$ with $\deg u \leq s$; these divisors form the factor base. We need to generate $F(s) + \epsilon$ relations of the form $\alpha_i\mathsf{D} + \beta_i\mathsf{D}' = \sum_{j=1}^{F(s)} m_{i,j}\mathsf{D}_j$, with the D_j in the factor base, in order to construct three matrices $\alpha = (\alpha_i)^{\mathsf{T}}$, $\beta = (\beta_i)^{\mathsf{T}}$, and $M = (m_{i,j})$ with coefficients in $\mathbb{Z}/r\mathbb{Z}$. Once that this task is completed, we compute an element γ of the kernel of M^{T}; then $\big(\gamma^{\mathsf{T}}\alpha\big)\,\mathsf{D} + \big(\gamma^{\mathsf{T}}\beta\big)\,\mathsf{D}' = 0$. If $\gamma^{\mathsf{T}}\beta = 0$, then we must repeat the whole procedure (or at least try a different γ); otherwise, the discrete logarithm of D' with respect to D is $\lambda = -(\gamma^{\mathsf{T}}\alpha)/(\gamma^{\mathsf{T}}\beta)$.

When the genus of the curve produced by the (g)GHS Weil descent attack is large with respect to the finite field size, the most efficient choice for the DLP

on higher-genus hyperelliptic curves is the Enge-Gaudry algorithm [26, 27], with a subexponential running-time complexity of

$$L_{q^g}\left[\frac{1}{2}, \sqrt{2}+o(1)\right] = \exp\Big(\sqrt{2}+o(1)\Big)\sqrt{\log q^g}\sqrt{\log\log q^g}.$$

We say that the elliptic curve $\mathcal{E}/\mathbb{F}_{q^\ell}$ is vulnerable (or weak) against the (g)GHS Weil descent attack if the computational cost of the Enge-Gaudry algorithm on the hyperelliptic curve constructed by the GHS attack is less than that of Pollard's rho algorithm.

In the concrete discrete logarithm computation of §6.2, (g)GHS Weil descent produces a hyperelliptic genus-32 curve $\mathcal{H}/\mathbb{F}_q$, with $q = 2^5$. In other words, for our discrete logarithm computation we work with the case $g = q$.

Remark 2. *For curves of small genus $g \geq 3$, the algorithm of Gaudry, Thomé, Thériault, and Diem [28] is the most efficient choice for solving DLPs. For genus-2 curves, Pollard's Rho algorithm is more efficient.*

2.4. Costs of the index-calculus based algorithm

The two main steps of the index-calculus approach are the search for s-smooth divisors, and the computation of a kernel element, which is handled as a linear algebra problem. For the first task, one can approximate the cost of finding s-smooth divisors search as follows (for more details see [12]): If $A_{s'}$ is the number of irreducible divisors $\mathsf{div}(u,v) \in \mathrm{Jac}_H(\mathbb{F}_q)$ with $\deg u = s'$, then

$$A_{s'} \approx \frac{1}{2}\cdot\frac{1}{s'}\sum_{d|s'}\mu\Big(\frac{s'}{d}\Big)q^d\,,$$

where μ denotes the Möbius function, i.e., $\mu(n) = (-1)^k$ if n is squarefree with k different prime factors, and 0 if n is not squarefree. Consequently, $F(s) \approx \sum_{i=1}^{s} A_i$. On the other hand, the number of s-smooth divisors $\mathsf{div}(u,v) \in \mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ with $\deg u \leq g$ is

$$M(g,s) = \sum_{i=1}^{g}\left([x^i]\prod_{s'=1}^{s}\left(\frac{1+x^{s'}}{1-x^{s'}}\right)^{A_{s'}}\right),$$

where [.] denotes the coefficient operator. When $A_{s'}$ is known, $M(g,s)$ can be computed by finding the first $(g+1)$ terms of the Taylor expansion of $\prod_{s'=1}^{s}\left(\frac{1+x^{s'}}{1-x^{s'}}\right)^{A_{s'}}$ around $x=0$, and summing the coefficients of $x, x^2, \ldots, x^g$.

The expected number of random-walk steps before encountering an s-smooth divisor is therefore

$$E(s) = \frac{\#\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)}{M(g,s)} \approx \frac{q^g}{M(g,s)}\,,$$

and the expected number of steps before $F(s)+\epsilon$ relations are generated is

$$T(s) = (F(s)+\epsilon)\,E(s)\,.$$

For the linear algebra task, Magma uses Lanczos' algorithm, with approximate running time $L(s) \approx d \cdot (F(s) + \epsilon)^2$ where d denotes the per-row density of the matrix M. In fact, it can be shown that $d \leq g$.

3. The GLS endomorphism

Let $\mathcal{E}$ be an elliptic curve over $\mathbb{F}_{q^\ell}$, defined by Equation (1) with $a \in \mathbb{F}_q \subset \mathbb{F}_{q^\ell}$ and $b \in \mathbb{F}_{2^\ell} \subset \mathbb{F}_{q^\ell}$. For each integer $i \geq 0$, we define an elliptic curve

$$\mathcal{E}_i/\mathbb{F}_{q^\ell} \colon y^2 + xy = x^3 + a^{2^i} x^2 + b^{2^i} .$$

The curves $\mathcal{E} = \mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_{n\cdot\ell-1}, \mathcal{E}_{n\cdot\ell} = \mathcal{E}$ are connected by a cycle of 2-power Frobenius maps $\mathcal{E}_i/\mathbb{F}_{q^\ell} \to \mathcal{E}_{i+1}/\mathbb{F}_{q^\ell}$ mapping $(x, y) \mapsto (x^2, y^2)$. Abusing notation, we will write π for each of these maps and π^k for the composition of any k successive ones. Since b is in $\mathbb{F}_{2^\ell}$, the curve $\mathcal{E}_\ell/\mathbb{F}_{q^\ell}$ is isomorphic to $\mathcal{E}/\mathbb{F}_{q^\ell}$; the isomorphism is

$$\begin{aligned} \phi \colon \mathcal{E}_\ell/\mathbb{F}_{q^\ell} &\longrightarrow \mathcal{E}/\mathbb{F}_{q^\ell} \\ (x, y) &\longmapsto (x, y + \delta x) , \end{aligned}$$

where $\delta^2 + \delta = a + a^{2^\ell}$. If $n \cdot \ell$ is odd, then $\delta \in \mathbb{F}_{q^\ell} \setminus \mathbb{F}_{2^\ell}$, so the isomorphism ϕ is defined over $\mathbb{F}_{q^\ell}$, and in particular $\delta = \sum_{j=0}^{\frac{n\cdot\ell-1}{2}} \left(a + a^{2^\ell}\right)^{2^{2j}}$.

Composing the 2^ℓ-power Frobenius $\pi^\ell \colon \mathcal{E} \to \mathcal{E}_\ell$ with the isomorphism $\phi \colon \mathcal{E}_\ell \to \mathcal{E}$, we obtain a generalized Galbraith–Lin–Scott (GLS) endomorphism

$$\psi \coloneqq \phi \circ \pi^\ell \colon (x, y) \longmapsto \left(x^{2^\ell}, y^{2^\ell} + \delta x^{2^\ell}\right) \in \mathrm{End}(\mathcal{E}).$$

The endomorphism ψ is defined over $\mathbb{F}_{q^\ell}$ and satisfies $\psi^n = \pm\pi^{n\ell}$; in particular, ψ^n acts as $[1]$ or $[-1]$ on points of $\mathcal{E}(\mathbb{F}_{q^\ell})$.

Endomorphisms such as ψ are cryptographically interesting because they can be used to accelerate scalar multiplication on $\mathcal{E}$, by applying the technique of Gallant, Lambert, and Vanstone (for more details, see [22]). In the sequel, we will show that these endomorphisms can also be used to improve the efficiency of the Gaudry–Hess–Smart Weil descent attack on weak curves of this kind.

4. Extending the GLS endomorphism

From now on, we fix an element w of $\mathbb{F}_{q^\ell}$ such that $w + w^2 + \cdots + w^{2^{\ell-1}} = 1$ and

$$\mathbb{F}_{q^\ell} = \mathbb{F}_q(w) = \left\langle w, w^2, w^4, \ldots, w^{2^{\ell-1}} \right\rangle_{\mathbb{F}_q};$$

that is, $\{w, w^2, w^4, \ldots, w^{2^{\ell-1}}\}$ is a normal basis for $\mathbb{F}_{q^\ell}$ over $\mathbb{F}_q$.

Recall that the Weil restriction

$$\mathcal{A}_i/\mathbb{F}_q \coloneqq \mathcal{W}_{\mathbb{F}_q}^{\mathbb{F}_{q^\ell}} (\mathcal{E}_i)$$

of $\mathcal{E}_i$ from $\mathbb{F}_{q^\ell}$ to $\mathbb{F}_q$ is an ℓ-dimensional abelian variety over $\mathbb{F}_q$, and that there is an isomorphism of groups $\mathcal{A}_i(\mathbb{F}_q) \cong \mathcal{E}_i(\mathbb{F}_{q^\ell})$.[4] The various isogenies and endomorphisms of $\mathcal{E}_i$ induce isogenies and endomorphisms of $\mathcal{A}_i$.

We will use the following explicit affine model for $\mathcal{A}_i$. Consider the polynomial ring $R = \mathbb{F}_q[x_0, x_1, \ldots, x_{\ell-1}, y_0, \ldots, y_{\ell-1}]$, and set

$$X = \sum_{j=0}^{\ell-1} x_j w^{2^j} \qquad \text{and} \qquad Y = \sum_{j=0}^{\ell-1} y_j w^{2^j}$$

in $R \otimes \mathbb{F}_{q^\ell}$. Expanding the defining equation of $\mathcal{E}_i$ in the variables X and Y, there exist $W_0, \ldots, W_{\ell-1}$ in R such that $Y^2 + XY - (X^3 - (a^{2^i})X^2 - b) = \sum_{j=0} W_j w^{2^j}$ in $R \otimes \mathbb{F}_{q^\ell}$. The affine scheme $\mathrm{Spec}(R/(W_0, \ldots, W_{\ell-1}))$ is then $\mathbb{F}_q$-isomorphic to an open affine subset of $\mathcal{A}_i$. By construction, we have a bijection of sets

$$\begin{aligned} \iota \colon \mathcal{E}_i(\mathbb{F}_{q^\ell}) &\longrightarrow \mathcal{A}_i(\mathbb{F}_q) \\ (x, y) &\longmapsto (x_0, \ldots, x_{n-1}, y_0, \ldots, y_{n-1}), \end{aligned}$$

where $x = \sum_{j=0}^{\ell-1} x_j w^{2^j}$ and $y = \sum_{j=0}^{\ell-1} y_j w^{2^j}$. In fact, ι is an isomorphism of groups.

We want to make the isogenies and endomorphisms of $\mathcal{A}_i$ corresponding to π, ϕ, and ψ completely explicit with respect to this affine model of $\mathcal{A}_i$. First, observe that if $X = \sum_{j=0}^{\ell-1} x_j w^{2^j}$, then $X^2 = \sum_{j=0}^{\ell-1} x_j^2 w^{2^{j+1}}$, so the 2-powering Frobenius isogeny $\pi : \mathcal{E}_i \to \mathcal{E}_{i+1}$ corresponds to an isogeny $\Pi : \mathcal{A}_i \to \mathcal{A}_{i+1}$ that squares and cyclically permutes the coordinates:

$$\Pi : (x_0, \ldots, x_{\ell-1}, y_0, \ldots, y_{\ell-1}) \longmapsto (x_{\ell-1}^2, x_0^2, \ldots, x_{\ell-2}^2, y_{\ell-1}^2, y_0^2, \ldots, y_{\ell-2}^2).$$

The isomorphism $\phi : \mathcal{E}_\ell \to \mathcal{E}$ maps (X, Y) to $(X, Y + \delta X)$, and δ is in $\mathbb{F}_q$ because a is, so ϕ corresponds to an isomorphism $\Phi : \mathcal{A}_\ell \to \mathcal{A}$ defined by

$$\Phi : (x_0, \ldots, x_{\ell-1}, y_0, \ldots, y_{\ell-1}) \longmapsto (x_0, \ldots, x_{\ell-1}, y_0 + \delta x_0, \ldots, y_{\ell-1} + \delta x_{\ell-1}).$$

As with π^ℓ and ϕ on the elliptic curves, composing $\Pi^\ell : \mathcal{A} \to \mathcal{A}_\ell$ with $\Phi : \mathcal{A}_\ell \to \mathcal{A}$ yields an endomorphism Ψ of $\mathcal{A}$, defined (over $\mathbb{F}_q$) by

$$\Psi : (x_0, \ldots, x_{\ell-1}, y_0, \ldots, y_{\ell-1}) \longmapsto \left(x_0^{2^\ell}, \ldots, x_{\ell-1}^{2^\ell}, y_0^{2^\ell} + \delta x_0^{2^\ell}, \ldots, y_{\ell-1}^{2^\ell} + \delta x_{\ell-1}^{2^\ell}\right).$$

On groups of points we have $\Pi = \iota \circ \pi \circ \iota^{-1}$, $\Phi = \iota \circ \phi \circ \iota^{-1}$, and $\Psi = \iota \circ \psi \circ \iota^{-1}$. The relationships between all of these various maps are summarized in Figure 1.

We note that if $\mathcal{G}$ is a cyclic subgroup of $\mathcal{E}(\mathbb{F}_{q^\ell})$ of order r, and ψ acts on $\mathcal{G}$ as multiplication by some eigenvalue $\lambda \pmod r$, then Ψ must act on $\iota(\mathcal{G}) \subseteq \mathcal{A}(\mathbb{F}_q)$ as multiplication by exactly the same eigenvalue λ.

[4] More generally, for any algebra K over $\mathbb{F}_q$, there is an isomorphism between $\mathcal{E}_i(\mathbb{F}_{q^\ell} \otimes_{\mathbb{F}_q} K)$ and $\mathcal{A}_i(K)$; in fact, $\mathcal{A}_i$ is the group scheme realizing the functor $K \mapsto \mathcal{E}_i(\mathbb{F}_{q^\ell} \otimes_{\mathbb{F}_q} K)$.

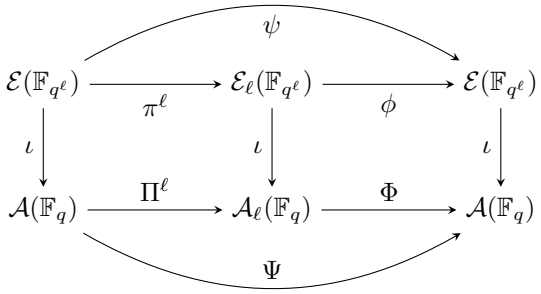

Figure 1: Endomorphism diagram

5. Combining the GLS and GHS techniques

The generalized GHS (gGHS) Weil descent technique constructs a genus-g algebraic curve $\mathcal{C}/\mathbb{F}_q$ (not necessary hyperelliptic) by computing the Weil restriction $\mathcal{A}/\mathbb{F}_q$ of $\mathcal{E}/\mathbb{F}_{q^\ell}$, intersecting $\mathcal{A}/\mathbb{F}_q$ with $(\ell-1)$-dimensional hyperplanes to obtain a subvariety $\mathcal{A}'/\mathbb{F}_q$ of $\mathcal{A}/\mathbb{F}_q$, and finding an irreducible component $\mathcal{C}/\mathbb{F}_q$ of $\mathcal{A}'/\mathbb{F}_q$ (for more details see [5, 6, 7, 8]).

Let us intersect $\mathcal{A}/\mathbb{F}_q$ with the hyperplanes $x_0 = x_1 = \cdots = x_{\ell-1} = x \in \mathbb{F}_q$. With $a \in \mathbb{F}_q$ and $b \in \mathbb{F}_{2^\ell}$, and using the linear independence property of a normal basis $\{w, w^2, w^4, \ldots, w^{2^{\ell-1}}\}$, we obtain a subvariety $\mathcal{A}'/\mathbb{F}_q$ of $\mathcal{A}/\mathbb{F}_q$ defined by

$$\mathcal{A}'/\mathbb{F}_q\colon \begin{cases} x^3 + a\cdot x^2 + x\cdot y_0 + y_{\ell-1}^2 + b_0 &= 0 \\ x^3 + a\cdot x^2 + x\cdot y_1 + y_0^2 + b_1 &= 0 \\ & \vdots \\ x^3 + a\cdot x^2 + x\cdot y_{\ell-1} + y_{\ell-2}^2 + b_{\ell-1} &= 0 \end{cases}$$

where $b = \sum_{i=0}^{\ell-1} b_i w^{2^i}$ and each b_i is in $\mathbb{F}_2$. Thus, if $\mathcal{A}'_\ell/\mathbb{F}_q$ is the variety determined by Equation (2), then Φ induces an endomorphism of $\mathcal{A}'_\ell(\mathbb{F}_q)$.

$$\mathcal{A}'_\ell/\mathbb{F}_q\colon \begin{cases} x^3 + a^{2^\ell}\cdot x^2 + x\cdot y_0 + y_{\ell-1}^2 + b_0 &= 0 \\ x^3 + a^{2^\ell}\cdot x^2 + x\cdot y_1 + y_0^2 + b_1 &= 0 \\ & \vdots \\ x^3 + a^{2^\ell}\cdot x^2 + x\cdot y_{\ell-1} + y_{\ell-2}^2 + b_{\ell-1} &= 0 \end{cases} \tag{2}$$

5.1. New endomorphism on the hyperelliptic curve

Let $\mathcal{H}/\mathbb{F}_q\colon y^2 + h(x)\cdot y = f(x)$ be a genus-g hyperelliptic curve that is an irreducible component of $\mathcal{A}'/\mathbb{F}_q$. Writing $h(x) = \sum_{i=0}^{g} h_i x^i$ and $f(x) = \sum_{i=0}^{2g+1} f_i x^i$, the corresponding hyperelliptic irreducible component $\mathcal{H}_\ell/\mathbb{F}_q$ of $\mathcal{A}'_\ell/\mathbb{F}_q$ is

$$\mathcal{H}_\ell/\mathbb{F}_q\colon y^2 + ({}^\sigma h)(x)\cdot y = ({}^\sigma f)(x)$$

where $({}^{\sigma}h)(x) = \sum_{i=0}^{g} \sigma(h_i) \cdot x^i$, $({}^{\sigma}f)(x) = \sum_{i=0}^{2g+1} \sigma(f_i) \cdot x^i$, and $\sigma(x) = x^{2^\ell}$ for all $x \in \mathbb{F}_q$. Therefore, the maps $\Pi^\ell \colon \mathcal{H}/\mathbb{F}_q \to \mathcal{H}_\ell/\mathbb{F}_q$ and $\Phi \colon \mathcal{H}_\ell/\mathbb{F}_q \to \mathcal{H}/\mathbb{F}_q$ are defined by

$$\Pi^\ell \colon (x, y) \longmapsto \left(x^{2^\ell}, y^{2^\ell}\right) \quad \text{and} \quad \Phi \colon (x, y) \longmapsto \left(\delta_1 \cdot x + \delta_2, \delta_3 \cdot y + t(x)\right)$$

for some $\delta_1, \delta_2, \delta_3 \in \mathbb{F}_q$ and $t(x) \in \mathbb{F}_q[x]$ with $\deg t(x) \leq g$ and $\delta_1 \neq 0$.[5] Consequently, $\Psi = \Phi \circ \Pi^\ell$ induces the following endomorphism:

$$\begin{aligned} \Psi^* \colon \mathrm{Jac}_{\mathcal{H}}\left(\overline{\mathbb{F}_q}\right) &\longrightarrow \mathrm{Jac}_{\mathcal{H}}\left(\overline{\mathbb{F}_q}\right) \\ \sum_j c_j \left(\mathsf{P}_j\right) &\longmapsto \sum_j c_j \left(\Psi(\mathsf{P}_j)\right). \end{aligned}$$

In Mumford's representation, the divisor $\mathsf{div}(u, v) = \sum_j c_j \cdot \mathsf{div}(x + x_{\mathsf{P}_j}, y_{\mathsf{P}_j})$ is mapped to $\sum_j c_j \cdot \mathsf{div}(x + x_{\Psi(\mathsf{P})_j}, y_{\Psi(\mathsf{P})_j})$, and therefore $\mathbb{F}_q$-irreducible factors of u are mapped to irreducible factors of the same degree, i.e., Ψ^* sends smooth divisors to smooth divisors.

The curve $\mathcal{H}/\mathbb{F}_q$ has genus $g \geq \ell$, so its Jacobian $\mathrm{Jac}_{\mathcal{H}}$ is g-dimensional. By the universal property of the Jacobian, the ℓ-dimensional $\mathcal{A}$ is a quotient (and so an isogeny factor) of $\mathrm{Jac}_{\mathcal{H}}$.[6] Hence, $\mathrm{Jac}_{\mathcal{H}} \cong \mathcal{A} \times \mathcal{B}$ for some $(g-\ell)$-dimensional abelian variety $\mathcal{B}$. The situation is illustrated by the diagram in Figure 2.

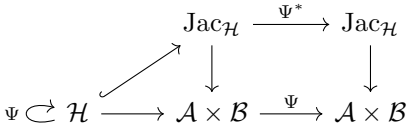

Figure 2: Endomorphism diagram for $\mathcal{H}/\mathbb{F}_q$

If $\mathcal{G}$ is a cyclic subgroup of $\mathcal{E}(\mathbb{F}_{2^{\ell \cdot n}})$ fixed by ψ, then ψ acts on $\mathcal{G}$ as multiplication by an eigenvalue λ, and so Ψ acts on $\iota(\mathcal{G})$ as multiplication by λ. Hence, $(\Psi^*)^n = [1]$ or $(\Psi^*)^n = [-1]$, and therefore $t(x) = \delta_4({}^{\sigma}h)(\delta_5 \cdot x)$ for some $\delta_4, \delta_5 \in \mathbb{F}_q$. The morphism $\Psi^n \colon \mathcal{H}/\mathbb{F}_q \to \mathcal{H}/\mathbb{F}_q$ fixes the x-coordinate, and

$$x_{\Psi^n(\mathsf{P})} = \delta_1^{\left(\sum_{k=0}^{n-1} 2^{k \cdot \ell}\right)} \cdot x_{\mathsf{P}}^{\left(2^{n \cdot \ell}\right)} + \sum_{i=0}^{n-1} \delta_1^{\left(\sum_{k=0}^{i-1} 2^{k \cdot \ell}\right)} \cdot \delta_2^{\left(2^{i \cdot \ell}\right)}.$$

But $(\ell, n) = 1$ and $q = 2^n$, so $\delta_1^{\left(\sum_{k=0}^{n-1} 2^{k \cdot \ell}\right)} = (\delta_1)^{2^n - 1} = (\delta_1)^{q-1} = 1$, while $x_{\mathsf{P}}^{2^{n \cdot \ell}} = x_{\mathsf{P}}^{q^\ell} = x_{\mathsf{P}}$ and $x_{\Psi^n(\mathsf{P})} = x_{\mathsf{P}} + \sum_{i=0}^{n-1} \delta_1^{\left(\sum_{k=0}^{i-1} 2^{k \cdot \ell}\right)} \cdot \delta_2^{\left(2^{i \cdot \ell}\right)}$. Therefore, $\sum_{i=0}^{n-1} \delta_1^{\left(\sum_{k=0}^{i-1} 2^{k \cdot \ell}\right)} \cdot \delta_2^{\left(2^{i \cdot \ell}\right)} = 0$. It follows that $\delta_2 = 0$.

[5] Any isomorphism of hyperelliptic curves over a finite field is in the form of Φ (for more details see [19, Section 10.2]).

[6] Universal property: let $\kappa \colon \mathcal{H} \to \tilde{\mathcal{A}}$ be a morphism, where $\tilde{\mathcal{A}}$ is an abelian variety. Let $\mathsf{P}_0 \in \mathcal{H}(\overline{\mathbb{F}_q})$ be such that $\kappa(\mathsf{P}_0) = 0$, and consider the map $\tilde{\kappa} \colon \mathcal{H} \to \mathrm{Jac}_{\mathcal{H}}$ given by $\mathsf{P} \mapsto (\mathsf{P}) - (\mathsf{P}_0)$. Then there is a unique homomorphism $\psi \colon \mathrm{Jac}_{\mathcal{H}} \to \tilde{\mathcal{A}}$ of abelian varieties such that $\kappa = \psi \circ \tilde{\kappa}$ (for more details see [19, Section 10.5]).

5.2. Explicit description of the new endomorphism

Recall that for any point $\mathsf{P} = (x_\mathsf{P}, y_\mathsf{P}) \in \mathcal{H}(\mathbb{F}_q)$ its corresponding divisor $(\mathsf{P}) - (\mathcal{O})$ is equal to $\mathsf{div}\big(x + x_\mathsf{P}, y_\mathsf{P}\big)$, and any divisor $\mathsf{div}(u, v) \in \mathrm{Jac}_\mathcal{H}(\mathbb{F}_{q^\ell})$ can be written as $\sum_i c_i \cdot \mathsf{div}\big(x + x_{\mathsf{P}_i}, y_{\mathsf{P}_i}\big)$, where $(x_{\mathsf{P}_i}, y_{\mathsf{P}_i}) \in \mathcal{H}(\overline{\mathbb{F}}_{q^\ell})$, $u = \prod_i (x + x_{\mathsf{P}_i})^{c_i}$, and $v(x_{\mathsf{P}_i}) = y_{\mathsf{P}_i}$. The divisor $\Psi^*\big((\mathsf{P})\big) = \big(\Psi(\mathsf{P})\big)$ is therefore equal to $\mathsf{div}\big(x + \big(\delta_1 x_\mathsf{P}^{2^\ell}\big), \delta_3 y_\mathsf{P}^{2^\ell} + \delta_4({}^\sigma h)\big(\delta_5\delta_1 x_\mathsf{P}^{2^\ell}\big)\big)$, and any divisor $\mathsf{div}(u, v) \in \mathrm{Jac}_\mathcal{H}(\mathbb{F}_q)$ satisfies $\mathsf{div}(u, v) = \sum_{i=0}^{\deg u} \mathsf{div}(x + x_i, v(u_i)) = \sum_{i=0}^{\deg u} \big((x_i, v(x_i))\big)$, where $x_0, x_1, \ldots, x_{\deg u} \in \overline{\mathbb{F}}_q$ are the roots of u. Further,

$$\begin{aligned}
\Psi^*\big(\mathsf{div}(u, v)\big) &= \Psi^*\left(\sum_{i=0}^{\deg u} \Big(\big(x_i, v(x_i)\big)\Big)\right) = \sum_{i=0}^{\deg u} \Psi^*\Big(\big(x_i, v(x_i)\big)\Big) \\
&= \sum_{i=0}^{\deg u} \mathsf{div}\Big(x + \big(\delta_1 {x_i}^{2^\ell}\big), \delta_3\big(v(x_i)\big)^{2^\ell} + \delta_4({}^\sigma h)\big(\delta_5\delta_1 {x_i}^{2^\ell}\big)\Big) \\
&= \sum_{i=0}^{\deg u} \mathsf{div}\Big(\delta_1\Big(\frac{x}{\delta_1} + {x_i}^{2^\ell}\Big), \delta_3({}^\sigma v)\big(x_i^{2^\ell}\big) + \delta_4({}^\sigma h)\big(\delta_5\delta_1 {x_i}^{2^\ell}\big)\Big).
\end{aligned}$$

We want to find polynomials $u^*, v^* \in \mathbb{F}_q[x]$ such that $\Psi^*\big(\mathsf{div}(u, v)\big) = \mathsf{div}(u^*, v^*)$, $u^*\big(\delta_1 {x_i}^{2^\ell}\big) = 0$, and $v^*\big(\delta_1 {x_i}^{2^\ell}\big) = \delta_3\big(v(x_i)\big)^{2^\ell} + \delta_4({}^\sigma h)\big(\delta_5\delta_1 {x_i}^{2^\ell}\big)$. In particular, $u^*(x) = \delta_1^{\deg u} \cdot ({}^\sigma u)\big(\frac{x}{\delta_1}\big) = \prod_{i=0}^{\deg u}\Big(x + \delta_1 \cdot {x_i}^{2^\ell}\Big)$, and $v^*(x) = \delta_3({}^\sigma v)\Big(\frac{x}{\delta_1}\Big) + \delta_4\big({}^\sigma h\big)(\delta_5 x)$. Moreover, $\deg v^* < \deg u^* \le g$ and $u^* \mid \big((v^*)^2 + (v^* \cdot h) + f\big)$, so we can set $v^*(x) \coloneqq \delta_3({}^\sigma v)\big(\frac{x}{\delta_1}\big) + \big(\delta_4\big({}^\sigma h\big)(\delta_5 x)\big) \mod u^*(x)$.

The endomorphism Ψ^* must be well-defined in the sense that v^* should be the same if we reduce h modulo u from the beginning. This observation implies

$$\delta_4({}^\sigma h)(\delta_5 x) \equiv \delta_4\big({}^\sigma(h \bmod u)\big)(\delta_5 x) \pmod{u^*(x)}. \tag{3}$$

Write $u = \sum_i u_i x^i$, $v = \sum_i v_i x^i$, and $h = \sum_i h_i x^i$. If $(h \mod u) = \sum_i h'_i x^i$ and $\delta_4({}^\sigma h)(\delta_5 x) \mod u^*(x) = \sum_i h^*_i x^i$, then we have $u^* = \delta_1^{\deg u} \sum_i ({u_i}^{2^\ell}/\delta_1^i) x^i$, $\delta_3({}^\sigma v)(x/\delta_1) = \delta_3 \sum_i ({v_i}^{2^\ell}/\delta_1^i) x^i$, and $\delta_4\big({}^\sigma(h \bmod u)\big)(\delta_5 x) = \delta_4 \sum_i \big(h'_i\big)^{2^\ell} \delta_5^i x^i$. In particular, Equation (3) holds for any $\mathsf{div}(u, v) \in \mathrm{Jac}_\mathcal{H}(\mathbb{F}_q)$.

Let us analyze the cases $\deg u = g$ and $\deg u \le g$ separately.

The case $\deg u = g$. In this case, we can write

$$h'_i = c_{h_i} + h_g \cdot u_i \quad \text{and} \quad h^*_i = \delta_4 \cdot \Big({h_i}^{2^\ell} \cdot \delta_5^i + \big({h_g}^{2^\ell} \cdot \delta_5^g\big) \cdot \delta_1^{g-i}\Big). \tag{4}$$

Taking Equations (3) and (4) together, for each $i = 0, \ldots, (g-1)$ we have

$$\delta_4 \cdot (h_i + h_g \cdot u_i)^{2^\ell} \cdot \delta_5^i = \delta_4 \cdot \big({h_i}^{2^\ell} \cdot \delta_5^i + \big({h_g}^{2^\ell} \cdot \delta_5^g\big) \cdot \delta_1^{g-i}\big).$$

This equation is satisfied if and only if $\delta_5^i = \delta_5^g \cdot \delta_1^{g-i}$ for each $0 \le i < g$. Moreover, $\delta_5 = \frac{1}{\delta_1}$ and

$$\Psi^*\big(\mathsf{div}(u, v)\big) = \mathsf{div}\Big(\delta_1^{\deg u} \cdot ({}^\sigma u)\Big(\frac{x}{\delta_1}\Big), \delta_3({}^\sigma v)\Big(\frac{x}{\delta_1}\Big) + \delta_4\big({}^\sigma(h \bmod u)\big)\Big(\frac{x}{\delta_1}\Big)\Big).$$

The case $\deg u \leq g$. Let us consider again the general case when $\deg u \leq g$. Suppose $\mathsf{div}(u, v)$ is a divisor of maximal prime order r (where r is a large prime factor of $\#\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ and $r^2 \nmid \#\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$). Then Ψ^* acts on $\langle \mathsf{div}(u, v) \rangle$ as multiplication by an eigenvalue $0 \leq \lambda < r$: that is, $\Psi^*(\mathsf{div}(u, v)) = [\lambda]\mathsf{div}(u, v)$. We can compute $\mathsf{div}(u', v') \coloneqq [\lambda]\mathsf{div}(u, v)$ using Cantor's algorithm, and then $\mathsf{div}(u^*, v^*) \coloneqq \Psi^*(\mathsf{div}(u, v))$ must be equal to $\mathsf{div}(u', v')$.

Write $u' = \sum_i u'_i x^i$ and $v' = \sum_i v'_i x^i$. Then $u^* = u'$ and $v^* = v'$ imply that $\tilde{\delta_1} = \frac{1}{\delta_1}, \delta_3, \delta_4 \in \mathbb{F}_q$ must belong to the varieties

$$V_1/\mathbb{F}_q\colon \begin{cases} \left(u_0\right)^{2^\ell} & = \left(\tilde{\delta}_1\right)^{\deg u} \cdot u'_0 \\ \left(u_1\right)^{2^\ell} \cdot \tilde{\delta}_1 & = \left(\tilde{\delta}_1\right)^{\deg u} \cdot u'_1 \\ \left(u_2\right)^{2^\ell} \cdot \left(\tilde{\delta}_1\right)^2 & = \left(\tilde{\delta}_1\right)^{\deg u} \cdot u'_2 \\ & \vdots \\ \left(u_{\deg u-1}\right)^{2^\ell} \cdot \left(\tilde{\delta}_1\right)^{\deg u-1} & = \left(\tilde{\delta}_1\right)^{\deg u} \cdot u'_{\deg u-1} \end{cases}$$

and

$$V_{3,4}/\mathbb{F}_q\colon \begin{cases} \delta_3 \cdot \left(v_0\right)^{2^\ell} + \delta_4 \cdot \left(h'_0\right)^{2^\ell} & = v'_0 \\ \delta_3 \cdot \left(v_1\right)^{2^\ell} \cdot \tilde{\delta}_1 + \delta_4 \cdot \left(h'_1\right)^{2^\ell} \cdot \tilde{\delta}_1 & = v'_1 \\ \delta_3 \cdot \left(v_2\right)^{2^\ell} \cdot \left(\tilde{\delta}_1\right)^2 + \delta_4 \cdot \left(h'_2\right)^{2^\ell} \cdot \left(\tilde{\delta}_1\right)^2 & = v'_2 \\ & \vdots \\ \delta_3 \cdot \left(v_{\deg v}\right)^{2^\ell} \cdot \left(\tilde{\delta}_1\right)^{\deg v} + \delta_4 \cdot \left(h'_{\deg v}\right)^{2^\ell} \cdot \left(\tilde{\delta}_1\right)^{\deg v} & = v'_{\deg v} \end{cases}$$

Observe that V_1 only depends on the parameter $\tilde{\delta}_1$, and it is determined by $(\deg u)$ polynomial equations of degree at most $\deg u$. In particular, the $(\deg u)$-th equation of V_1 implies $\tilde{\delta}_1 = \frac{\left(u_{\deg u-1}\right)^{2^\ell}}{u'_{\deg_u - 1}}$, if $u_{\deg u-1} \neq 0$. Otherwise, the i-th and j-th equations of V_1 with $j < i$ imply $\tilde{\delta_1}^{i-j} = \frac{(u_j)^{2^\ell} \cdot u'_i}{(u_i)^{2^\ell} \cdot u'_j}$ when $u_i \cdot u_j \neq 0$.

The variety $V_{3,4}/\mathbb{F}_q$ only depends on the parameters δ_3 and δ_4, and it is determined by $(\deg v + 1)$ linear equations; in fact, $V_{3,4}(\mathbb{F}_q)$ consists of a unique point $(\delta_3, \delta_4) \in \mathbb{F}_q \times \mathbb{F}_q$. Combining the i-th and j-th equations of $V_{3,4}$ yields

$$\delta_3 = \frac{v'_i \cdot (\delta_1)^i + \delta_4 \cdot \left(h'_i\right)^{2^\ell}}{\left(v_i\right)^{2^\ell}} \text{ and} \tag{5}$$

$$\delta_4 = \frac{v'_i \cdot (\delta_1)^i \cdot \left(v_j\right)^{2^\ell} + v'_j \cdot (\delta_1)^j \cdot \left(v_i\right)^{2^\ell}}{\left(h'_i\right)^{2^\ell} \cdot \left(v_j\right)^{2^\ell} + \left(h'_j\right)^{2^\ell} \cdot \left(v_i\right)^{2^\ell}} \tag{6}$$

where the denominators of Equations (5) and (6) are different from zero.

6. Speeding-up the Index-Calculus algorithm in $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$

We now focus on the application of Ψ^* to index calculus in $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$.

6.1. The speed-up in theory

Recall from §2.3 that there are two main steps in index calculus. First, the **relation generation** step: having fixed a smoothness bound s, we generate $F(s)+\epsilon$ relations of the form $\alpha_i \mathsf{D} + \beta_i \mathsf{D}' = \sum_{j=1}^{F(s)} m_{i,j}\mathsf{D}_j$, where $\mathsf{D}' = \lambda\mathsf{D}$ is the target DLP, the D_j are irreducible divisors of degree $\leq s$, where $F(s)$ is the number of irreducible divisors $\mathsf{div}(u,v)$ with $\deg u \leq s$. For the **linear algebra** step, we construct the matrices $\alpha = (\alpha_i)^{\mathsf{T}}$, $\beta = (\beta_i)^{\mathsf{T}}$ and $M = (m_{i,j})$ with coefficients in $\mathbb{Z}/r\mathbb{Z}$; the discrete logarithm can be recovered from a kernel vector γ of M.

From §2.4, relation generation requires $T(s) = (F(s)+\epsilon)E(s)$ random-walk steps, where $E(s)$ denotes the expected number of steps before an s-smooth divisor is found. The kernel computation in the linear algebra step requires $L(s) \approx d \cdot (F(s)+\epsilon)^2$ field operations.

Since the eigenvalue λ of Ψ^* satisfies $\lambda \neq \mp 1$ and $\lambda^n \pm 1 \equiv 0 \mod r$, the divisors $\mathsf{D}, [\lambda]\mathsf{D}, \ldots, [\lambda^{n-2}]\mathsf{D}$ and $[\lambda^{n-1}]\mathsf{D}$ must be linearly dependent. Hence, whenever an s-smooth divisor is found, the endomorphism Ψ^* allows us to obtain up to $n-1$ more s-smooth divisors at essentially no cost. However, the kernel vector γ could produce the undesirable situation $\gamma \cdot \alpha \equiv 0$ and $\gamma \cdot \beta \equiv 0$. In order to prevent this, it seems more prudent to use only $n-1$ related divisors, namely, $\mathsf{D}, [\lambda]\mathsf{D}, \ldots, [\lambda^{n-3}]\mathsf{D}$, and $[\lambda^{n-2}]\mathsf{D}$. In other words, using Ψ^* reduces the cost of relation generation from $T(s)$ to just $\frac{T(s)}{n-1}$.

We can do even better by exploiting Ψ^* to reduce the factor base size from $F(s)$ to $\frac{F(s)}{n}$. Mathematically speaking, we work with the quotient of $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ by the action of Ψ^*, taking a factor base of irreducible $\mathsf{div}(u,v) \in \mathrm{Jac}_H(\mathbb{F}_q)$ with $\deg u \leq s$ consisting of Ψ^*-orbit representatives. (For example, the orbit of $\mathsf{div}(u,v)$ might be represented by $\max\left(\{(\Psi^*)^i(\mathsf{div}(u,v)) \colon 0 \leq i < n-1\}\right)$ with respect to the lexicographic ordering.) In other words, s-smooth divisors factorize in our factor base as $\sum_i [(\lambda^{j_i})^{-1}]D_i$, where $[\lambda^{j_i}]D_i = (\Psi^*)^{j_i}(D_i)$ is in the factor base for some $0 \leq j_i < n$. Using Ψ^*-orbits lets us reduce the costs $T(s)$ and $L(s)$ to $\left(\frac{F(s)}{n}+\epsilon\right)E(s) \approx \frac{T(s)}{n}$ and $d\cdot\left(\frac{F(s)}{n}+\epsilon\right)^2 \approx \frac{L(s)}{n^2}$, respectively.

While the factor-n^2 speed-up in the linear algebra phase is more impressive than the factor-n speed-up in relation generation, linear algebra is not the bottleneck in the entire DLP computation. The overall speed-up mostly corresponds to the speed-up in the relation generation phase, though the reduction in factor-base size (and the resulting reduction in dimension of the linear algebra problem) is still a very welcome improvement in practice.

6.2. Problem instance: Solving discrete logarithms on $\mathcal{E}/\mathbb{F}_{2^{5\times 31}}$

In order to put the analysis above into practice, we solved the DLP on an weak GLS binary curve over $\mathbb{F}_{q^\ell}$ where $q = 2^n$ with $n = 5$ and $\ell = 31$.

Let $\mathbb{F}_q = \mathbb{F}_2[u]/\langle u^5+u^2+1\rangle$ and $\mathbb{F}_{q^\ell} = \mathbb{F}_q[v]/\langle v^{31}+v^3+1\rangle$. The curve

$$\mathcal{E}/\mathbb{F}_{q^\ell} \colon y^2 + x\cdot y = x^3 + x^2 + \left(v^{18}+v^{17}+v^{12}+v^8+v^5+v^4+1\right)$$

satisfies $\#\mathcal{E}\left(\mathbb{F}_{q^\ell}\right) = c \cdot r$ where $r = 3515327356765562060155662043792542 1$ is a 115-bit prime number and $c = 1299222562550$.

To construct a discrete logarithm challenge, we randomly selected an order-r point $\mathsf{P} = (X_\mathsf{P}, Y_\mathsf{P})$ using the `Random()` function of Magma, and we set $\mathsf{P}' = [c]\,(\pi_x, \pi_y)$ where $\pi_x = v^{355}/v^{133} + (v + u + 1)$ and π_y is one of the roots of $y^2 + \pi_x y + \pi_x^3 + \pi_x^2 + \left(v^{18} + v^{17} + v^{12} + v^8 + v^5 + v^4 + 1\right)$. Our goal was to find $1 \le \lambda \le r$ such that $\mathsf{P}' = [\lambda]\mathsf{P}$. Magma code to set up this DLP instance is given in Appendix A.1.

Using the function `WeilDescent()` of Magma, we reduced the problem into a hyperelliptic genus-32 curve $\mathcal{H}/\mathbb{F}_q : y^2 + h(x)y = f(x)$ where

$$\begin{aligned} h(x) &= u^7x^{32} + u^{12}x^{16} + u^{30}x^8 + u^{28}x^2 + u^7x\,, \\ f(x) &= u^4x^{65} + u^{14}x^{64} + u^{14}x^{33} + u^{19}x^{17} + u^{16}x^8 \\ &\qquad + u^{15}x^5 + u^{25}x^4 + u^4x^3 + u^{24}x. \end{aligned}$$

The points P and P' are mapped to divisors D and D', respectively. The translated DLP instance in the Jacobian $\mathcal{H}/\mathbb{F}_q$ is described by the Magma code in Appendix A.2. The endomorphism Ψ^* of $\mathrm{Jac}_\mathcal{H}(\mathbb{F}_q)$ is defined by

$$\Psi^*\colon \mathsf{div}(u,v) \longmapsto \mathsf{div}\Big(\big(u^{21}\big)^{\deg u} \cdot ({}^\sigma u)\Big(\frac{x}{u^{21}}\Big), \big(u^{14}\big) \cdot ({}^\sigma v)\Big(\frac{x}{u^{21}}\Big)\Big).$$

In this setting, we implemented a parallel version of the Enge–Gaudry algorithm in Magma [9]. We successfully accelerated the relation step of the Enge–Gaudry algorithm by using the endomorphism Ψ^* as discussed in §5. The factor base was dynamically built as in [15], using the smoothness bound $s = 4$. The i-th thread of our parallel implementation built its own local factor base as

$$\mathcal{F}_{i,s} = \Big\{ \max\big\{(\Psi^*)^i(\mathsf{div}(u,v))\colon 0 \le i < n\big\} \colon \mathsf{div}(u,v) \text{ irreducible}, \deg u \le s \Big\}.$$

We used 96 cores of 16 Intel Core i7 machines (3.20GHz, 3.40GHz, and 3.47GHz) and 32 cores of two Intel Xeon E5 2.60GHz machines to find our 4-smooth divisors. The linear algebra step was solved with one core of an Intel Xeon E5 2.60GHz machine by using the function `ModularSolution()` that Magma has implemented. We solved the DLP in $\mathrm{Jac}_\mathcal{H}(\mathbb{F}_{2^5})$ in 1034.596 CPU days, finding

$$\gamma = 31651293342165466420895111254857443\,.$$

The Magma implementation of the procedures described here is available at `https://github.com/JJChiDguez/combining_GLS_with_GHS.git`.

6.3. Comparison with related work

Velichka, Jacobson, and Stein [10] report the solution of a discrete logarithm problem for the same elliptic curve $\mathcal{E}/\mathbb{F}_{2^{5\times 31}}$ using hyperelliptic index calculus without the endomorphism technique. Table 1 compares our results with theirs.

The factor base in [10] had 136,533 divisors. Using the endomorphism described here, we reduced this to 27271 $\approx \frac{136533}{5}$ divisors, in line with our theoretical analysis.

	This work	Velichka *et al.* [10]		
		JMS EG	Opt. EG	Vollmer
Relation generation	**1034.572**	*8492.67*	*6338.01*	1720.818
Linear algebra step	**0.024**	*2.470*	*2.800*	14.244
Total	**1034.597**	*8495.650*	*6340.810*	1735.063
Speedup		***8.212***	***6.129***	**1.677**

Table 1: CPU days to solve the DLP on tbe hyperelliptic genus-32 curve $\mathcal{H}/\mathbb{F}_{2^5}$ of §6.2, using index calculus with smoothness bound 4. Values in parentheses are estimates. "JMS EG" and "Opt. EG" are estimates from [10] for the Enge–Gaudry algorithm with the strategy and optimal parameters from [12], and an optimized large-prime variant, respectively. "Vollmer" lists experimental timings from [10] using a sieve-based version of Vollmer's algorithm.

The discrete logarithm in [10] was computed using a sieve-based version of Vollmer's algorithm implemented with the GNU Multi-Precision C library version 4.2.2, Automatically Tuned Linear Algebra Software (ATLAS) version 3.7.31 ([2]), and linbox version 1.1.3 compiled with GCC version 3.4.4 for the linear algebra. Their experiments were run on 152 dual Intel P4 Xeon machines (2.4GHz and 2.8GHz) with 512 kb cache and 2 GB of RAM. We also find estimated timings in [10] for hypothetical computations using the Enge–Gaudry algorithm using parameters derived from [12], and for a large-prime variation.

Remarkably, we managed to produce a faster discrete logarithm attack than the one reported in [10], despite using a non-optimized implementation based on Magma. Due to the more advanced micro-architecture used in our experiments, the speedup achieved by our approach was higher than expected.[7]

The endomorphism $\Psi^*\colon \mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q) \to \mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ can also be used in the sieve-based version of Vollmer's algorithm [10]. Extrapolating the timing costs given in Table 1, we would expect 344.164 and 0.569 CPU-days for the relation generation and linear algebra steps, respectively.

7. Conclusions

We have shown that the GLS endomorphism on $\mathcal{E}/\mathbb{F}_{2^{n\cdot\ell}}$ induces an efficient endomorphism $\Psi^*\colon \mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q) \to \mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$ on the Jacobian of the image of GHS Weil descent applied to $\mathcal{E}/\mathbb{F}_{2^{n\cdot\ell}}$. This endomorphism permits a factor-n speedup over standard index-calculus procedures for solving the DLP on $\mathrm{Jac}_{\mathcal{H}}(\mathbb{F}_q)$. Our analysis is backed up by the explicit computation of a DLP in a prime-order subgroup of a GLS elliptic curve over the field $\mathbb{F}_{2^{5\cdot 31}}$. A Magma implementation of a standard index-calculus procedure boosted with the GLS endomorphism found this discrete logarithm in about $1,035$ CPU-days.

[7] It is worth mentioning that Magma's implementation of Lanczos algorithm takes advantage of both, a more advanced micro-architecture instruction set and a concurrent multi-core computation.

While binary GLS curves offer a tempting speedup for scalar multiplication, our results show that this is tempered by a substantial speedup in DLP computations. This must be taken into account if binary GLS curves are considered for use in cryptographic applications.

Acknowledgement

The authors would like to acknowledge the anonymous referees whose comments and suggestions greatly helped us to improve the manuscript. We thank "Consejo Nacional de Ciencia y Tecnología" (CONACyT) for the scholarship they provided to the first author during the period that he was a Ph.D. candidate at the Computer Science Department of Cinvestav-IPN.

References

[1] S. D. Galbraith, P. Gaudry, Recent progress on the elliptic curve discrete logarithm problem, Des. Codes Cryptography 78 (1) (2016) 51–72.

[2] T. Oliveira, J. C. López-Hernández, D. Cervantes-Vázquez, F. Rodríguez-Henríquez, Koblitz curves over quadratic fields, J. Cryptol. 32 (3) (2019) 867–894.

[3] G. Frey, How to disguise an elliptic curve. Talk at ECC'98, Waterloo, public version available at `https://cr.yp.to/bib/1998/frey-disguise.ps` (1998).

[4] S. D. Galbraith, N. P. Smart, A cryptographic application of Weil descent, in: Cryptography and Coding, 1999, pp. 191–200.

[5] P. Gaudry, F. Hess, N. P. Smart, Constructive and destructive facets of Weil descent on elliptic curves, J. Cryptology 15 (1) (2002) 19–46.

[6] S. D. Galbraith, F. Hess, N. P. Smart, Extending the GHS weil descent attack, in: Advances in Cryptology - EUROCRYPT, 2002, pp. 29–44.

[7] F. Hess, The GHS attack revisited, in: E. Biham (Ed.), Advances in Cryptology - EUROCRYPT 2003, Vol. 2656 of Lecture Notes in Computer Science, Springer, 2003, pp. 374–387.

[8] F. Hess, Generalising the GHS attack on the elliptic curve discrete logarithm problem, LMS Journal of Computation and Mathematics 7 (2004) 167–192.

[9] Magma Computational Algebra System version 2.19-7, available at `http://magma.maths.usyd.edu.au/magma/`.

[10] M. D. Velichka, M. J. Jacobson Jr, A. Stein, Computing discrete logarithms in the Jacobian of high-genus hyperelliptic curves over even characteristic finite fields, Math. Comput. 83 (286) (2014) 935–963.

[11] A. Menezes, M. Qu, Analysis of the Weil Descent Attack of Gaudry, Hess and Smart, in: Topics in Cryptology - CT-RSA, 2001, pp. 308–318.

[12] M. Jacobson, A. Menezes, A. Stein, Solving elliptic curve discrete logarithm problems using Weil descent, J. Ramanujan Math. Soc. 16 (3) (2001) 231–260.

[13] M. Maurer, A. Menezes, E. Teske, Analysis of the GHS weil descent attack on the ECDLP over characteristic two finite fields of composite degree, in: Progress in Cryptology - INDOCRYPT, 2001, pp. 195–213.

[14] D. Hankerson, K. Karabina, A. Menezes, Analyzing the Galbraith-Lin-Scott point multiplication method for elliptic curves over binary fields, IEEE Trans. Computers 58 (10) (2009) 1411–1420.

[15] J. Chi, T. Oliveira, Attacking a binary GLS elliptic curve with magma, in: Progress in Cryptology - LATINCRYPT, 2015, pp. 308–326.

[16] S. D. Galbraith, R. Granger, S. Merz, C. Petit, On index calculus algorithms for subfield curves, IACR Cryptol. ePrint Arch. 2020 (2020) 1315.
URL https://eprint.iacr.org/2020/1315

[17] A. J. Menezes, Y.-H. Wu, R. J. Zuccherato, An elementary introduction to hyperelliptic curves, Appendix in [29] (1996).

[18] H. Cohen, G. Frey, R. Avanzi, C. Doche, T. Lange, K. Nguyen, F. Vercauteren (Eds.), Handbook of Elliptic and Hyperelliptic Curve Cryptography, Chapman and Hall/CRC, 2005.

[19] S. D. Galbraith, Mathematics of Public Key Cryptography, 1st Edition, Cambridge University Press, New York, NY, USA, 2012, public version 2.0 available at https://www.math.auckland.ac.nz/~sgal018/crypto-book/main.pdf.

[20] L. C. Washington, Elliptic Curves: Number Theory and Cryptography, Second Edition, 2nd Edition, Chapman & Hall/CRC, 2008.

[21] S. D. Galbraith, X. Lin, M. Scott, Endomorphisms for Faster Elliptic Curve Cryptography on a Large Class of Curves, J. Cryptology 24 (3) (2011) 446–469.

[22] R. P. Gallant, R. J. Lambert, S. A. Vanstone, Faster point multiplication on elliptic curves with efficient endomorphisms, in: Advances in Cryptology - CRYPTO, 2001, pp. 190–200.

[23] A. U. Ay, C. Mancillas-López, E. Öztürk, F. Rodríguez-Henríquez, E. Savas, Constant-time hardware computation of elliptic curve scalar multiplication around the 128 bit security level, Microprocess. Microsystems 62 (2018) 79–90.

[24] T. Oliveira, D. F. Aranha, J. López, F. Rodríguez-Henríquez, Improving the performance of the GLS254, Presentation at CHES 2016 rump session (2016).

[25] T. Oliveira, J. López, D. F. Aranha, F. Rodríguez-Henríquez, Two is the fastest prime: lambda coordinates for binary elliptic curves, J. Cryptographic Engineering 4 (1) (2014) 3–17.

[26] P. Gaudry, An algorithm for solving the discrete log problem on hyperelliptic curves, in: B. Preneel (Ed.), Advances in Cryptology - EUROCRYPT 2000, Vol. 1807 of Lecture Notes in Computer Science, Springer, 2000, pp. 19–34.

[27] A. Enge, P. Gaudry, A general framework for subexponential discrete logarithm algorithms, Acta Arithmetica 102 (1) (2002) 83–103.
URL `http://eudml.org/doc/278301`

[28] P. Gaudry, E. Thomé, N. Thériault, C. Diem, A double large prime variation for small genus hyperelliptic index calculus, Math. Comput. 76 (257) (2007) 475–492.

[29] N. Koblitz, Algebraic Aspects of Cryptography, Springer, 1998.

Appendix A. Magma codes

Appendix A.1. Elliptic curve instances: EC_instance.mag

```
n := 5; l := 31; q := 2^n; N := 2^l;
F_2 := GF(2); P_2<t> := PolynomialRing(F_2);

F_q<u> := ext<F_2| t^5 + t^2 + 1>;
F_qn<v>:= ext<F_q| t^31 + t^3 + 1>;

a_qn := F_qn!1; b_qn := v^18 + v^17 + v^12 + v^8 + v^5 + v^4 + 1;
E_qn := EllipticCurve([F_qn| 1, a_qn, 0, 0, b_qn]);
c := 1299222562550; r := 3515327356765562060155662043792542 1;

Pt_x := F_qn![ u^10, u^30, u^24, u^17, u^26, u^23, u^22, u^8,
u^4, u^25, u^24, u^19, 0, u^30, u^2, u^8, u^24, u^16, u^21,
u^19, u^3, u^2, u^21, u^7, u^11, u^4, u^23, u^13, u^3, u^23, u^23 ];
Pt_y := F_qn![ u^25, u^29, u^16, u^20, 0, 1, u^10, u^6, u^13,
u^30, u^8, u^30, u^9, u^9, 0, u^9, u^8, u^28, u^21, u^23, u^23,
u^16, u^27, u^22, u^8, u^4, u^8, u^12, u^17, u^7, u^9 ];
Pt := E_qn![Pt_x, Pt_y];

Pt_prime_x := v^355/v^133 + (v+u+1);
Pt_prime_y := F_qn![ u^15, u^12, u^12, 1, u^15, u^22, u^16, 0,
u^17, u^3, u^19, u^10, u^9, u^25, u^18, u^23, u^13, u^9, u^12,
u^22, u^30, u^17, u^15, u^22, u^2, u^22, u^21, u^16, u^13, u^7, u^20 ];
Pt_prime := c*E_qn![Pt_prime_x, Pt_prime_y];
```

Appendix A.2. Hyperelliptic curve instances: HEC_instance.mag

```
P_q<w> := PolynomialRing(F_q);
h_q := u^7*w^32 + u^12*w^16 + u^30*w^8 + u^28*w^2 + u^7*w;
f_q := u^4*w^65 + u^14*w^64 + u^14*w^33 + u^19*w^17 + u^16*w^8
+ u^15*w^5 + u^25*w^4 + u^4*w^3 + u^24*w;

H_q := HyperellipticCurve(f_q, h_q);
J_q := Jacobian(H_q);

D_x := P_q![ u^9, u^18, u^28, u^3, u^29, u^21, u^17, u^19, u^26,
u^16, u^8, u^25, u^11, u^8, u^5, u^18, 0, u^2, u^21, u^3, u^28,
u^19, u^22, u^14, u^24, u^6, u^28, u^19, u^16, u^21, u^20, u^18, 1 ];
D_y := P_q![ u^4, u^24, 0, u^2, u^20, u^18, u^30, u, u^6, u^6,
u^27, u^29, u^14, u^29, u^17, u^10, u^12, u^23, u^11, u^3, u^12,
u^11, u^9, u^14, u^30, u^25, u^6, 0, u^5, u^2, u^29, u^25 ];
D := J_q![D_x, D_y];

D_prime_x := P_q![ u^19, u^8, u^23, u^7, u^26, 0, u^2, u^4, u^21,
u^12, u^17, u^20, u^22, u^2, u^5, u^17, u, u^27, u^28, u^16, u^6,
u^18, u^5, u^27, u^19, u^15, u^11, u^14, u^8, u^6, u^26, u^11, 1 ];
D_prime_y := P_q![ u^2, u^24, u^21, u^13, u^10, u^17, 1, u^15,
u^29, u^3, u^16, u^4, u, u^17, u^13, u^22, u^26, u^18, u^8, u^16,
u^21, u^26, u, u^16, u^16, u^3, u^5, u^24, u^26, u^26, u^14, u^14];
D_prime := J_q![D_prime_x, D_prime_y];
```

Appendix A.3. Testing the solution: checking_dlog.mag

```
load "EC_instance.mag";
load "HEC_instance.mag";

dLog := 0x618877C96DE350E8C7980393356E3;
(Pt * dLog) eq Pt_prime; (D * dLog) eq D_prime;
```