On the Discrete Logarithm Problem on Algebraic Tori *

R. Granger[1] and F. Vercauteren[2]

[1] University of Bristol, Department of Computer Science,
Merchant Venturers Building, Woodland Road,
Bristol, BS8 1UB, United Kingdom
granger@cs.bris.ac.uk

[2] Department of Electrical Engineering
University of Leuven
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
frederik.vercauteren@esat.kuleuven.ac.be

Abstract. Using a recent idea of Gaudry and exploiting rational representations of algebraic tori, we present an index calculus type algorithm for solving the discrete logarithm problem that works directly in these groups. Using a prototype implementation, we obtain practical upper bounds for the difficulty of solving the DLP in the tori $T_2(\mathbb{F}_{p^m})$ and $T_6(\mathbb{F}_{p^m})$ for various p and m. Our results do not affect the security of the cryptosystems LUC, XTR, or CEILIDH over prime fields. However, the practical efficiency of our method against other methods needs further examining, for certain choices of p and m in regions of cryptographic interest.

1 Introduction

The first instantiation of public key cryptography, the Diffie-Hellman key agreement protocol [5], was based on the assumption that discrete logarithms in finite fields are hard to compute. Since then, the discrete logarithm problem (DLP) has been used in a variety of cryptographic protocols, such as the signature and encryption schemes due to ElGamal [6] and its variants. During the 1980's, these schemes were formulated in the full multiplicative group of a finite field $\mathbb{F}_p$. To speed-up exponentiation and obtain shorter signatures, Schnorr [24] proposed to work in a small prime order subgroup of the multiplicative group $\mathbb{F}_p^\times$ of a prime finite field. Most modern DLP-based cryptosystems, such as the Digital Signature Algorithm (DSA) [9], follow Schnorr's idea.

Lenstra [15] showed that by working in a prime order subgroup G of $\mathbb{F}_{p^m}^\times$, for extensions that admit an optimal normal basis, one can obtain a further

* The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT. The information in this document reflects only the authors' views, is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

speed-up. Furthermore, Lenstra proved that when $|G| \mid \Phi_m(p)$ with $\Phi_m(x)$ the m-th cyclotomic polynomial and $|G| > m$, the minimal surrounding field of G truly is $\mathbb{F}_{p^m}$ and not a proper subfield. Lacking any knowledge to the contrary, the security of this cryptosystem has been based on two assumptions: firstly, the group G should be large enough such that square root algorithms [18] are infeasible and secondly, the minimal finite field in which G embeds should be large enough to thwart index calculus type attacks [18]. In these attacks one does not make any use of the particular form of the minimal surrounding finite field, i.e., $\mathbb{F}_{p^n}$, but only its size and the size of the subgroup of cryptographic interest.

More recent proposals, such as LUC [25], XTR [16] and CEILIDH [22], improve upon Schnorr's and Lenstra's idea, the latter two working in a subgroup $G \subset \mathbb{F}_{q^6}^{\times}$ with $|G| \mid \Phi_6(q) = q^2 - q + 1$, where q is a prime power. Brouwer, Pellikaan and Verheul [2] were the first to give a cryptographic application of effectively representing elements in G using only two $\mathbb{F}_q$-elements, instead of six, effectively reducing the communication cost by a factor of three.

Rubin and Silverberg [22] showed how to interpret and generalise the above cryptosystems using the algebraic torus $T_n(\mathbb{F}_q)$ which is isomorphic to the subgroup $G_{q,n} \subset \mathbb{F}_{q^n}^{\times}$ of order $\Phi_n(q)$. For "rational" tori, elements of $T_n(\mathbb{F}_q)$ can be compactly represented by $\varphi(n)$ elements of $\mathbb{F}_q$, obtaining a compression factor of $n/\varphi(n)$ over the field representation.

In this paper we develop an index calculus algorithm that works directly on rational tori $T_n(\mathbb{F}_q)$ and consequently show that the hardness of the DLP can depend on the form of the minimal surrounding finite field. The algorithm is based on the purely algebraic index calculus approach by Gaudry [10] and exploits the compact representation of elements of rational tori. The very existence of such an algorithm shows that the lower communication cost offered by these tori, may also be exploited by the cryptanalyst.

In practice, the DLP in T_2 and T_6 are most important, since they determine the security of the cryptosystems LUC [25], XTR [16], CEILIDH [22], and MNT curves [19]. We stress that when defined over prime fields $\mathbb{F}_p$, the security of these cryptosystems is not affected by our algorithm. Over extension fields however, this is not always the case. In this paper, we provide a detailed description of our algorithm for $T_2(\mathbb{F}_{q^m})$ and $T_6(\mathbb{F}_{q^m})$. Note that this includes precisely the systems presented in [17], and also those described in [28, 27] via the inclusion of $T_n(\mathbb{F}_p)$ in $T_2(\mathbb{F}_{p^{n/2}})$ and $T_6(F_{p^{n/6}})$ when n is divisible by two or six, respectively, which for efficiency reasons is always the case. Our method is fully exponential for fixed m and increasing q. From a complexity theoretic point of view, it is noteworthy that for certain very specific combinations of q and m, for example when $m! \approx q$, the algorithms run in expected time $L_{q^m}(1/2, c)$, which is comparable to the index calculus algorithm by Adleman and DeMarrais [1]. However, our focus will be on parameter ranges of practical cryptographic interest rather than asymptotic results.

A complexity analysis and prototype implementation of these algorithms, show that they are faster than Pollard-Rho in the *full* torus $T_2(\mathbb{F}_{q^m})$ for $m \geq 5$

and in the *full* torus $T_6(\mathbb{F}_{q^m})$ for $m \geq 3$. However, in cryptographic applications one would work in a prime order subgroup of $T_n(\mathbb{F}_{q^m})$ of order around 2^{160}; in this case, our algorithm is only faster than Pollard-Rho for larger m.

From a practical perspective, our experiments show that in the cryptographic range, the algorithm for $T_6(\mathbb{F}_{q^m})$ outperforms the corresponding algorithm for $T_2(\mathbb{F}_{q^{3m}})$ and that it is most efficient when $m = 4$ or $m = 5$. Furthermore, for $m = 5$, both algorithms in practice outperform Pollard-Rho in a subgroup of $T_6(\mathbb{F}_{q^5})$ of order 2^{160}, for q^{30} up to and including the 960-bit scheme based in $T_{30}(\mathbb{F}_p)$ proposed in [27]. Compared to Pollard ρ our method seems to achieve in practice a 1000 fold speedup; its practical comparison with Adleman-DeMarrais is yet to be explored. Our experiments show that it is currently feasible to solve the DLP in $T_{30}(\mathbb{F}_p)$ with $\lceil \log_2 p \rceil = 20$, where we assume that a computation of around 2^{45} seconds is feasible.

The remainder of this paper is organised as follows. In Section 2 we briefly review algebraic tori and the notion of rationality. In Section 3 we present the philosophy of our algorithm and explain how it is related to classical index calculus algorithms. In Sections 4 and 5 we give a detailed description of the algorithm for $T_2(\mathbb{F}_{q^m})$ and $T_6(\mathbb{F}_{q^m})$ respectively. Finally, we conclude and give pointers for further research in Section 6.

2 Discrete Logs in Extension Fields and Algebraic Tori

Extension fields possess a richer algebraic structure than prime fields, in particular those with highly composite extension degrees. This has led some researchers to suspect that such fields may be cryptographically weak. For instance, in 1984 Odlyzko stated that fields with a composite extension degree 'may be very weak' [21]. The main result of this paper shows that these concerns may indeed be valid. A naive attempt to exploit the available subfield structure of extension fields in solving discrete logarithms, naturally leads one to consider the DLP on algebraic tori, as we show below.

2.1 A simple reduction of the DLP

Let $k = \mathbb{F}_q$ and let $K = \mathbb{F}_{q^n}$ be an extension of k of degree $n > 1$. Assume that $g \in K$ is a generator of $K^\times$ and let $h = g^s$ with $0 \leq s < q^n - 1$ be an element we wish to find the discrete logarithm of with respect to g.

Then by applying to g and h the norm maps N_{K/k_d} with respect to each intermediate subfield k_d of K, and solving the resulting discrete logarithms in these subfields, a simple argument shows that one can determine $s \bmod \mathrm{lcm}\{\Phi_d(q)\}_{d|n, d \neq n}$, where $\Phi_d(q)$ is the d-th cyclotomic polynomial evaluated at q. Modulo a cryptographically negligible factor, the remaining modular information required to determine the full discrete logarithm comes from the order $\Phi_n(q)$ subgroup of $K^\times$. As observed by Rubin and Silverberg [22], this subgroup is precisely the algebraic torus $T_n(\mathbb{F}_q)$.

2.2 The algebraic torus

In their CRYPTO 2003 paper [22], Rubin and Silverberg introduced the notion of torus-based cryptography. Their central idea was to interpret the subgroups of $K^\times$ as algebraic tori, and by exploiting birational maps from these groups to affine space, they obtained an efficient compression mechanism for elements of extension fields. Along with the existing public key cryptosystems XTR [16] and LUC [25], their method provides a reduction in bandwidth requirements for finite field discrete logarithm based protocols, which is becoming increasingly relevant as key-size recommendations become larger in order to maintain security levels.

Definition 1. *Let $k = \mathbb{F}_q$ and let $K = \mathbb{F}_{q^n}$ be an extension of k of degree $n > 1$. We define the algebraic torus $T_n(\mathbb{F}_q)$ as*

$$T_n(\mathbb{F}_q) = \{\alpha \in K \mid N_{K/k_d}(\alpha) = 1 \textit{ for all subfields } k \subseteq k_d \subsetneq K\}.$$

Strictly speaking, $T_n(\mathbb{F}_q)$ refers only to the $\mathbb{F}_q$-rational points on the affine algebraic variety T_n, rather than the torus itself (see [22] for the exact construction).

Note that since $T_n(\mathbb{F}_q)$ is simply a subgroup of $\mathbb{F}_{q^n}^\times$, the group operation can be realised as ordinary multiplication in the field $\mathbb{F}_{q^n}$. The dimension of the variety T_n is $\phi(n) = \deg(\Phi_n(x))$, with $\phi(\cdot)$ the Euler totient function.

Let $G_{q,n}$ denote the subgroup of $\mathbb{F}_{q^n}^\times$ of order $\Phi_n(q)$. The following lemma from [22] provides some useful properties of T_n.

Lemma 1.

1. *$T_n(\mathbb{F}_q) \cong G_{q,n}$ and hence $\#T_n(\mathbb{F}_q) = \Phi_n(q)$.*
2. *If $h \in T_n(\mathbb{F}_q)$ is an element of prime order not dividing n, then h does not lie in a proper subfield of $\mathbb{F}_{q^n}/\mathbb{F}_q$.*

It follows that $T_n(\mathbb{F}_q)$ may be regarded as the 'primitive' subgroup of $\mathbb{F}_{q^n}^\times$, since by Lemma 1 it does not embed into a proper subfield. Hence in practice, one always uses a subgroup of $T_n(\mathbb{F}_q)$ in cryptographic applications, since otherwise a given DLP embeds into a proper subfield of $\mathbb{F}_{q^n}$ (see also [15]). In fact, using the decomposition

$$x^n - 1 = \prod_{d|n} \Phi_d(x)$$

in $\mathbb{Z}[x]$, the group $\mathbb{F}_{q^n}^\times$ can be seen to be almost the same as the direct product $\prod_{d|n} T_n(\mathbb{F}_q)$. Hence finding an efficient algorithm to solve the DLP on algebraic tori enables one to solve DLPs in extension fields, as well as vice versa.

2.3 Rationality of tori over $\mathbb{F}_q$

In order to compress elements of the variety T_n, we make use of rationality, for particular values of n. The rationality of T_n means there exists a birational map from T_n to $\phi(n)$-dimensional affine space $\mathbb{A}^{\phi(n)}$. This allows one to represent nearly all elements of $T_n(\mathbb{F}_q)$ with just $\phi(n)$ elements of $\mathbb{F}_q$, providing an effective compression factor of $n/\phi(n)$ over the embedding of $T_n(\mathbb{F}_q)$ into $\mathbb{F}_{q^n}$. Since T_n has dimension $\phi(n)$, this compression factor is optimal. T_n is known to be rational when n is either a prime power, or is a product of two prime powers, and is conjectured to be rational for all n [22].

Formally, rationality can be defined as follows.

Definition 2. *Let T_n be an algebraic torus over $\mathbb{F}_q$ of dimension $d = \phi(n)$, then T_n is said to be* rational *if there is a birational map $\rho : T_n \to \mathbb{A}^{\phi(n)}$ defined over $\mathbb{F}_q$.*

This means that there are subsets $W \subset T_n$ and $U \subset \mathbb{A}^{\phi(n)}$, and rational functions $\rho_1, \ldots, \rho_{\phi(n)} \in \mathbb{F}_q(x_1, \ldots, x_n)$ and $\psi_1, \ldots, \psi_n \in \mathbb{F}_q(y_1, \ldots, y_{\phi(n)})$ such that $\rho = (\rho_1, \ldots, \rho_{\phi(n)}) : W \to U$ and $\psi = (\psi_1, \ldots, \psi_n) : U \to W$ are inverse isomorphisms. Furthermore, the differences $T \setminus W$ and $\mathbb{A}^{\phi(n)} \setminus U$ should be algebraic varieties of dimension $\leq (d-1)$, which implies that W (resp. U) is 'almost the whole' of T (resp. $\mathbb{A}^{\phi(n)}$).

The public key cryptosystem CEILIDH [22] is based on the algebraic torus T_6, which achieves a compression factor of three over the extension field representation. Rationality whilst useful, is not essential, since Van Dijk and Woodruff [28] showed that one can obtain key-agreement, signature and encryption schemes with bandwidth compressed by this factor asymptotically with the number of keys/signatures/messages, without relying on the conjecture stated above. Indeed, their result applies to any torus T_n, which helps explain the recent and increasing interest in torus-based cryptography.

3 Algorithm Philosophy

The algorithm as presented in Sections 4 and 5 is based on an idea first proposed by Gaudry [10], in reference to the DLP on general abelian varieties. While Gaudry's method is in principle an index calculus algorithm, the ingredients are very algebraic: for instance one need not rely on unique factorisation to obtain a notion of 'smoothness', as in finite field discrete logarithm algorithms.

As an introduction, in this section we consider Gaudry's idea in the context of computing discrete logarithms in $\mathbb{F}_{q^m}^{\times}$, and show how it is related to classical index calculus.

3.1 Classical method

Let $\mathbb{F}_{q^m} = \mathbb{F}_q[t]/(f(t))$ for some monic irreducible degree m polynomial and let the basis be $\{1, t, \ldots, t^{m-1}\}$. Let g be a generator of $\mathbb{F}_{q^m}^{\times}$ and let $h \in \langle g \rangle$ be an element we are to compute the logarithm of w.r.t. g. Suppose also, for this example, that we are able to deal with a factor base of size q.

Classically, one would first reduce the problem to considering only monic polynomials, i.e., one considers the quotient $\mathbb{F}_{q^m}^{\times}/\mathbb{F}_q^{\times}$, and defines a factor base

$$\mathcal{F} = \{t + a : a \in \mathbb{F}_q\}.$$

Then for random $j, k \in \mathbb{Z}/((q^m - 1)/(q - 1))\mathbb{Z}$ one computes $r = g^j h^k$ and tests whether $r/\mathrm{lc}(r)$ decomposes over $\mathcal{F}$, with $\mathrm{lc}(r)$ the leading coefficient of r. This occurs with probability approximately $1/(m - 1)!$ for large q since the set of all products of $m - 1$ elements of $\mathcal{F}$ generates roughly $q^{m-1}/(m - 1)!$ elements of $\mathbb{F}_{q^m}^{\times}/\mathbb{F}_q^{\times}$.

Computing more than q such relations allows one to compute $\log_g h \bmod (q^m - 1)/(q - 1)$ as usual with a linear algebra elimination (and one applies the norm $N_{\mathbb{F}_{q^m}/\mathbb{F}_q}$ to g and h and solves the corresponding DLP in $\mathbb{F}_q^{\times}$ to recover the remaining modular information).

3.2 Gaudry's method

Two essential points taken for granted in the above description are that there exist efficient procedures to compute:

- whether a given r decomposes over $\mathcal{F}$; this happens precisely when $r \in \mathbb{F}_q[t]$ splits over $\mathbb{F}_q$ or equivalently when $\gcd(t^q - t, r/\mathrm{lc}(r)) = r/\mathrm{lc}(r)$,
- the actual decomposition of r, i.e., to compute the roots of $r \in \mathbb{F}_q[t]$ in $\mathbb{F}_q$.

One may equivalently consider the following problem: determine whether the system of equations obtained by equating powers of t in the equality

$$\prod_{i=1}^{m-1}(t + a_i) = r/\mathrm{lc}(r) = r_0 + r_1 t + \cdots + r_{m-2}t^{m-2} + t^{m-1}, \tag{1}$$

has a solution $(a_1, \ldots, a_{m-1}) \in \mathbb{F}_q^{m-1}$ and if so, to compute one such solution. Of course, in this trivial example the roots a_i can be read off from the factorisation of $r/\mathrm{lc}(r)$. However, one obtains a non-trivial example if the group operation on the left is more sophisticated than polynomial multiplication, such as elliptic curve point addition, which was Gaudry's original motivation for developing the algorithm. In this case the decomposition of a group element over the factor base can become more sophisticated, but the principle remains the same.

The central benefit of this perspective is that it can be applied in the absence of unique factorisation, since with a suitable choice of factor base, or more accurately a decomposition base, one can simply induce relations algebraically. For example, approaching the above problem from this slightly different perspective gives an algorithm for working directly in $\mathbb{F}_{q^m}^{\times}$, which is perhaps more natural than the stated quotient, $\mathbb{F}_{q^m}^{\times}/\mathbb{F}_q^{\times}$. Define a decomposition base

$$\mathcal{F} = \{1 + at : a \in \mathbb{F}_q\},$$

and again associate to the equality

$$\prod_{i=1}^{m}(1 + a_i t) \equiv r \equiv r_0 + r_1 t + \cdots + r_{m-1}t^{m-1} \pmod{f(t)}, \tag{2}$$

the algebraic system obtained by equating powers of t.

Note that in (2) one must multiply m elements of $\mathcal{F}$ in order to obtain a probability of $1/m!$ for obtaining a relation, rather than the $m-1$ elements (and probability $1/(m-1)!$) of (1). The reason these probabilities differ is simply that the algebraic groups $\mathbb{F}_{q^m}^\times/\mathbb{F}_q^\times$ and $\mathbb{F}_{q^m}^\times$ over $\mathbb{F}_q$ are $m-1$ and m-dimensional respectively.

Ignoring for the moment that $\mathcal{F}$ essentially consists of degree one polynomials, and assuming that we want to solve this system without factoring $r/\mathrm{lc}(r)$, we are faced with finding a solution to a non-linear system, which would ordinarily require a Gröbner basis computation to solve. However writing out the left hand side in the polynomial basis $\{1, \dots, t^{m-1}\}$ gives

$$\begin{aligned}\prod_{i=1}^{m}(1 + a_i t) &= 1 + \overline{\sigma}_1 t + \cdots + \overline{\sigma}_m t^m \\ &\equiv 1 + \overline{\sigma}_1 t + \cdots + \overline{\sigma}_{m-1} t^{m-1} + \overline{\sigma}_m (t^m - f(t)) \pmod{f(t)},\end{aligned}$$

with $\overline{\sigma}_i$ the i-th elementary symmetric polynomial in the a_i. Equating powers of t then gives a linear system of equations in the $\overline{\sigma}_i$ for $i = 1, \dots, m$. Given a solution $(\sigma_1, \dots, \sigma_m)$ to this system of equations, r will decompose over $\mathcal{F}$ precisely when the polynomial

$$p(x) := x^m - \sigma_1 x^{m-1} + \sigma_2 x^{m-2} - \cdots + (-1)^m \sigma_m$$

splits over $\mathbb{F}_q$. Thus exploiting the symmetry in the construction of the algebraic system makes solving it much simpler. Although in this contrived example, solving the system directly and solving it using its symmetry are essentially the same, in general the latter makes infeasible computations feasible.

Following from this example, a simple observation is that for an algebraic group over $\mathbb{F}_q$ whose representation is m-dimensional, then using a decomposition base $\mathcal{F}$ of q elements, one must multiply m elements of $\mathcal{F}$ to obtain a constant probability of decomposition $1/m!$. Therefore, we conclude that the more efficient the representation of the group is, the higher the probability of obtaining a relation, and thus the corresponding index calculus algorithm will be more efficient.

In the following two sections, we apply this idea to rational representations of algebraic tori, and show that the above probability of $1/m!$ can be reduced significantly to $1/(m/2)!$ when m is divisible by 2 and to $1/(m/3)!$ when m is divisible by 6.

4 An Index Calculus Algorithm for $T_2(\mathbb{F}_{q^m}) \subset \mathbb{F}_{q^{2m}}^\times$

For q any odd prime power, we describe an algorithm to compute discrete logarithms in $T_2(\mathbb{F}_{q^m})$.

4.1 Setup

With regard to the extension $\mathbb{F}_{q^{2m}}/\mathbb{F}_{q^m}$, by Lemma 1 we know that

$$\#T_2(\mathbb{F}_{q^m}) = \Phi_2(q^m) = q^m + 1,$$

and hence we presume the DLP we consider is in the subgroup of this order. By applying the reduction of the DLP via norms as in Section 2, it is clear that the hard part actually is $T_{2m}(\mathbb{F}_q) \subsetneq T_2(\mathbb{F}_{q^m})$. Since in this section we use the properties of T_2 rather than T_{2m}, we only consider $T_2(\mathbb{F}_{q^m})$, or more accurately $(\mathrm{Res}_{\mathbb{F}_{q^m}/\mathbb{F}_q} T_2)(\mathbb{F}_q)$, where here Res denotes the Weil restriction of scalars (see also [22]).

Let $\mathbb{F}_{q^m} \cong \mathbb{F}_q[t]/(f(t))$ with $f(t) \in \mathbb{F}_q[t]$ an irreducible monic polynonmial of degree m and take the polynomial basis $\{1, t, \ldots, t^{m-1}\}$. Assuming that q is an odd prime power, we let $\mathbb{F}_{q^{2m}} = \mathbb{F}_{q^m}[\gamma]/(\gamma^2 - \delta)$ with basis $\{1, \gamma\}$, for some non-square $\delta \in \mathbb{F}_{q^m} \setminus \mathbb{F}_q$. Then using Definition 1, we see that

$$T_2(\mathbb{F}_{q^m}) = \{(x, y) \in \mathbb{F}_{q^m} \times \mathbb{F}_{q^m} : x^2 - \delta y^2 = 1\}.$$

This representation uses two elements of $\mathbb{F}_{q^m}$ to represent each point. The torus T_2 is one-dimensional, rational, and has the following equivalent affine representation:

$$T_2(\mathbb{F}_{q^m}) = \left\{ \frac{z - \gamma}{z + \gamma} : z \in \mathbb{F}_{q^m} \right\} \cup \{O\}, \tag{3}$$

where O is the point at infinity.

Here a point $g = g_0 + g_1\gamma \in T_2(\mathbb{F}_{q^m})$ in the $\mathbb{F}_{q^{2m}}$ representation has a corresponding representation as given above by the rational function $z = -(1 + g_0)/g_1$ if $g_1 \neq 0$, whilst the elements -1 and 1 map to $z = 0$ and $z = O$ respectively. The representation (3) thus gives a compression factor of two for the elements of $\mathbb{F}_{q^{2m}}$ that lie in $T_2(\mathbb{F}_{q^m})$. Furthermore since $T_2(\mathbb{F}_{q^m})$ has $q^m + 1$ elements, this compression is optimal (since for this example, including the point at infinity, we really have a map from $T_2(\mathbb{F}_{q^m}) \to \mathbb{P}^1(\mathbb{F}_{q^m})$).

4.2 Decomposition base

As with any index calculus algorithm, we need to define a factor base, or in the case of Gaudry's algorithm, a decomposition base. Let

$$\mathcal{F} = \left\{ \frac{a - \gamma}{a + \gamma} : a \in \mathbb{F}_q \right\} \subset T_2(\mathbb{F}_{q^m}),$$

which contains q elements, since the map, given above, is a birational isomorphism from T_2 to $\mathbb{A}^1$. Note that if $\delta \in \mathbb{F}_q$, then $\mathcal{F}$ would lie in the subvariety $T_2(\mathbb{F}_q)$ and would not aid in our attack, which is why we ensured that $\delta \in \mathbb{F}_{q^m} \setminus \mathbb{F}_q$ during the setup.

4.3 Relation finding

Writing the group operation additively, let P be a generator, and let $Q \in \langle P \rangle$ be a point we wish to find the discrete logarithm of with respect to P. For a given $R = [j]P + [k]Q$, we test whether it decomposes as a sum of m points in the decomposition base:

$$P_1 + \cdots + P_m = R, \tag{4}$$

with $P_1, \ldots, P_m \in \mathcal{F}$. From the representation we have chosen for T_2 we may equivalently write this as

$$\prod_{i=1}^{m} \left(\frac{a_i - \gamma}{a_i + \gamma} \right) = \frac{r - \gamma}{r + \gamma},$$

where the a_i are unknown elements in $\mathbb{F}_q$, and $r \in \mathbb{F}_{q^m}$ is the affine representation of R. Note that the left hand side is symmetric in the a_i. Upon expanding the product for both the numerator and denominator, we obtain two polynomials of degree m in γ whose coefficients are just plus or minus the elementary symmetric polynomials $\sigma_i(a_1, \ldots, a_m)$ of the a_i:

$$\frac{\sigma_m - \sigma_{m-1}\gamma + \cdots + (-1)^m \gamma^m}{\sigma_m + \sigma_{m-1}\gamma + \cdots + \gamma^m} = \frac{r - \gamma}{r + \gamma}.$$

Therefore, when we reduce modulo the defining polynomial of γ, we obtain an equation of the form

$$\frac{b_0(\sigma_1, \ldots, \sigma_m) - b_1(\sigma_1, \ldots, \sigma_m)\gamma}{b_0(\sigma_1, \ldots, \sigma_m) + b_1(\sigma_1, \ldots, \sigma_m)\gamma} = \frac{r - \gamma}{r + \gamma},$$

where b_0, b_1 are linear in the σ_i and have coefficients in $\mathbb{F}_{q^m}$. More explicitly, since $\gamma^2 = \delta \in \mathbb{F}_{q^m}$, these polynomials are given by

$$b_0 = \sum_{k=0}^{\lfloor m/2 \rfloor} \sigma_{m-2k}\delta^k \quad \text{and} \quad b_1 = \sum_{k=0}^{\lfloor (m-1)/2 \rfloor} \sigma_{m-2k-1}\delta^k \, ,$$

where we define $\sigma_0 = 1$.

In order to obtain a simple set of algebraic equations amongst the σ_i, we first reduce the left hand side to the affine representation (3) and obtain the equation

$$b_0(\sigma_1, \ldots, \sigma_m) - b_1(\sigma_1, \ldots, \sigma_m) r = 0.$$

Since the unknowns σ_i are elements of $\mathbb{F}_q$, we express the above equation on the polynomial basis of $\mathbb{F}_{q^m}$ to obtain m linear equations over $\mathbb{F}_q$ in the m unknowns $\sigma_i \in \mathbb{F}_q$. This gives an $m \times m$ matrix M over $\mathbb{F}_q$ such that

- the $(m - 2k)$-th column contains the coefficients of δ^k,
- the $(m - 2k - 1)$-th column contains the coefficients of $-r\delta^k$.

Furthermore, let V be the $m \times 1$ vector containing the coefficients of $r\delta^{(m-1)/2}$ when m is odd or $-\delta^{m/2}$ when m is even, then $\Sigma = (\sigma_1, \ldots, \sigma_m)^T$ is a solution of the linear system of equations

$$M\Sigma = V \, .$$

If there is a solution Σ, to see whether this corresponds to a solution of (4) we test whether the polynomial

$$p(x) := x^m - \sigma_1 x^{m-1} + \sigma_2 x^{m-2} - \cdots + (-1)^m \sigma_m$$

splits over $\mathbb{F}_q$ by computing $g(x) := \gcd(x^q - x, p(x))$. If $g(x) = p(x)$, then the roots $a_1, \ldots, a_m$ will be the affine representation of the elements of the factor base which sum to R and we have found a relation.

4.4 Complexity analysis and experiments

The number of elements of $T_2(\mathbb{F}_{q^m})$ generated by all sums of m points in $\mathcal{F}$ is roughly $q^m/m!$, assuming no repeated summands and that most points admit a unique factorisation over the factor base. Hence the probability of obtaining a relation is approximately $1/m!$. Therefore in order to obtain q relations we must perform roughly $m!q$ such decompositions. Each decomposition consists of the following steps:

- computing the matrix M and vector V takes $O(m^3)$ operations in $\mathbb{F}_q$, using a naive multiplication routine,
- solving for $\varSigma$ also requires $O(m^3)$ operations in $\mathbb{F}_q$,
- computing the polynomial $g(x)$ requires $O(m^2 \log q)$ operations in $\mathbb{F}_q$,
- if the polynomial $p(x)$ splits over $\mathbb{F}_q$, then we have to find the roots $a_1, \dots, a_m$ which requires $O(m^2 \log m(\log q + \log m))$ operations in $\mathbb{F}_q$.

Note that the last step only has to be executed $O(q)$ times. The overall complexity to find $O(q)$ relations is therefore

$$O(m! \cdot q \cdot (m^3 + m^2 \log q))\,.$$

operations in $\mathbb{F}_q$.

Since in each row of the final relations matrix there will be $O(m)$ non-zero elements, we conclude that finding a kernel vector using sparse matrix techniques [13] requires $O(mq^2)$ operations in $\mathbb{Z}/(q^m+1)\mathbb{Z}$ or about $O(m^3q^2)$ operations in $\mathbb{F}_q$. This proves the following theorem.

Theorem 1. *The expected running time of the T_2-algorithm to compute DLOGs in $T_2(\mathbb{F}_{q^m})$ is*

$$O(m! \cdot q \cdot (m^3 + m^2 \log q) + m^3q^2)$$

operations in $\mathbb{F}_q$.

Note that when $m > 1$ and the q^2 term dominates, by reducing the size of the decomposition base, the complexity may be reduced to $O(q^{2-2/m})$ for $q \to \infty$ using the results of Thériault [26], and a refinement reported independently by Gaudry and Thomé [11] and Nagao [20].

The expected running time of the T_2-algorithm is minimal when the relation stage and the linear algebra stage take comparable time, i.e. when $m! \cdot q \cdot (m^3 + m^2 \log q) \simeq m^3q^2$ or $m! \simeq q$. The complexity of the algorithm then becomes $O(m^3q^2)$, which can be rewritten as

$$\begin{aligned} O(m^3q^2) &= O\big(\exp(3\log m + 2\log q)\big) \\ &= O\big(\exp(2(\log q)^{1/2}(\log q)^{1/2})\big) \\ &= O\big(\exp(2(m\log m)^{1/2}(\log q)^{1/2})\big) \\ &= O\big(L_{q^m}(1/2, c)\big) \end{aligned}$$

with $c \in \mathbb{R}_{>0}$. Note that for the second and third equality we have used that $m! \simeq q$, and thus by taking logarithms $\log q \simeq m \log m$.

To assess the practicality of the T_2 algorithm, we ran several experiments using a simple Magma implementation, the results of which are given in Table 1. This table should be read as follows: the size of the torus cardinality, i.e., $\log_2(q^m)$, is constant across each row; for a given q^m, the table contains for $m = 1, \ldots, 15$, the $\log_2$ of the expected running times in seconds for the entire algorithm, i.e. both relation collection stage and linear algebra. For instance, for $q^m \cong 2^{300}$ and $m = 15$, the total time would be approximately 2^{51} seconds on one AMD 1700+ using our Magma implementation. For the fields where the torus is less than 160 bits in size, we use the full torus otherwise we use a subgroup of 160 bits to estimate the Pollard ρ costs.

Note that Table 1 does not take into account *memory* constraints imposed by the linear algebra step; since the number of relations is approximately q, we conclude that the algorithm is currently only practical for $q \leq 2^{23}$. Assuming that 2^{45} seconds, which is about 1.1×10^6 years, is feasible and assuming it is possible to find a kernel vector of a sparse matrix of dimension 2^{23}, Table 1 contains, in bold, the combinations of q and m which can be handled using our Magma implementation.

Table 1. $\log_2$ of expected running times (s) of the T_2-algorithm and Pollard-Rho in a subgroup of size 2^{160}

			m														
$\log_2 \lvert\mathbb{F}_{q^{2m}}\rvert$	$\log_2 \lvert T_2(\mathbb{F}_{q^m})\rvert$	ρ	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
200	100	34	88	40	52	36	**26**	**20**	**16**	**17**	**18**	**21**	**23**	**26**	**31**	**33**	**37**
300	150	59	138	66	87	62	48	38	**31**	**26**	**25**	**26**	**28**	**31**	**34**	**37**	**40**
400	200	65	188	92	121	88	68	55	46	39	**34**	**32**	**33**	**35**	**38**	**41**	**44**
500	250	66	238	117	155	114	89	73	61	52	45	40	**38**	**40**	**42**	**44**	47
600	300	66	289	142	189	139	110	90	76	65	57	51	45	44	46	48	51
700	350	66	339	168	223	165	130	107	91	78	69	61	55	50	50	52	54
800	400	66	389	193	256	190	150	124	105	91	80	71	64	58	56	55	58
900	450	68	439	219	290	215	171	141	120	104	92	82	74	67	62	61	62
1000	500	69	489	244	324	241	191	158	134	117	103	92	83	76	69	66	67

4.5 Comparison with other methods

In this section we compare the T_2-algorithm with the Pollard-Rho and index calculus algorithms.

Pollard-Rho in the full torus Using the Pohlig-Hellman reduction, the overall running time is determined by executing the Pollard-Rho algorithm in the subgroup of $T_2(q^m)$ of largest prime order l. Since $\#T_2(q^m) = q^m + 1$, we have to analyse the size of the largest prime factor l. Note that the factorisation of

$x^m + 1$ over $\mathbb{Z}[x]$ is given by

$$x^m + 1 = \frac{x^{2m} - 1}{x^m - 1} = \frac{\prod_{d|2m} \Phi_d(x)}{\prod_{d|m} \Phi_d(x)} = \prod_{d|2m, d\nmid m} \Phi_d(x) \,,$$

which implies that the maximum size of the prime l is $O(q^{\phi(2m)})$, since the degree of $\Phi_{2m}(x)$ is $\phi(2m)$. The overall worst case complexity of this method is therefore $O(q^{\phi(2m)/2})$ operations in $\mathbb{F}_{q^{2m}}$ or $O(m^2 \cdot q^{\phi(2m)/2})$ operations in $\mathbb{F}_q$.

From a complexity theoretic point of view, we therefore conclude that for $m! \leq q$, our algorithm is as fast as Pollard-Rho whenever $m \geq 5$, since then $\phi(2m)/2 > 2$. As a consequence, we note that the T_2 algorithm does not lead to an improvement over existing attacks on LUC [25], XTR [16] or CEILIDH [22] over $\mathbb{F}_p$. Furthermore, also the security of MNT curves [19] defined over $\mathbb{F}_p$, where p is a large prime remains unaffected.

Pollard-Rho in a subgroup of prime order $\simeq 2^{160}$ In cryptographic applications however, one would work in a subgroup of $T_2(\mathbb{F}_{q^m})$ of prime order l with $l \simeq 2^{160}$. To this end, we measured the average time taken for one multiplication for the various fields in Magma, and multiplied this time by the expected 2^{80} operations required by the Pollard-Rho algorithm. The results can be found in the third column of Table 1. The column for $m = 15$ is especially interesting since this determines the security of the T_{30} cryptosystem introduced in [27]. In this case, the T_2 is always faster than Pollard-Rho, and the matrices occurring in the linear algebra step would be feasible up to 700-bit fields.

Adleman/Demarrais in $\mathbb{F}_{q^{2m}}^{\times}$ The alternative approach would be to embed $T_2(\mathbb{F}_{q^m})$ into $\mathbb{F}_{q^{2m}}^{\times}$ and to apply a subexponential algorithm, which for all m and q can attain a complexity of $L_{q^{2m}}(1/2, c)$ as shown by Adleman and Demarrais [1]. Clearly, using the T_2 algorithm this is only possible for certain combinations of m and q, e.g. for $q \simeq m!$, which is also indicated by Table 1. Of course, when $q = p^n$ for p a prime, then we can choose a different $\bar{m}$ with $\bar{m}|n \cdot m$ such that $\bar{m}! \simeq p^{nm/\bar{m}}$. We do not know how the Adleman-DeMarrais algorithms performs.

Remark 1. The linearity of the decomposition method in fact holds for any torus T_{p^r}. However the savings are optimal for T_{2^r}, since $p^r/\phi(p^r)$ is maximal in this case. When one considers T_n for which n is divisible by more than one distinct prime factor, the rational parametrisation becomes non-linear, and hence so does the corresponding decomposition, as we see in the following section.

5 An Index Calculus Algorithm for $T_6(\mathbb{F}_{q^m}) \subset \mathbb{F}_{q^{6m}}^{\times}$

In this section we detail our algorithm to compute discrete logarithms in $T_6(\mathbb{F}_{q^m})$. The main difference with the T_2-algorithm is the non-linearity of the equations involved in the decomposition step.

5.1 Setup

Again, let $\mathbb{F}_{q^m} \cong \mathbb{F}_q[t]/(f(t))$, with $f(t)$ an irreducible polynomial of degree m and where we use the polynomial basis $\{1, t, t^2, \ldots, t^{m-1}\}$. Since T_6 is two-dimensional and rational, it is an easy exercise to construct a birational map from T_6 to $\mathbb{A}^2$ for a given representation of $\mathbb{F}_{q^{6m}}$. For the following exposition we make use of the the CEILIDH field representation and maps, as described in [22].

Let $q^m \equiv 2$ or $5 \bmod 9$, and for $(r, q) = 1$ let ζ_r denote a primitive r-th root of unity in $\overline{\mathbb{F}}_{q^m}$. Define $x = \zeta_3$ and let $y = \zeta_9 + \zeta_9^{-1}$, then clearly $x^2 + x + 1 = 0$ and $y^3 - 3y + 1 = 0$. Let $\mathbb{F}_{q^{3m}} = \mathbb{F}_{q^m}(y)$ and $\mathbb{F}_{q^{6m}} = \mathbb{F}_{q^{3m}}(x)$, then the bases we use are $\{1, y, y^2 - 2\}$ for the degree three extension and $\{1, x\}$ for the degree two extension.

Let $V(f)$ be the zero set of $f(\alpha_1, \alpha_2) = 1 - \alpha_1^2 - \alpha_2^2 + \alpha_1\alpha_2$ in $\mathbb{A}^2(\mathbb{F}_{q^m})$, then we have the following inverse birational maps:

- $\psi : \mathbb{A}^2(\mathbb{F}_{q^m}) \setminus V(f) \xrightarrow{\sim} T_6(\mathbb{F}_{q^m}) \setminus \{1, x^2\}$, defined by

$$\psi(\alpha_1, \alpha_2) = \frac{1 + \alpha_1 y + \alpha_2(y^2 - 2) + (1 - \alpha_1^2 - \alpha_2^2 + \alpha_1\alpha_2)x}{1 + \alpha_1 y + \alpha_2(y^2 - 2) + (1 - \alpha_1^2 - \alpha_2^2 + \alpha_1\alpha_2)x^2}, \qquad (5)$$

- $\rho : T_6(\mathbb{F}_{q^m}) \setminus \{1, x^2\} \xrightarrow{\sim} \mathbb{A}^2(\mathbb{F}_{q^m}) \setminus V(f)$, which is defined as follows: for $\beta = \beta_1 + \beta_2 x$, with $\beta_1, \beta_2 \in \mathbb{F}_{q^{3m}}$, let $(1 + \beta_1)/\beta_2 = u_1 + u_2 y + u_3(y^2 - 2)$, then $\rho(\beta) = (u_2/u_1, u_3/u_1)$.

5.2 Decomposition base

In this case the decomposition base consists of $\psi(at, 0)$, where a runs through all elements of $\mathbb{F}_q$ and t generates the polynomial basis, i.e.

$$\mathcal{F} = \left\{ \frac{1 + (at)y + (1 - (at)^2)x}{1 + (at)y + (1 - (at)^2)x^2} : a \in \mathbb{F}_p \right\}$$

which clearly contains q elements, for much the same reason as given in Section 4. The reason for considering $\psi(at, 0)$ instead of $\psi(a, 0)$ is that the minimal polynomials of x and y are defined over $\mathbb{F}_q$. Note that this implies that $\psi(a, 0) \in T_6(\mathbb{F}_q)$ for $a \in \mathbb{F}_q$ and so does not generate a fixed proportion of $T_6(\mathbb{F}_{q^m})$, as is needed.

5.3 Relation finding

Since $(\mathrm{Res}_{\mathbb{F}_{q^m}/\mathbb{F}_q} T_6)(\mathbb{F}_q)$ is $2m$-dimensional, we need to solve

$$P_1 + \cdots + P_{2m} = R\,, \qquad (6)$$

with $P_1, \ldots, P_{2m} \in \mathcal{F}$. Assuming that R is expressed in its canonical form, i.e. $R = \psi(r_1, r_2)$, we get

$$\prod_{i=1}^{2m} \left(\frac{1 + (a_i t)y + (1 - (a_i t)^2)x}{1 + (a_i t)y + (1 - (a_i t)^2)x^2} \right)$$
$$= \frac{1 + r_1 y + r_2(y^2 - 2) + (1 - r_1^2 - r_2^2 + r_1 r_2)x}{1 + r_1 y + r_2(y^2 - 2) + (1 - r_1^2 - r_2^2 + r_1 r_2)x^2} \, .$$

After expanding the product of the numerators and denominators, the left hand side becomes the fairly general expression

$$\frac{b_0 + b_1 y + b_2(y^2 - 2) + \big(c_0 + c_1 y + c_2(y^2 - 2)\big)\, x}{b_0 + b_1 y + b_2(y^2 - 2) + (c_0 + c_1 y + c_2(y^2 - 2))\, x^2} \tag{7}$$

with b_i, c_i polynomials over $\mathbb{F}_{q^m}$ of degree $4m$ in $a_1, \ldots, a_{2m}$. In general, these polynomials are rather huge and thus difficult to work with.

Example 1. For $m = 5$, the number of terms in the b_i (resp. c_i) is given by $B = [35956, 30988, 25073]$ (resp. $C = [35946, 31034, 24944]$) for finite fields of large characteristic.

However, note that these polynomials are by construction symmetric in the $a_1, \ldots, a_{2m}$ so we can rewrite the b_i and c_i in terms of the $2m$ elementary symmetric polynomials $\sigma_j(a_1, \ldots, a_{2m})$ for $j = 1, \ldots, 2m$. This has quite a dramatic effect on the complexity of these polynomials, i.e., the degree is now only quadratic and the number of terms is much lower, since the maximum number of terms in a quadratic polynomial in $2m$ variables is $4m + \binom{2m}{2} + 1$.

Example 2. For $m = 5$, when we rewrite the equations using the symmetric functions σ_i, the number of terms of the polynomials b_i and c_i reduces to $B = [16, 19, 18]$ and $C = [20, 16, 16]$.

Note that the polynomials b_i and c_i only have to be computed once and can be reused for each random point R.

To generate the system of non-linear equations, we use the embedding of $T_6(\mathbb{F}_{q^m})$ into $T_2(\mathbb{F}_{q^{3m}})$ and consider the Weil restriction of the following equality:

$$\frac{b_0 + b_1 y + b_2(y^2 - 2)}{c_0 + c_1 y + c_2(y^2 - 2)} = \frac{1 + r_1 y + r_2(y^2 - 2)}{1 - r_1^2 - r_2^2 + r_1 r_2} \, .$$

The above equation leads to 3 non-linear equations over $\mathbb{F}_{q^m}$ or equivalently, to $3m$ non-linear equations over $\mathbb{F}_q$ in the $2m$ unknowns $\sigma_1, \ldots, \sigma_{2m}$. Note that amongst the $3m$ equations, there will be at least m dependent equations, caused by the fact that we only considered the embedding in T_2 and not strictly in T_6.

The efficiency with which one can find the solutions of this system of non-linear equations depends on many factors such as the multiplicities of the zeros or the number of solutions at infinity. For each random R, the resulting system of equations has the same structure, since only the value of some coefficients changes, but for finite fields of large enough characteristic, not the degrees nor the numbers of terms. To determine the properties of these systems of equations we computed the Gröbner basis w.r.t. the lexicographic ordering using the Magma implementation of the F4-algorithm [7] and concluded the following:

- The ideal generated by the system non-linear equations is zero-dimensional, which implies that there is only a finite number of candidates for the σ_i.
- After homogenizing the system of equations, we concluded that there is only a finite number of solutions at infinity. This property is quite important, since we can then use an algorithm by Lazard [14] with proven complexity.
- The Gröbner basis w.r.t. the lexicographic ordering satisfies the so called Shape Lemma, i.e. the basis has the following structure:

  $$\sigma_1 - g_1(\sigma_{2m}),\ \sigma_2 - g_2(\sigma_{2m}),\ \ldots,\ \sigma_{2m-1} - g_{2m-1}(\sigma_{2m}),\ g_{2m}(\sigma_{2m})\,,$$

  where $g_i(\sigma_{2m})$ is a univariate polynomial in σ_{2m} for each i. By reducing modulo g_{2m} we can assume that $\deg(g_i) < \deg(g_{2m})$ and by Bezout's theorem we have $\deg(g_{2m}) \leq 2^{2m}$, since the non-linear equations are quadratic. However, our experiments show that in all cases we have $\deg(g_{2m}) = 3^m$.
- The polynomial $g_{2m}(\sigma_{2m})$ is squarefree, which implies that the ideal is in fact a radical ideal.

To test if a random point decomposes over the factor base, we first find the roots of $g_{2m}(\sigma_{2m})$ in $\mathbb{F}_q$, and then substitute these in the g_i to find the values of the σ_i for $i = 1, \ldots, 2m-1$. For each such $2m$-tuple, we then test if the polynomial

$$p(x) := x^{2m} - \sigma_1 x^{2m-1} + \sigma_2 x^{2m-2} - \cdots + (-1)^{2m}\sigma_{2m}$$

splits completely over $\mathbb{F}_q$. If it does, then the roots a_i for $i = 1, \ldots, 2m$ lead to a possible relation of the form (6).

5.4 Complexity analysis and experiments

The probability of obtaining a relation is now $1/(2m)!$ and since the factor base again consists of q elements, we need to perform $(2m)!q$ decompositions. Each decomposition consists of the following steps:

- Since the polynomials b_i and c_i only need to be computed once, generating the system of non-linear equations requires $O(1)$ multiplications of multivariate polynomials with $O(m^2)$ terms with an $\mathbb{F}_{q^m}$-element. Using a naive multiplication routine, the overall time to generate one such system is therefore $O(m^4)$ operations in $\mathbb{F}_q$.
- Computing the Gröbner basis using the F5-algorithm algorithm [8] requires $O(\binom{4m}{2m}^{\omega})$ operations in $\mathbb{F}_q$, with ω the complexity of matrix multiplication, i.e. $\omega = 3$ using a naive algorithm. Using the fact that

  $$\binom{2n}{n} \cong \sqrt{\frac{\pi}{2}}(2n)^{-1/2}2^{2n} \in O(2^{2n})$$

  we obtain a complexity of $O(2^{12m})$ operations in $\mathbb{F}_q$.

– Since $\deg(g_{2m}) = 3^m$, computing $\gcd(g_{2m}(z), z^q - z)$ requires $O(3^{2m} \log q)$ operations in $\mathbb{F}_q$. On average, the polynomial will have one root in $\mathbb{F}_q$, so finding the actual roots takes negligible time.
– Testing if the polynomial $p(x)$ has roots in $\mathbb{F}_q$ requires $O(m^2 \log q)$ operations in $\mathbb{F}_q$. Since this only happens with probability $1/(2m)!$, when it does split, finding the actual roots is negligible.

The overall time complexity to generate sufficient relations therefore amounts to

$$O\left((2m)! \cdot q \cdot (2^{12m} + 3^{2m} \log q)\right)$$

operations in $\mathbb{F}_q$.

Finding an element in the kernel of a matrix of dimension q with $2m$ non-zero elements per row requires $O(mq^2)$ operations in $\mathbb{Z}/(\Phi_6(q^m)\mathbb{Z})$, which finally justifies the following complexity estimate:

Run Time Heuristic 1 *The expected running time of the T_6-algorithm to compute DLOGs in $T_6(\mathbb{F}_{q^m})$ is*

$$O((2m)! \cdot q \cdot (2^{12m} + 3^{2m} \log q) + m^3 q^2)$$

operations in $\mathbb{F}_q$.

Again, the results of [26, 11, 20] imply that the complexity can be reduced to $O(q^{2-1/m})$ as $q \to \infty$, since in this case the dimension is $2m$.

The expected running time of the T_6-algorithm is minimal precisely when the relation collection stage takes about the same time as the linear algebra stage, i.e. when $(2m)! \cdot 2^{12m} \simeq q$. Note that for such q and m, the term $3^{2m} \log q$ is negligible compared to 2^{12m}. The overall running time then again becomes

$$\begin{aligned} O(m^3 q^2) &= O\big(\exp(3 \log m + 2 \log q)\big) \\ &= O\big(\exp(2(\log q)^{1/2} (\log q)^{1/2})\big) \\ &= O\big(\exp(2(2m \log 2m + 12m)^{1/2} (\log q)^{1/2})\big) \\ &= O\big(L_{q^m}(1/2, c)\big) \end{aligned}$$

with $c \in \mathbb{R}_{>0}$. Note that for the second and third equality we have used $\log q \simeq 2m \log m + 12m \log 2$.

The practicality of the T_6-algorithm clearly depends on the efficiency of the Gröbner basis computation. Note that for small m, the complexity of the Gröbner basis computation is greatly overestimated by the $O(2^{12m})$ operations in $\mathbb{F}_q$.

Due to the use of the symmetric polynomials, the input polynomials are only quadratic instead of degree $4m$. As one can see from Table 2, this makes the algorithm quite practical. The table should be interpreted as for Table 1, i.e., the torus size is constant across each row and for a given size q^m, the table contains for $m = 1, \ldots, 5$, the $\log_2$ of the expected running times in seconds for the entire algorithm. Taking into account the memory restrictions on the matrix, i.e., the dimension should be limited by 2^{23}, the timings given in bold are feasible with the current Magma implementation.

Table 2. $\log_2$ of expected running times (s) of the T_6-algorithm and Pollard-Rho in a subgroup of size 2^{160}

			m				
$\log_2 \lvert\mathbb{F}_{p^{6m}}\rvert$	$\log_2 \lvert T_6(\mathbb{F}_{p^m})\rvert$	ρ	1	2	3	4	5
200	67	18	25	**18**	**14**	**20**	**29**
300	100	34	42	36	**21**	**24**	**32**
400	134	52	59	54	**32**	**29**	**36**
500	167	66	75	71	44	**33**	**39**
600	200	66	93	88	55	40	**42**
700	234	66	109	105	67	48	46
800	267	66	127	122	78	57	51
900	300	68	144	139	90	65	56
1000	334	69	161	156	101	74	60

Remark 2. Note that the column for $m = 5$ provides an upper bound for the hardness of the DLP in $T_{30}(\mathbb{F}_q)$, since this can be embedded in $T_6(\mathbb{F}_{q^5})$. This group was recently proposed [27] and also in [15] for cryptographic use where keys of length 960 bits were recommended, i.e., with q of length 32 bits. The above table shows that even with a Magma implementation it would be feasible to compute discrete logarithms in $T_{30}(\mathbb{F}_p)$ with p a prime of around 20 bits. The embedding in $T_2(\mathbb{F}_{p^{15}})$ is about 2^{10} times less efficient as can be seen from the column for $m = 15$ in Table 1. In light of this attack, the security offered by the DLP in finite fields of the form $\mathbb{F}_{q^{30}}$ should be completely reassessed. Note that by simply comparing the complexities given in Theorem 1 and the above run time heuristic, it is a priori not clear that the T_6-algorithm is in fact faster than the corresponding T_2-algorithm. This phenomenon is caused by the overestimating the complexity of the Gröbner basis computation.

5.5 Comparison with other methods

In this section we compare the T_6-algorithm with the Pollard-Rho and index calculus algorithms.

Pollard-Rho in the full torus Since the size of $T_6(\mathbb{F}_{q^m})$ is given by $\Phi_6(q^m) \simeq q^{2m}$, we conclude that the Pollard-Rho algorithm takes, in the worst case, $O(q^m)$ operations in $T_6(\mathbb{F}_{q^m})$ or $O(m^2 q^m)$ operations in $\mathbb{F}_q$. If we assume that q is large enough such that the term q^2 determines the overall running time, i.e., $(2m)!2^{12m} \leq q$, then the T_6-algorithm will be at least as fast as Pollard-Rho whenever $m \geq 3$. Again we note that the T_6 algorithm does not lead to an improvement over the existing attacks on LUC [25], XTR [16], CEILIDH [22] or MNT curves [19] as long as these systems are defined over $\mathbb{F}_p$. However, the security of XTR over extension fields, as proposed in [17] or of the recent proposal that works in $T_{30}(\mathbb{F}_p)$ [27], needs to be reassessed as shown below.

Pollard-Rho in a subgroup of prime order $\simeq \mathbf{2^{160}}$ As for the T_2-algorithm, the third column of Table 2 contains the expected running time of the Pollard-Rho algorithm in a subgroup of $T_6(\mathbb{F}_{q^m})$ of prime order l with $l \simeq 2^{160}$. In this case, the column for $m = 5$ gives an upper bound of the security of the T_{30} cryptosystem introduced in [27]. As is clear from Table 2, for $m = 5$, our algorithm is always faster than Pollard-Rho, and the matrices occurring in the linear algebra step would be feasible up to 700-bit fields.

Adleman/Demarrais in $\mathbb{F}^{\times}_{\boldsymbol{q^{6m}}}$ Using the embedding of $T_6(\mathbb{F}_{q^m})$ into $\mathbb{F}^{\times}_{q^{6m}}$ one can apply the subexponential algorithm of Adleman-Demarrais [1] which runs, for all m and q, in time $L_{q^{6m}}(1/2, c)$. Using the T_6 algorithm, it is possible to obtain a complexity of $L_{q^m}(1/2, c')$, but only when m and q grow according to a specific relation such as $(2m)!2^{12m} \simeq q$. Again, when $q = p^n$ with p a prime, we could choose a different $\bar{m}$ with $\bar{m}|n \cdot m$ such that $(2\bar{m})!2^{12\bar{m}} \simeq p^{mn/\bar{m}}$.

However, as was the case for the T_2-algorithm, the importance of Table 2 is that it contains the first practical upper bounds for the hardness of the DLP in extension fields $\mathbb{F}^{\times}_{q^{6m}}$, since there are no numerical experiments available based on the existing subexponential algorithms.

6 Conclusion and Future Work

In this paper we have presented an index calculus algorithm, following ideas of Gaudry, to compute discrete logarithms on rational algebraic tori. Our algorithm works directly in the torus and depends fundamentally on the compression mechanisms previously used in a constructive context for systems such as LUC, XTR and CEILIDH.

We have also provided upper bounds for the difficulty of solving discrete logarithms on the tori $T_2(\mathbb{F}_{q^m})$ and $T_6(\mathbb{F}_{q^m})$ for various q and m in the cryptographic range. These upper bounds indicate that if the techniques in this paper can be made fully practical and optimized, then they may weaken the security of practical systems based on T_{30}.

In the near future we wish to investigate the approach by Diem [4], who allows a larger decomposition base when necessary. The disadvantage of this approach is that it destroys the symmetric nature of the polynomials defining the decomposition of a random element over the factor base, which makes Gröbner basis techniques virtually impossible.

It is clear that the Magma implementations described in this paper are not optimised and many possible improvements exist. Two factors mainly determine the running time of the algorithm: first of all, the probability that a random element decomposes over the factor base and secondly, the time it takes to solve a system of non-linear equations over a finite field. The first factor could be influenced by designing some form of sieving, if at all possible, whereas the second factor could be improved by exploiting the fact that many very similar Gröbner bases have to be computed.

In addition the method needs to be compared in practice to the method of Adleman and DeMarrais.

Acknowledgements

The authors would like to thank Daniel Lazard for his invaluable comments regarding the details of the complexity of the Gröbner basis computation in the T_6-algorithm, and anonymous referees for constructive comments on earlier versions of this paper.

References

1. L. M. Adleman and J. DeMarrais. *A subexponential algorithm for discrete logarithms over all finite fields.* Math. Comp., **61 (203)**, 1–15, 1993.
2. A. E. Brouwer, R. Pellikaan and E. R. Verheul. *Doing more with fewer bits.* In Advances in Cryptology (ASIACRYPT 1999), Springer LNCS 1716, 321–332, 1999.
3. B. Buchberger. *A theoretical basis for the reduction of polynomials to canonical forms.* ACM SIGSAM Bull., **10 (3)**, 19–29, 1976.
4. C. Diem. *On the discrete logarithm problem in elliptic curves over non-prime fields.* Preprint 2004. Available from the author.
5. W. Diffie and M. E. Hellman. *New directions in cryptography.* IEEE Trans. Inform. Theory **22 (6)**, 644–654, 1976.
6. T. ElGamal. *A public key cryptosystem and a signature scheme based on discrete logarithms.* In Advances in Cryptology (CRYPTO 1984), Springer LNCS 196, 10–18, 1985.
7. J.-C. Faugère. *A new efficient algorithm for computing Gröbner bases* (F_4), J. Pure Appl. Algebra **139 (1-3)**, 61-88, 1999.
8. J.-C. Faugère. *A new efficient algorithm for computing Gröbner bases without reduction to zero* (F_5), In Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation, 75–83, 2002.
9. FIPS 186-2, *Digital signature standard.* Federal Information Processing Standards Publication 186-2, February 2000.
10. P. Gaudry. *Index calculus for abelian varieties and the elliptic curve discrete logarithm problem.* Cryptology ePrint Archive, Report 2004/073. Available from `http://eprint.iacr.org/2004/073`.
11. P. Gaudry and E. Thomé. *A double large prime variation for small genus hyperelliptic index calculus.* Cryptology ePrint Archive, Report 2004/153. Available from `http://eprint.iacr.org/2004/153`.
12. R. Granger, D. Page and M. Stam. *A comparison of CEILIDH and XTR.* In Algorithmic Number Theory Symposium (ANTS-VI), Springer LNCS 3076, 235–249, 2004.
13. B. A. LaMacchia and A. M. Odlyzko. *Solving large sparse linear systems over finite fields.* In Advances in Cryptology (CRYPTO 1990), Springer LNCS 537, 109–133, 1991.
14. D. Lazard. Résolution des systèmes d'équations algébriques, Theoret. Comput. Sci., **15 (1)**, 77–110, 1981.
15. A. K. Lenstra. *Using cyclotomic polynomials to construct efficient discrete logarithm cryptosystems over finite fields.* In Proceedings of ACISP97, Springer LNCS 1270, 127–138, 1997.

16. A. K. Lenstra and E. Verheul. *The XTR public key system.* In Advances in Cryptology (CRYPTO 2000), Springer LNCS 1880, 1–19, 2000.

17. S. Lim, S. Kim, I. Yie, J. Kim and H. Lee. *XTR extended to GF(p^{6m}).* In Selected Areas in Cryptography (SAC 2001), Springer LNCS 2259, 301–312, 2001.

18. A. J. Menezes, P. van Oorschot and S. A. Vanstone. *The Handbook of Applied Cryptography*, CRC press, 1996.

19. A. Miyaji, M. Nakabayashi and S. Takano. *New explicit conditions of elliptic curve traces for FR-reduction.* IEICE Trans. Fundamentals **E84-A (5)**, 1234–1243, 2001.

20. K. Nagao. *Improvement of Thériault algorithm of index calculus for Jacobian of hyperelliptic curves of small genus.* Cryptology ePrint Archive, Report 2004/161. Available from `http://eprint.iacr.org/2004/161`.

21. A. M. Odlyzko. *Discrete logarithms in finite fields and their cryptographic significance.* In Advances in Cryptology (EUROCRYPT 1984), Springer LNCS 209, 224–314, 1985.

22. K. Rubin and A. Silverberg. *Torus-based cryptography.* In Advances in Cryptology (CRYPTO 2003), Springer LNCS 2729, 349–365, 2003.

23. K. Rubin and A. Silverberg. *Using primitive subgroups to do more with fewer bits.* In Algebraic Number Theory Symposium (ANTS-VI), Springer LNCS 3076, 18–41, 2004.

24. C. P. Schnorr. *Efficient signature generation by smart cards.* J. Cryptology, **4**, 161–174, 1991.

25. P. Smith and C. Skinner. *A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms.* In Advances in Cryptology (ASIACRYPT 1995), Springer LNCS 917, 357–364, 1995.

26. N. Thériault. *Index calculus attack for hyperelliptic curves of small genus.* In Advances in Cryptology (ASIACRYPT 2003), Springer LNCS 2894, 75–92, 2003.

27. M. van Dijk, R. Granger, D. Page, K. Rubin, A. Silverberg, M. Stam and D. Woodruff. *Practical cryptography in high dimensional tori.* In Advances in Cryptology (EUROCRYPT 2005), Springer LNCS 3494, 234–250, 2005.

28. M. van Dijk and D. P. Woodruff. *Asymptotically optimal communication for torus-based cryptography.* In Advances in Cryptology (CRYPTO 2004), Springer LNCS 3152, 157–178, 2004.