Can we Beat the Square Root Bound for ECDLP over $\mathbb{F}_{p^2}$ via Representations?

Claire Delaplace and Alexander May*

Horst Görtz Institute for IT Security
Ruhr University Bochum, Germany

Abstract. We give a 4-list algorithm for solving the Elliptic Curve Discrete Logarithm (ECDLP) over some quadratic field $\mathbb{F}_{p^2}$. Using the representation technique, we reduce ECDLP to a multivariate polynomial zero testing problem. Our solution of this problem using bivariate polynomial multi-evaluation yields a $p^{1.314}$-algorithm for ECDLP. While this is inferior to Pollard's Rho algorithm with square root (in the field size) complexity $\mathcal{O}(p)$, it still has the potential to open a path to an $o(p)$-algorithm for ECDLP, since all involved lists are of size as small as $p^{\frac{3}{4}}$, only their computation is yet too costly.

1 Introduction

Given an elliptic curve $\mathbf{E}$, defined over a finite field $\mathbb{F}_q$ and two points P and Q on this curve, the *elliptic curve discrete logarithm problem* (`ECDLP`) consists in recovering k modulo order of P such that $Q = kP$, if it exists.

Nowadays, ECDLP-based cryptosystems are omnipresent in everyday life cryptography, as they are widely spread in cryptographic suites such as TLS.

Many algorithms for `ECDLP` have been proposed in the literature from Shanks *Baby-Step Giant-Step* [Sha71], to Weil descent and index calculus methods [FG98,GS99,GHS02a,GHS02b,Gau09].
Other algorithms have been designed for particular cases, such as elliptic curves over a field $\mathbb{F}_p$ of group order exactly p [SA+98,Sma99], or in a subgroup of order p [Sem98]. We refer the reader to [GG16] for an overview of `ECDLP` algorithms.

Pollard's Rho algorithm offers the best time and space complexity for elliptic curves defined over a finite field $\mathbb{F}_p$, where p is prime, with a running time of $\mathcal{O}\left(\sqrt{p}\right)$ and constant memory (e.g. using Floyd's cycle finding algorithm). Pollard's Rho method basically creates a sequence $(R_i)_{i\geq 1}$ of elements of $\langle P\rangle$ with $R_0 = O$ and $R_{i+1} = g(R_i)$ for a well chosen function g, such that each R_i satisfies $R_i = a_iP + b_iQ$. Once a collision appears between two elements R_i and R_j, we have $(a_i - a_j)P = (b_j - b_i)Q$. Provided that $b_i \neq b_j$, it follows that $Q = \frac{a_i - a_j}{b_j - b_i}P$.

When the elliptic curve is defined over $\mathbb{F}_{p^n}$ for $n \geq 2$, and p being a prime or a power of a prime, it is possible to perform what is called a *Weil Descent*.

* Funded by DFG under Germany's Excellence Strategy - EXC 2092 CASA - 390781972.

The core idea of Gaudry's algebraic factor-base algorithm [Gau09] for $\mathbb{F}_{p^n}$ is to split a point into an n-sum of points from a factor base that is defined over the base field $\mathbb{F}_p$. Using Semaev's summation polynomials [Sem04], the split can be computed via a Gröbner basis computation. Such a computation yields a relation between factor base elements. If at least q – the size of the factor base – such relations are collected, ECDLP can be solved via linear algebra. In total, Gaudry's method runs in time $\mathcal{O}\left(p^{2-\frac{2}{n}}\right)$, which is for $n = 2$ no better than Pollard's Rho algorithm. Moreover, with a factor base of size p, there is no hope to obtain an $o(p)$-algorithm. Yet, variations of the factor base might be possible as demonstrated in Petit et al. [PKM16].

Our contribution. Our (ultimate) goal is to break the square root bound for ECDLP over $\mathbb{F}_{p^2}$ by designing a 4-list algorithm using the representation technique. The representation technique already proved to be useful to break the square-root barrier in the context of the subset-sum problem [HGJ10,BCJ11].

We define a problem, called Zero-Join (`ZJ-Problem`), which given two lists A and B of points of $\mathbb{F}_p^2$, and a polynomial $\mathtt{f} \in \mathbb{F}_p[X_1, X_2, X_3, X_4]$ consists in returning a list C of all $((x_1, y_1), (x_2, y_2)) \in A \times B$ such that $\mathtt{f}(x_1, y_1, x_2, y_2) = 0$.

We show that A and B are such that $|A| \cdot |B| = p^{\frac{3}{2}}$. Moreover, we show that any algorithm which solves the `ZJ-Problem` in time T and in memory M also solves `ECDLP` over $\mathbb{F}_{p^2}$ in time T and memory M. In particular, if $|A| = |B| = p^{\frac{3}{4}}$, and in (the extreme) case that the `ZJ-Problem` could be solved in time linear in $|A|$ and $|B|$, `ECDLP` could be solved in $\mathcal{O}\left(p^{\frac{3}{4}}\right)$. A trivial solution to the `ZJ-Problem` can be achieved in quadratic time which results in a $p^{\frac{3}{2}}$-algorithm. However, we show that the `ZJ-Problem` admits sub-quadratic solutions. When using multi-evaluation techniques for bivariate polynomials, we achieve a $p^{1.314}$-algorithm. We leave it as an open problem whether this can be further improved.

Organisation of the paper. In Section 3, we present our new `ECDLP` 4-list algorithm for an elliptic curve defined over $\mathbb{F}_{p^2}$, for any prime $p > 3$. We also show that ECDLP reduces to the `ZJ-Problem`. Finally, we present a sub-quadratic algorithm to solve the `ZJ-Problem` in Section 4, resulting in our $p^{1.314}$-algorithm.

2 Preliminaries

For any integer r, we denote by $\mathbb{Z}_r$ the ring $\mathbb{Z}/r\mathbb{Z}$. For any two integers $a < b \in \mathbb{Z}$, we denote by $[a, b]$ the set of all integers $a \leq k \leq b$ and by $[a, b)$ the set of all integers $a \leq k < b$. For any $n > 0$, we denote by $\log n$ the logarithm in basis 2 of n. For any real x, we denote by $\lfloor x \rceil$ the rounding to the closest integer to x, when tied, we round to the greater one. Let us denote $\mathbb{F}_p$ the finite field with p elements. Let x be an element of $\mathbb{F}_{p^2}$, $x = x^{(0)} + \alpha x^{(1)}$, such that there exists $\beta \in \mathbb{F}_p$ and α is a root of $X^2 - \beta$, which is irreducible over $\mathbb{F}_p$. Let $\mathcal{B}$ be a basis of $\mathbb{F}_p^2$ with respect to α. The vector $(x^{(0)}, x^{(1)}) \in \mathbb{F}_p^2$ in basis $\mathcal{B}$ is uniquely associated to x.

Let $q = p^n$ for $n \geq 1$, and $p > 3$ prime. Let $\mathbb{F}_q$ be a finite field and let $\mathbf{E}$ be an elliptic curve over $\mathbb{F}_q$. We denote by O the point at infinity of $\mathbf{E}$. The group $\mathbf{E}(\mathbb{F}_q)$ is the group of all the points in $\mathbf{E}$. As $p > 3$, we can define $\mathbf{E}$ in Weierstraß form. In particular, there is a function $f : \mathbb{F}_q \to \mathbb{F}_q$ that maps elements x to $f(x) = x^3 + ax + b$, for some $a, b \in \mathbb{F}_q$ such that the set of points $\mathbf{E}(\mathbb{F}_q)$ equals $\{P = (x, y) \in \mathbb{F}_q^2 : \ y^2 = f(x)\} \cup \{O\}$. Although there are other ways to represent an elliptic curve, in this paper, we focus on the Weierstraß model.

Let $\mathcal{S}_1$ and $\mathcal{S}_2$ be subsets of $\mathbb{N}$, we denote by $\mathcal{S}_1 + \mathcal{S}_2$ the set of integers z such that $z = x + y$ with $x \in \mathcal{S}_1$ and $y \in \mathcal{S}_2$, and for all $\ell \in \mathbb{N}$, we denote by $\ell\mathcal{S}_1$ the set of integers z such that $z = \ell x$ with $x \in \mathcal{S}_1$.

Let L be a list. We consider L as a tuple $(\ell_0, \ldots, \ell_{N-1})$, where $N = |L| \geq 0$. We denote by $L[i]$ the $(i + 1)$-th element of the list. Given two lists $L_1 = (\ell_0, \ldots, \ell_N)$ and $L_2 = (t_0, \ldots, t_M)$, we denote by $L_1 + L_2$ the list $(\ell_0 + t_0, \ell_0 + t_1 \ldots \ell_0 + t_M \ldots \ell_N + t_M)$. An empty list is denoted by $\perp$.

Complexities. We use the standard Landau notations to describe the complexities presented in this paper. The time complexities are given in number of elementary operations (negation, addition, multiplication, inverse) over $\mathbb{F}_p$, and the memory complexities in number of elements of $\mathbb{F}_p$ that are to be stored. Using this convention, elementary operations over $\mathbb{F}_{p^2}$ and sum of two points of the group $\mathbf{E}(\mathbb{F}_{p^2})$ are performed in time $\mathcal{O}(1)$, and storing an element of $\mathbb{F}_{p^2}$ or a point of $\mathbf{E}(\mathbb{F}_{p^2})$ requires $\mathcal{O}(1)$ memory.

Discrete logarithm over an elliptic curve. Let $\mathbf{E}$ be an elliptic curve over a finite field $\mathbb{F}_q$. Let us denote by $|\mathbf{E}(\mathbb{F}_q)|$ the order of the group (i.e. the cardinality of $\mathbf{E}(\mathbb{F}_q)$). By Hasse's Theorem, we know that $|\mathbf{E}(\mathbb{F}_q)| = \mathcal{O}(q)$. Let P be a point of $\mathbf{E}(\mathbb{F}_q)$. The *order* r of P is the cardinality of the cyclic group $\langle P \rangle = \{O, P, 2P, \ldots, (r-1)P\}$. It is also the first integer $\ell > 0$ such that $\ell P = O$. As $\langle P \rangle$ is a subgroup of $\mathbf{E}(\mathbb{F}_q)$, it is clear that $r \leq |\mathbf{E}(\mathbb{F}_q)|$.

Definition 1 (ECDLP). *Let $\mathbb{F}_q$ be a finite field, and let $\mathbf{E}$ be an elliptic curve over $\mathbb{F}_q$. Let P be a point of $\mathbf{E}(\mathbb{F}_q)$ of prime order $r = \mathcal{O}(q)$. Let $G = \langle P \rangle$. Given P and Q in G, solving the* discrete logarithm problem over $\mathbf{E}$ *consists in recovering $k \in \mathbb{Z}_r$ such that $kP = Q$.*

We are interested in the case where $q = p^2$, for a prime $p > 3$. We call this particular variant p^2-ECDLP.

Representation Technique. In [HGJ10], Howgrave-Graham and Joux introduced the representation technique to improve methods for the subset-sum problem. Given a vector $\mathbf{a} \in \mathbb{Z}^n$ and a target t in $\mathbb{Z}$ the problem basically consists in finding a vector $\mathbf{e} \in \{0, 1\}^n$ such that $\langle \mathbf{a}, \mathbf{e} \rangle = t$. When the solution $\mathbf{e}$ consists exactly of $n/2$ non-zero coefficients, Howgrave-Graham and Joux noticed that it can be split into sums of two vectors $\mathbf{e} = \mathbf{e}_1 + \mathbf{e}_2$, where $\mathbf{e}_1$ and $\mathbf{e}_2$ both consist of $n/4$ one coefficients, in $\binom{n/2}{n/4}$ different ways. It is enough to find only one of these representations to recover a solution to the problem, thus additional

constraints can be enforced on the dot products $\langle \mathbf{a}, \mathbf{e}_1 \rangle$ and $\langle \mathbf{a}, \mathbf{e}_2 \rangle$, so that only one of these representation survives. This yields a $2^{0.337n}$ algorithm for subset-sum that breaks the square root bound.

We would like to transfer this idea to the p^2-ECDLP problem.

Definition 2 (Representation). *Let $\ell > 0$, $r \in \mathbb{N}$, $k \in \mathbb{Z}_r$, and let $\mathcal{S}_1, \dots \mathcal{S}_\ell$ be subsets of $\mathbb{N}$. A tuple $(k_1, \dots, k_\ell) \in \mathcal{S}_1 \times \cdots \times \mathcal{S}_\ell$ is a* representation *of k if $k = k_1 + \cdots + k_\ell \mod r$.*

3 New ideas to solve p^2-ECDLP

Let $\mathbf{E}(\mathbb{F}_{p^2})$ be an elliptic curve defined over $\mathbb{F}_{p^2}$, with $p > 3$. Let r be the order of $\mathbf{E}(\mathbb{F}_{p^2})$, which can be computed in time polynomial in $\log p$ with Schoof's algorithm [Sch85,Sch95]. Since we are interested in cryptographic applications we assume that r is prime. By Hasse's theorem we know that $r \approx p^2$. Let P generate $\mathbf{E}(\mathbb{F}_{p^2})$ and let $Q = kP$ for some unknown $k \in \mathbb{Z}_r$ that we want to compute.

First notice that for all $(k_1, k_2) \in \mathbb{Z}_r^2$, $(k_1 + k_2)P = Q$ is equivalent to $k_1 P = Q - k_2 P$. In the following we try to recover such a tuple (k_1, k_2).

Heuristic. For the complexity analysis of our algorithm, we assume that the points of $\mathbf{E}(\mathbb{F}_{p^2})$ are uniformly distributed over $\mathbb{F}_{p^2}^2$. The results of Kim, Tibouchi [KT19] provide some theoretical justification for our assumption.

3.1 Solving p^2-ECDLP Using the Representation Technique

First note that every positive integer s can be uniquely decomposed as

$$s = s_0 + s_1 \lfloor \sqrt{p} \rceil + s_2 p, \tag{1}$$

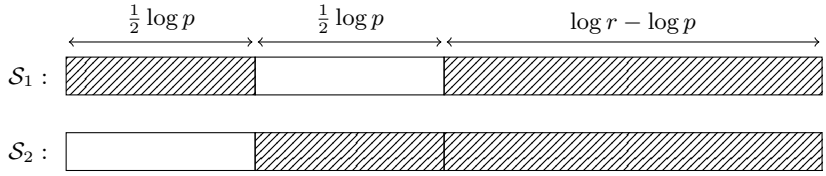
by computing first the integer division of $s = s_2 p + \bar{s}$ by p, and then the integer division of $\bar{s} = s_0 + s_1 \lfloor \sqrt{p} \rceil$ by $\lfloor \sqrt{p} \rceil$. In this case, s_0, s_1, s_2 are in $\mathbb{N}$ such that $0 \leq s_0 < \lfloor \sqrt{p} \rceil$, $0 \leq s_1 < p \cdot \lfloor \sqrt{p} \rceil^{-1}$ and $s_2 \geq 0$.

With respect to this decomposition, we denote by $\mathcal{S}_1$ and $\mathcal{S}_2$ the subsets of $[0, r)$ such that

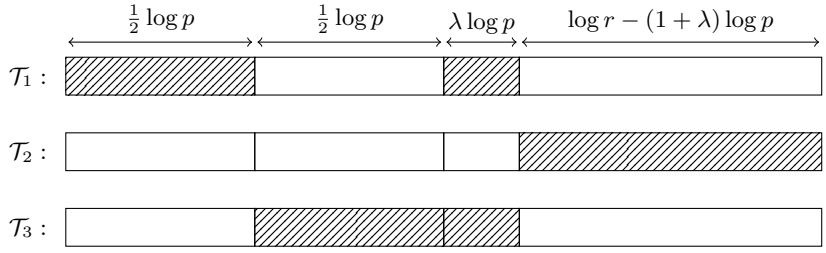
$$\mathcal{S}_1 = \{s = s_0 + s_2 p\}, \quad \mathcal{S}_2 = \{s = s_1 + s_2 p\}.$$

Informally, the bits of the elements of $\mathcal{S}_1$ and $\mathcal{S}_2$ are dispatched as shown in Figure 1a. We consider k as the sum $k_1 + k_2$ where $k_1 \in \mathcal{S}_1$ and $k_2 \in \mathcal{S}_2$. Since k_1, k_2 overlap in $\log(\frac{r}{p})$ bits we expect that each k has at least $\frac{r}{p} \approx p$ representations as a sum $k_1 + k_2$. This is shown more formally in the following lemma.

Lemma 1. *There are at least $\frac{r}{p} + 1$ representations of k as the sum $k_1 + k_2$ with $(k_1, k_2) \in \mathcal{S}_1 \times \mathcal{S}_2$.*



(a) Bits repartition of elements from $\mathcal{S}_1$ and $\mathcal{S}_2$, as vector of $\mathbb{F}_2^{\log r}$, the leftmost part being the least significant bits of the numbers. The white rectangles represent a portion of zeroes.



(b) Bits repartition of elements from $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ as vector of $\mathbb{F}_2^{\log r}$, the leftmost part being the least significant bits of the numbers. The white rectangles represent a portion of zeroes.

Fig. 1: Bits repartition of the elements of $\mathcal{S}_i$ and $\mathcal{T}_j$ for $i \in [1,2],\ j \in [1,3]$.

Proof. Let $\bar{k} = \bar{k}_0 + \bar{k}_1 \left\lfloor \sqrt{p} \right\rceil + \bar{k}_2 p \in [0, r)$ be such that $\bar{k} \pmod r = k$, and let $\hat{k} = \hat{k} + r = \hat{k}_0 + \hat{k}_1 \left\lfloor \sqrt{p} \right\rceil + \hat{k}_2 p$ when decomposed as in Equation 1. We denote by $\mathcal{S}_{11}$ the subset of $\mathcal{S}_1$ of the elements $s = \bar{k}_0 + s_2 p$, and by $\mathcal{S}_{12}$ the subset of $\mathcal{S}_1$ of the elements $s = \hat{k}_0 + s_2 p$.

We first show that for any $k_1 \in \mathcal{S}_{11}$ such that $k_1 \leq \bar{k}$, there exists some $k_2 \in \mathcal{S}_2$ such that $k_1 + k_2 = \bar{k}$. Then we claim that for any $k_1 \in \mathcal{S}_{12}$, such that $k_1 > \bar{k}$, there is a $k_2 \in \mathcal{S}_2$ such that $k_1 + k_2 = \hat{k}$.

Let $k_1 \in \mathcal{S}_{11}$ be arbitrary such that $k_1 \leq \bar{k}$. From the definition of $\mathcal{S}_{11}$, $k_1 = \bar{k}_0 + k_{12} p$. In particular $k_1 \leq \bar{k}$ means that $\bar{k}_2 - k_{12} \geq 0$, and as $\bar{k} < r$, $\bar{k} - k_1 \in [0, r)$. Finally we have

$$\begin{aligned} \bar{k} - k_1 &= \bar{k}_0 + \bar{k}_1 \left\lfloor \sqrt{p} \right\rceil + \bar{k}_2 p - \bar{k}_0 - k_{12} p \\ &= \bar{k}_1 \left\lfloor \sqrt{p} \right\rceil + (\bar{k}_2 - k_{12}) p \in \mathcal{S}_2. \end{aligned}$$

Let now $k_1 \in \mathcal{S}_{12}$, such that $k_1 > \bar{k}$. First note that $k_1 < r < \hat{k}$. From the definition of $\mathcal{S}_{12}$, $k_1 = \hat{k}_0 + k_{12} p$, and $\hat{k} - k_1 \geq 0$ means that in particular

$\hat{k}_2 - k_{12} \geq 0$. Furthermore as $k_1 > \bar{k}$, $\hat{k} - k_1 = r + \bar{k} - k_1 \in [0, r)$. Finally

$$\begin{aligned}\hat{k} - k_1 &= \hat{k}_0 + \hat{k}_1 \lfloor \sqrt{p} \rceil + \hat{k}_2 p - \hat{k}_0 - k_{12} p \\ &= \hat{k}_1 \lfloor \sqrt{p} \rceil + (\hat{k}_2 - k_{12}) p \in \mathcal{S}_2.\end{aligned}$$

It follows that the number of representations $(k_1, k_2) \in \mathcal{S}_1 \times \mathcal{S}_2$ of k is at least equal to the number N_1 of elements of $\mathcal{S}_{11}$ that are smaller than $\bar{k}$ plus the number N_2 of elements of $\mathcal{S}_{12}$ that are bigger than $\bar{k}$. N_1 is the number of $k_1 = \bar{k}_0 + k_{12} p$ such that $0 \leq k_{12} \leq \bar{k}_2$, meaning that $N_1 = \bar{k}_2 + 1$. N_2 is the number of $k_1 = \hat{k}_0 + k_{12} p$ such that $\bar{k}_2 \leq k_{12} < \frac{r}{p}$, meaning that $N_2 \approx \frac{r}{p} - \hat{k}_2$ It follows that there are at least $\frac{r}{p} + 1$ representations of k in $\mathcal{S}_1 \times \mathcal{S}_2$. □

Out of the $\frac{r}{p} + 1 \approx p$ representations of k as $k_1 + k_2$, $(k_1, k_2) \in \mathcal{S}_1 \times \mathcal{S}_2$ it suffices to compute a single one. Therefore, computing only a $\frac{1}{p}$-fraction of all points $(P_1, P_2) = (k_1 P, Q - k_2 P)$ is sufficient to compute k. In order to compute a $\frac{1}{p}$-fraction we restrict to those points (P_1, P_2) whose x-coordinate lies in the base field (i.e. $P_i = (x, y)$ with $x \in \mathbb{F}_p$). [1]

More precisely, we proceed as follows. We compute a list L of all $P_1 = k_1 P = (x, y)$ such that $x \in \mathbb{F}_p$ and $k_1 \in \mathcal{S}_1$. Moreover, we compute a list L' of all $P_2 = Q - k_2 P = (x', y')$ for $k_2 \in \mathcal{S}_2$ such that $x' \in \mathbb{F}_p$. Then we search for a collision in L and L'. This gives us k as the sum of the corresponding k_1 and k_2.

This results in `RepECDLP`, described in Algorithm 2. Notice that the resulting lists L, L' have expected size only $p^{1/2}$, since we impose a $\frac{1}{p}$ restriction on a search space of size $p^{2/3}$.

Now let us turn to the tricky part, the computation of L, L'. Let $0 < \lambda < \frac{1}{2}$ be a parameter and let $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ be three sets of elements of $[0, r)$ such that

$$\begin{aligned}\mathcal{T}_1 &= \left\{ s = s_0 + p s_1 \in \mathcal{S}_1 : s_1 \in \left[0, \lfloor p^{\lambda} \rceil\right) \right\} \\ \mathcal{T}_2 &= \left\{ s = p s_1 \in \mathcal{S}_1 \cap \mathcal{S}_2 : s_1 = t \lfloor p^{\lambda} \rceil \in [0, p) \right\} \\ \mathcal{T}_3 &= \left\{ s = s_0 + p s_1 \in \mathcal{S}_2 : s_1 \in \left[0, \lfloor p^{\lambda} \rceil\right) \right\}\end{aligned}$$

The bit repartition of the elements of $\mathcal{T}_1, \mathcal{T}_2$ and $\mathcal{T}_3$ is given by Figure 1b. The reader is advised to use the illustration in Figure 2 for the following description.

We consider k_1, k_2, k_3, k_4, such that $k_1 \in \mathcal{T}_1$, $k_2, k_4 \in \mathcal{T}_2$, and $k_3 \in \mathcal{T}_3$. It is clear that $k_1 + k_2 \in \mathcal{S}_1$ and $k_3 + k_4 \in \mathcal{S}_2$ (see Figure 1). We create the list L_1 of (P_1, k_1) such that $P_1 = k_1 P$ for all $k_1 \in \mathcal{T}_1$, the list L_2 of (P_2, k_2) such that $P_2 = k_2 P$ for all $k_2 \in \mathcal{T}_2$, the list L_3 of (P_3, k_3) with $P_3 = Q - k_3 P$ for all $k_3 \in \mathcal{T}_3$, and the list L_4 of (P_4, k_4) with $P_4 = -k_4 P$ for all $k_4 \in \mathcal{T}_2$. From here, we compute the list $L = \{(P_{12}, k_1 + k_2) : (P_1, k_1) \in L_1,\ (P_2, k_2) \in L_2,\ P_{12} = P_1 + P_2 = (x, y) \text{ with } x \in \mathbb{F}_p\}$. Then for all $(x', y') = P_3 + P_4$ with $x' \in \mathbb{F}_p$ we check if $(x', y') \in L$, and if so return the sum of the corresponding k_1, k_2, k_3, k_4. In other words, we consider the following problem.

[1] If this approach fails to produce a representation of k, we could rerandomize our ECDLP-instance and start over.

Algorithm 1 `BuildLists`$(\mathbf{E}, P)$

Require: An elliptic curve $\mathbf{E}$ over $\mathbb{F}_{p^2}$, a point $P \in \mathbf{E}(\mathbb{F}_{p^2})$.
Ensure: L_1, L_2, L_3, L_4.

1: **for** $i \in [1, 4]$ **do**
2: $\quad L_i \leftarrow \perp$
3: $(k_1, k_2) \leftarrow (1, 0)$
4: $P_2 = O$
5: **for** $1 \leq i \leq p^{\lambda}$ **do** ▷ Build lists L_1 and L_3
6: $\quad P_1 \leftarrow P + P_2$
7: $\quad$ **for** $1 \leq j < p^{\frac{1}{2}}$ **do**
8: $\quad\quad L_1 \leftarrow L_1 \cup \{(P_1, k_1)\}$
9: $\quad\quad P_1 \leftarrow P_1 + P$; $k_1 \leftarrow k_1 + 1$
10: $\quad$ **if** $i = 1$ **then** ▷ Only on the first iteration
11: $\quad\quad P_0 \leftarrow P_1$, $k_0 \leftarrow k_1$
12: $\quad P_2 \leftarrow P_1$; $k_2 \leftarrow k_1$
13: $\quad$ **for** $1 \leq j < p^{\frac{1}{2}}$ **do**
14: $\quad\quad L_3 \leftarrow L_3 \cup \{(Q - P_2, k_2)\}$
15: $\quad\quad P_2 \leftarrow P_2 + P_0$; $k_2 \leftarrow k_2 + k_0$
16: $\quad$ **if** $i < p^{\lambda}$ **then** ▷ In all iterations except the last one
17: $\quad\quad L_1 \leftarrow L_1 \cup \{(P_2, k_2)\}$
18: $\quad\quad L_3 \leftarrow L_3 \cup \{(Q - P_2, k_2)\}$
19: $P_1 \leftarrow P_2$, $k_1 \leftarrow k_2$
20: **for** $0 \leq i < rp^{-(1+\lambda)}$ **do** ▷ Build lists L_2 and L_4
21: $\quad L_2 \leftarrow L_2 \cup \{(P_2, k_2)\}$
22: $\quad L_4 \leftarrow L_4 \cup \{(-P_2, k_2)\}$
23: $\quad P_2 \leftarrow P_2 + P_1$, $k_2 \leftarrow k_2 + k_1$

Problem 1. Given two lists L_1 and L_2 of points $P \in \mathbf{E}(\mathbb{F}_{p^2})$, compute the list $L = \{(x, y) = P_1 + P_2 \ P_1 \in L_1, \ P_2 \in L_2, \ x \in \mathbb{F}_p\}$

We claim that any algorithm solving Problem 1 can be used as the main routine to solve the p^2-`ECDLP` problem. Our whole `RepECDLP` algorithm is summarised in Algorithm 2. The `BuildLists` algorithm, described in Algorithm 1 builds the lists L_i for $i \in [1, 4]$ and we denote by `ECJoin` an algorithm solving Problem 1, with a slight modification that does not influence the run time: the input lists contain tuples (P_i, k_i), where k_i is an element of $[0, r)$, the output list must also contain tuples $(P_1 + P_2, k_1 + k_2)$.

The following lemma gives the complexities of `BuildLists`.

Lemma 2. *The* `BuildLists` *procedure constructs lists of sizes*

$$|L_1| = |L_3| = p^{\frac{1}{2}+\lambda} \text{ and } |L_2| = |L_4| = p^{(1-\lambda)}$$

in $\mathcal{O}\left(\max\left(p^{\frac{1}{2}+\lambda}, p^{(1-\lambda)}\right)\right)$ *field operations using memory* $\mathcal{O}\left(\max\left(p^{\frac{1}{2}+\lambda}, p^{(1-\lambda)}\right)\right)$.

Proof. We start by proving the time complexity. It is dominated by the runtime of the two main for-loops: the one at line 5, which fills L_1 and L_3, and the one

Algorithm 2 `RepECDLP`$(\mathbf{E}, P, Q)$

Require: An elliptic curve $\mathbf{E}$ over $\mathbb{F}_{p^2}$, two points $P, Q \in \mathbf{E}(\mathbb{F}_{p^2})$.
Ensure: k such that $Q = kP$.
1: $(L_1, L_2, L_3, L_4) \leftarrow$ `BuildLists`$(\mathbf{E}, P)$
2: $L \leftarrow$ `ECJoin`(L_1, L_2)
3: $L' \leftarrow$ `ECJoin`(L_3, L_4)
4: **for all** P_{34} : $\exists k_{34}, (P_{34}, k_{34}) \in L'$ **do**
5: **if** $\exists (P_{12}, k_{12}) \in L$ such that $P_{34} = P_{12}$ **then**
6: **return** $k_{12} + k_{34}$
return $\perp$

at line 20, which fills L_2 and L_4. The first one is iterated p^λ times, and each iteration takes time $p^{1/2}$. Indeed, each iteration consists in a constant number of elementary operations in $\mathbb{F}_p$ and two for-loops which are each iterated $p^{1/2}$, and whose iterations consist in $\mathcal{O}(1)$ elementary operations in $\mathbb{F}_p$. Thus the whole execution of the for-loop line 5 requires to perform $\mathcal{O}\left(p^{\frac{1}{2}+\lambda}\right)$ operations. The for-loop line 20 is iterated $rp^{-(1+\lambda)}$ times, and each iterations consists in a constant number of elementary operations over $\mathbb{F}_p$. The claimed complexity follows.

We now show that the memory required is indeed $\mathcal{O}\left(\max(p^{\frac{1}{2}+\lambda}, rp^{-(1+\lambda)})\right)$. The memory complexity is dominated by the storage of the four lists. Now, for $i \in [1,4]$, L_i contains $\mathcal{O}(|L_i|)$ elements of $\mathbb{F}_p$. As such, the memory complexity of the whole procedure is $\mathcal{O}(\max_i(|L_i|))$. Furthermore, $|L_1| = |L_3| = p^{\frac{1}{2}+\lambda}$ and $|L_2| = |L_4| = rp^{-(1+\lambda)}$. As $r = \mathcal{O}(p^2)$, $|L_2| = |L_4| = \mathcal{O}(p^{1-\lambda})$, which proves the given complexities. □

Remark 1. Notice that the choice $\lambda = \frac{1}{4}$ implies that all lists have size $\mathcal{O}\left(p^{\frac{3}{4}}\right)$. However, as we will discuss in section 4, unbalanced list sizes might lead to time improvements.

3.2 Computing the Join

We now need to find a way to check whether a point $(x, y) = P_1 + P_2$ satisfies $x \in \mathbb{F}_p$, knowing only the coordinates (x_1, y_1) of P_1 and (x_2, y_2) of P_2. We have

$$x = \frac{(y_1 - y_2)^2}{(x_1 - x_2)^2} - x_1 - x_2,$$
$$\text{and therefore} \quad (x_1 - x_2)^2(x_1 + x_2 + x) = y_1^2 + y_2^2 - 2y_1y_2$$

Using Weierstraß' equation to discard the y_1 and y_2 terms, we obtain

$$\left((x_1 - x_2)^2(x_1 + x_2 + x) - f(x_1) - f(x_2)\right)^2 = 4f(x_1)f(x_2). \tag{2}$$

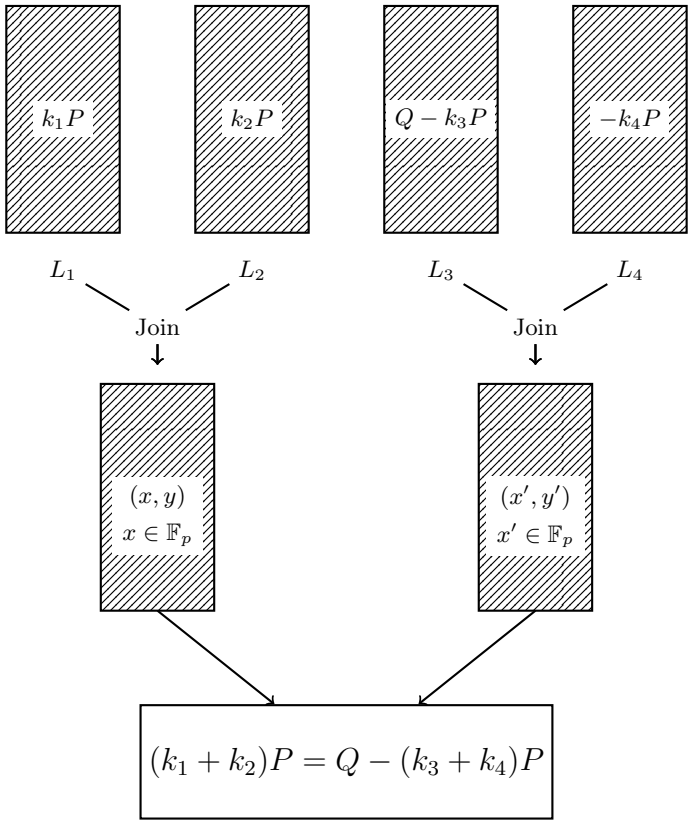


Fig. 2: Illustration of `RepECDLP`.

We denote $x = x^{(0)} + \alpha x^{(1)}$ where $x^{(0)}, x^{(1)}$ are in $\mathbb{F}_p$ and where α satisfies $\alpha^2 = \beta$ for some β quadratic non-residue modulo p, and we denote $x_1 = x_1^{(0)} + \alpha x_1^{(1)}$, $x_2 = x_2^{(0)} + \alpha x_2^{(1)}$ accordingly. Having $x \in \mathbb{F}_p$ implies $x^{(1)} = 0$. Let us denote by $\mathsf{G} \in \mathbb{F}_{p^2}[Y_1, Y_2, Y_3]$ the polynomial

$$\mathsf{G}(Y_1, Y_2, Y_3) = ((Y_1 - Y_2)^2(Y_1 + Y_2 + Y_3) - f(Y_1) - f(Y_2))^2 - 4f(Y_1)f(Y_2).$$

As each Y_i can be expressed as $Y_i = X_{2i-1} + \alpha X_{2i}$ where the X_j variables are over $\mathbb{F}_p$. It follows that there exists a pair of unique polynomials g_0 and g_1 of $\mathbb{F}_p[X_1 \dots X_6]$ such that

$$\mathsf{g}_0(X_1, \dots, X_6) + \alpha \mathsf{g}_1(X_1 \dots X_6) = \mathsf{G}(X_1 + \alpha X_2, X_3 + \alpha X_4, X_5 + \alpha X_6).$$

Now, for any $x_1, x_2, x \in \mathbb{F}_{p^2}$ satisfying Equation (2)

$$\mathsf{g}_0(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, \dots, x^{(1)}) + \alpha \mathsf{g}_1(x_1^{(0)}, \dots, x^{(1)}) = 0.$$

Taking into account that $x = x^{(0)} \in \mathbb{F}_p$, we come up with the following polynomial system in five variables

$$\begin{cases} \mathsf{g}'_0\left(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}, x^{(0)}\right) := \mathsf{g}_0\left(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}, x^{(0)}, 0\right) = 0 \\ \mathsf{g}'_1\left(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}, x^{(0)}\right) := \mathsf{g}_1\left(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}, x^{(0)}, 0\right) = 0 \end{cases}.$$

Computing the elimination ideal of $\langle \mathsf{g}'_0, \mathsf{g}'_1 \rangle$, in the variable $x^{(0)}$, we obtain a polynomial f of constant degree such that for all $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$ summing up to (x, y) with $x \in \mathbb{F}_p$, we have

$$\mathsf{f}\left(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}\right) = 0.$$

Remark 2. Speaking in terms of ideal and algebraic varieties, we claim that $(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}) \in V(\langle \mathsf{f} \rangle) = \{(z_1, \ldots, z_4) \in \mathbb{F}_p^4,\ \mathsf{f}(z_1, \ldots z_4) = 0\}$. This comes from the fact that by definition $\langle \mathsf{f} \rangle \subseteq \langle \mathsf{g}'_0, \mathsf{g}'_1 \rangle$, It follows that $V(\langle \mathsf{g}'_0, \mathsf{g}'_1 \rangle) \subseteq V(\langle \mathsf{f} \rangle)$. Since by construction of g'_0 and g'_1, $(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}) \in V(\langle \mathsf{g}'_0, \mathsf{g}'_1 \rangle)$, our claim follows.

We are now ready to define the `ZJ-Problem`, which lies at the heart of our ECDLP algorithm.

Problem 2 (`ZJ-Problem`). Given two lists A and B respectively of points (x_A, y_A) and (x_B, y_B) in $\mathbb{F}_p^2$, and given a polynomial f of $\mathbb{F}_p[X_1, \ldots X_4]$ compute the list C of all $((x_A, y_A), (x_B, y_B)) \in A \times B$ such that $\mathsf{f}(x_A, y_A, x_B, y_B) = 0$.

We claim that any algorithm solving the `ZJ-Problem` can be used as the main routine to solve Problem 1. Indeed, given our lists L_1 and L_2 respectively of points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, and given the polynomial f defined as above, we compute the list A of all $\left(x_1^{(0)}, x_1^{(1)}\right)$ where $x_1 = x_1^{(0)} + \alpha x_1^{(1)}$. Then we compute the list B of all $\left(x_2^{(0)}, x_2^{(1)}\right)$ where $x_2 = x_2^{(0)} + \alpha x_2^{(1)}$. Now, let us denote by `ZeroJoin` any algorithm solving the `ZJ-Problem`. Let $C = \texttt{ZeroJoin}(A, B, \mathsf{f})$.

We claim that for every P_1, P_2 satisfying $P_1 + P_2 = (x, y)$ with $x \in \mathbb{F}_p$, $\left(\left(x_1^{(0)}, x_1^{(1)}\right), \left(x_2^{(0)}, x_2^{(1)}\right)\right) \in C$. This comes from the definition of f.

However, there may be false positives — meaning tuples $\left(\left(x_1^{(0)}, x_1^{(1)}\right), \left(x_2^{(0)}, x_2^{(1)}\right)\right)$ for which $\mathsf{f}\left(x_1^{(0)} \ldots x_2^{(1)}\right) = 0$ holds, but $P_1 + P_2 = (x, y)$ with $x \notin \mathbb{F}_p$. Therefore, we need to check the corresponding sum (x, y), while computing the list L. In other words, for all $\left(\left(x_1^{(0)}, x_1^{(1)}\right), \left(x_2^{(0)}, x_2^{(1)}\right)\right) \in C$, we first retrieve the corresponding $(P_1, P_2) \in A \times B$. Then we compute $(x, y) = P_1 + P_2$. If $x \in \mathbb{F}_p$, we add (x, y) to L.

This is summarized in Algorithm 3. The following lemma provides a reduction from the `ZJ-Problem` to Problem 1.

Algorithm 3 $\texttt{ECJoin}(\mathbf{E}, \mathsf{f}, L_1, L_2)$

Require: An elliptic curve $\mathbf{E}$ over $\mathbb{F}_{p^2}$, two lists of point L_1, L_2, the polynomial f associated to $\mathbf{E}$.

Ensure: $L = \{(P_0, k_1 + k_2) : P_0 = P_1 + P_2 = (x, y),\ (P_i, k_i) \in L_i, i \in [1, 2],\ x \in \mathbb{F}_p\}$.

$A, B, L \leftarrow \perp$
for all $(P_1, k_1) \in L_1$ **do**
    $A \leftarrow A \cup \{((x_1^{(0)}, x_1^{(1)}), k_1)\}$, where $P_1 = (x_1, y_1)$
for all $(P_2, k_2) \in L_2$ **do**
    $B \leftarrow B \cup \{((x_2^{(0)}, x_2^{(1)}), k_2)\}$, where $(x_2, y_2) = P_2$
$C \leftarrow \texttt{ZeroJoin}(A, B, \mathsf{f})$
for all $(i, j) \in C$ **do**
    $(P_1, k_1) \leftarrow L_1[i]$, $(P_2, k_2) \leftarrow L_2[j]$
    $P_0 = (x, y) \leftarrow P_1 + P_2$
    if $x \in \mathbb{F}_p$ **then**
        $L \leftarrow L \cup \{(P_0, k_1 + k_2)\}$

Lemma 3. *If* `ZeroJoin` *solves the* `ZJ-Problem` *in time T and memory M, The* `ECJoin` *algorithm solves Problem 1 in time T and memory M.*

Proof. Let us denote by $T_{\texttt{ECJoin}}$ the time complexity of the `ECJoin` algorithm. This algorithm consists of three for-loops and the `ZeroJoin` procedure. The first for-loop is iterated $|L_1|$ times, and each iteration takes time $\mathcal{O}(1)$. The second for-loop is iterated $|L_2|$ times. Once again each iteration requires a constant number of field operations. Then comes the `ZeroJoin` procedure which has a runtime T. Finally, the last for-loop requires to perform $|C|$ constant time iterations. It follows that:

$$T_{\texttt{ECJoin}} = \mathcal{O}\left(\max(|L_1|, |L_2|, T, |C|)\right).$$

It is clear that $T \geq |C|$, as the list C is built during the `ZeroJoin` procedure. We claim that T also satisfies $T \geq |L_1| + |L_2|$, as each of the $|L_1|$ polynomial and each of the $|L_2|$ point have to be considered at least once in order to create C. It follows that

$$T_{\texttt{ECJoin}} = T.$$

Let us focus on the memory. We denote by $M_{\texttt{ECJoin}}$ the memory required by the `ECJoin` procedure. We have:

$$M_{\texttt{ECJoin}} = \mathcal{O}\left(\max(|L_1|, |L_2|, M, |C|)\right).$$

Once again it is clear that $M \geq |C|$. We also claim that $M \geq |L_1| + |L_2|$ as our algorithm requires to store lists A and B of respective size $|L_1|$ and $|L_2|$. It follows that

$$M_{\texttt{ECJoin}} = M.$$

□

We now reduce the `ZJ-Problem` to p^2-`ECDLP`.

Theorem 1. *Under our heuristic, if there exists an algorithm* `ZeroJoin` *solving the* `ZJ-Problem` *in time T and memory M, then* `RepECDLP` *solves the p^2-*`ECDLP` *problem in time T and memory M.*

Proof. We start by proving the time complexity. The `RepECDLP` algorithm consists basically of three steps. In the first step the lists L_i for $i \in [1,4]$ are created using the `BuildLists` procedure. According to Lemma 2 this can be done in time $\mathcal{O}(\max_i(|L_i|))$. The second step consists in creating both the join of L_1 and L_2, and of L_3 and L_4. According to Lemma 3, this takes time T. It remains to search for a collision between two lists. This can be done in time proportional to the size of the two lists provided that they are sorted according to the first component. Under our heuristic, we can assume that the points (x,y) contained in the lists are uniformly distributed over $\mathbb{F}_p \times \mathbb{F}_{p^2}$, as such the sorting step can be done in time proportional to the size of lists. Recall that $T \geq \max(|L_i|, |L|, |L'|)$ for the same argument than the one given in the proof of Lemma 3. It follows that the time complexity of the whole algorithm is dominated by T.

We claim that the memory complexity is dominated by the one of the `ECJoin` routines. Indeed as explained before, the space M required by this routine is at least $\mathcal{O}\left(\max(p^{\frac{1}{2}+\lambda}, p^{1-\lambda}, |L|, |L'|)\right)$, which is enough to prove our claim.

For completeness, we now prove that this procedure actually solves the p^2-`ECDLP` problem. For simplicity, we consider only the first component of the lists elements. By construction

$$L_i = \begin{cases} \{k_i P,\ k_i \in \mathcal{T}_i\} & \text{if } i \in \{1,3\} \\ \{Q - k_i P,\ k_i \in \mathcal{T}_i\} & \text{if } i = 2 \\ \{-k_i P,\ k_i \in \mathcal{T}_i\} & \text{if } i = 4. \end{cases}$$

We claim that $L = \texttt{ECJoin}(L_1, L_2)$ is the list

$$L = \{k_{12}P = (x,y),\ k_{12} \in \mathcal{S}_1,\ x \in \mathbb{F}_p\}. \tag{3}$$

Indeed, by construction, it is obvious that $\forall (x,y) \in L,\ x \in \mathbb{F}_p$. Furthermore $(x,y) = k_{12}P$ for some $k_{12} \in \mathcal{S}_1$ as $(x,y) = k_1 P + k_2 P = (k_1 + k_2)P$ for some $(k_1, k_2) \in L_1 \times L_2$. It follows that $L \subseteq \{k_{12}P = (x,y),\ k_{12} \in \mathcal{S}_1,\ x \in \mathbb{F}_p\}$. We show that this is indeed an equality.

Recall that $\mathsf{f} \in \mathbb{F}_p[X_1, X_2, X_3, X_4]$ is constructed so that, for all points $P_1 = (x_1, y_1),\ P_2 = (x_2, y_2) \in \mathbf{E}(\mathbb{F}_{p^2})$, if $(x,y) = P_1 + P_2$ satisfies $x \in \mathbb{F}_p$ then $\mathsf{f}(x_1^{(0)}, x_1^{(1)}, x_2^{(0)}, x_2^{(1)}) = 0$. Now for any $P_i = (x_i, y_j)$, let us consider the polynomial $\mathsf{f}_i \in \mathbb{F}_p[X,Y]$ such that $\mathsf{f}_i(X,Y) = \mathsf{f}(x_i^{(0)}, x_i^{(1)}, X, Y)$. It is clear that, for all $P_j = (x_j, y_j)$ such that $P_i + P_j = (x,y)$ with $x \in \mathbb{F}_p$, $\mathsf{f}_i(x_j^{(0)}, x_j^{(1)}) = 0$. As A is the set of the f_i polynomials for all $(x_i, y_i) \in L_1$, and B is the set of the $(x_j^{(0)}, x_j^{(1)})$ tuples for all $(x_j, y_j) \in L_2$, it follows, from the definition of the `ZeroJoin` procedure, that if $P_i + P_j = (x,y)$ satisfies that $x \in \mathbb{F}_p$, then $(i,j) \in C = \texttt{ZeroJoin}(A, B)$. As such $\{k_{12}P = (x,y),\ k_{12} \in \mathcal{S}_1,\ x \in \mathbb{F}_p\} \subseteq \{P_i + P_j,\ (i,j) \in C\}$, only those points which indeed satisfy $x \in \mathbb{F}_p$ are kept in

L. A similar argument is used to show that

$$L' = \{Q - k_{34}P = (x', y'),\ k_{34} \in \mathcal{S}_2,\ x \in \mathbb{F}_p\}.$$

It remains to show that a representation (k_{12}, k_{34}) of our solution k is contained in the list $L \times L'$. We argue that our algorithm succeeds in solving p^2-ECDLP if there is a representation $(k_{12}, k_{34}) \in \mathcal{S}_1 + \mathcal{S}_2$ of k such that $k_{12}P = (x, y)$ with $x \in \mathbb{F}_p$.

It is clear that if there is no such representation of k our procedure fails, as only $k_{12} \in \mathcal{S}_1$ for which $k_{12}P = (x, y)$ with $x \in \mathbb{F}_p$ are considered. Now, if such a k_{12} exists, according to Equation 3, $k_{12}P$ is in L. Similarly, if k_{34} satisfies that $Q - k_{34}P = (x', y')$ with $x' \in \mathbb{F}_p$, then k_{34} is in L'. Furthermore, as $k_{12}P = Q - k_{34}P$, it is enough to know that k_{12} satisfying $k_{12}P = (x, y)$ with $x \in \mathbb{F}_p$ exists, to ensure that the algorithm recovers the solution.

We also claim that if the elements of $\mathbf{E}(\mathbb{F}_{p^2})$ are uniformly distributed (as we assume by our heuristic), then on expectation, there exists a representation (k_{12}, k_{34}) which satisfies $k_{12}P = (x, y)$ with $x \in \mathbb{F}_p$. First note that the uniform distribution implies

$$\mathbb{P}[x \in \mathbb{F}_p | (x, y) = \ell P, \ell \in \mathcal{S}_1] = \mathbb{P}[x \in \mathbb{F}_p] = \frac{1}{p}.$$

We denote by $N_{\mathtt{rep}}$ the number of representation of k as the sum $k_{12} + k_{34}$, $k_{12} \in \mathcal{S}_1$, $k_{34} \in \mathcal{S}_2$. In other words $N_{\mathtt{rep}}$ is the number of elements $k_{12} \in \mathcal{S}_1$ such that there exists $k_{34} \in \mathcal{S}_2$, with $k_{12} + k_{34} = k$. We denote by Y the number of surviving representations in $L + L'$. In other words, Y is the number of $k_{12} \in \mathcal{S}_1$ such that there exists $k_{34} \in \mathcal{S}_2$ with $k_{12} + k_{34} = k$, and such that $k_{12}P = (x, y)$ with $x \in \mathbb{F}_p$. By Lemma 1, we have

$$\mathbb{E}[Y] = N_{\mathtt{rep}} \frac{1}{p} > \frac{r}{p^2}.$$

By Hasse's theorem, $r \geq p^2 - 2p$ and therefore for sufficiently large p we expect that one representation survives. □

4 Solving the ZJ-Problem

The running time of our new p^2-ECDLP algorithm is dominated by solving the ZJ-Problem. Recall that in the ZJ-Problem we are given two lists A and B consisting respectively of N points $(x_{11}, y_{11}) \dots (x_{1N}, y_{1N}) \in \mathbb{F}_p^2$ and M points $(x_{21}, y_{21}) \dots (x_{2M}, y_{2M}) \in \mathbb{F}_p^2$, and a polynomial f for which we have to find all roots $((x_{1i}, y_{1i}), (x_{2j}, y_{2j}))$ in our lists, that is $\mathsf{f}(x_{1i}, y_{1i}, x_{2j}, y_{2j}) = 0$.

Naively, we can solve the ZJ-Problem in time $T = \mathcal{O}(NM)$. Namely, from the first list A of points (x_i, y_i) we construct a list $\bar{A}$ of bivariate polynomials

$$\mathsf{f}_i(X, Y) = \mathsf{f}(x_{1i}, y_{1i}, X, Y).$$

Then we successively evaluate all $\mathsf{f}_i(X,Y)$ in all M points $(x_1,y_1)\dots(x_M,y_M) \in \mathbb{F}_p^2$ from the second list B. As each polynomial has constant degree, each of the NM evaluations costs time $\mathcal{O}(1)$. As a result we obtain a list C of all the $(f_i,(x_j,y_j)) \in \bar{A} \times B$, such that $f_i(x_j,y_j) = 0$. Since we have $NM = p^{\frac{3}{2}}$, this gives us an ECDLP algorithm with run time $\mathcal{O}\left(p^{\frac{3}{2}}\right)$.

Fast Polynomial multiplication. Let $I \subset [1,M]$. Obviously, if

$$\mathsf{F}_I(X,Y) = \prod_{i\in I} \mathsf{f}_i(X,Y) = 0,$$

then there is an $i \in I$ with $\mathsf{f}_i(x,y) = 0$. This gives rise to the following algorithm.

1. Partition the set $\bar{A}$ of polynomials into buckets $A_{I_1}, A_{I_2}, \dots A_{I_k}$, for some $k \geq 1$, such that $\mathsf{f}_i \in A_{I_\ell}$ iff $i \in I_\ell$.
2. For each I_ℓ, compute the set B_{I_ℓ} of points $(x_j,y_j) \in B$, such that $\mathsf{F}_{I_\ell}(x_i,y_j) = 0$.
3. For each $\mathsf{f}_i \in A_{I_\ell}$, find all $(x_j,y_j) \in B_{I_\ell}$, such that $\mathsf{f}_i(x_j,y_j) = 0$.

Our solution uses fast bivariate polynomial multi-point evaluation, that has on optimal choice $N = \sqrt{p}$, $M = p$, for which only a single $\mathsf{F}_I = \mathsf{F}$ has to be computed. This special case is given in Algorithm 4. The following multi-point evaluation result is due to Nüsken and Ziegler [NZ04].

Lemma 4 ([NZ04, Result 4]). *For any fixed $\epsilon > 0$, a bivariate polynomial $f \in \mathbb{F}_p[X,Y]$ of degree in $X,Y \leq d$, specified by its coefficients, can be evaluated simultaneously at M given points $(x,y) \in \mathbb{F}_p^2$, with pair-wise different x-coordinates using $\mathcal{O}\left((M+d^2)d^{\frac{\omega_2}{2}-1+\epsilon}\right)$ operations in $\mathbb{F}_p$, where $\omega_2 \leq 3.257$ is the exponent of $n \times n$ by $n \times n^2$ matrices multiplications.*

Algorithm 4 `MultiEval`$(\bar{A}, B)$

Require: A list A of $\sqrt{N}$ polynomials of $\mathbb{F}_p[X,Y]$ and a list B of p points of $\mathbb{F}_p^2$.
Ensure: The list C of all the $(\mathsf{f}_i,(x_j,y_j)) \in \bar{A} \times B$ such that $\mathsf{f}_i(x_j,y_j) = 0$.

$B', C \leftarrow \perp$
$\mathsf{F} \leftarrow \prod_{i=1}^{\sqrt{p}} \mathsf{f}_i$.
$E \leftarrow$ `BivariatePolynomialMultipointEvaluation`(F, B)
for all $1 \leq j \leq p$: $E[j] = 0$ **do**
    $B' \leftarrow B' \cup \{(x_j,y_j)\}$
for all $\mathsf{f}_i \in \bar{A}$ **do**
    for all $(x_j,y_j) \in B'$ **do**
        if $\mathsf{f}_i(x_j,y_j) = 0$ **then**
            $C \leftarrow C \cup (\mathsf{f}_i,(x_j,y_j))$
return C

We make use of the Schwartz-Zippel Lemma, from which we derive an upper bound on the number of zeroes of F in B in Algorithm 4.

Lemma 5 (Schwartz-Zippel). *Given a polynomial* $\mathsf{F} \in \mathbb{F}_p[X, Y]$ *of degree at most* d *in* X, Y*, a random point* $(x, y) \in \mathbb{F}_p^2$ *is a zero of* F *with probability*

$$\mathbb{P}[\mathsf{F}(x, y) = 0] \leq \frac{d}{p}.$$

Theorem 2. *RepECDLP in combination with* `MultiEval` *solves* p^2*-ECDLP in time* $T = \tilde{\mathcal{O}}\left(p^{1.314}\right)$.

Proof. We start by showing that the time complexity of `MultiEval` is dominated by the bivariate polynomial multi-point evaluation line 3. At this aim, we denote by $T_{\mathsf{F}}, T_E, T_{B'}$ and T_C respectively the time complexity required to compute F at line 2, to perform the multi-point evaluation at line 3, to execute the for-loop in line 4 and to execute the for-loop in line 6. The total runtime of the `MultiEval` procedure is given by

$$T = \max(T_{\mathsf{F}}, T_E, T_{B'}, T_C). \tag{4}$$

First we estimate T_{F}. Notice that the degree of F in X and in Y is bounded by a constant c times $\sqrt{p}$, since F is the product of $\sqrt{p}$ polynomials of constant degree in X and Y. It follows that F can be computed in $T_{\mathsf{F}} = \tilde{\mathcal{O}}\left(p\right)$ operations, using fast polynomial multiplication algorithms, where logarithmic factors are hidden in the $\tilde{\mathcal{O}}$.

Next, we have to evaluate this polynomial simultaneously in all the p points. From Lemma 4, this takes time $\mathcal{O}\left(p^{\frac{1}{2}(1+\frac{\omega_2}{2})+\epsilon}\right)$ for any fixed $\epsilon > 0$.

There is a small twist, however as lemma 4 can only be applied for a list of points (x, y) with pair-wise different x-coordinates. Considering that all x are in $\mathbb{F}_p$ and that $|B| = p$, this condition implies that all elements of $\mathbb{F}_p$ are present once and only once as the x-coordinate of an element of B. This is very unlikely. In fact, we proceed as follows. We partition B according to the x-coordinate and apply the Nüsken-Ziegler algorithm with one element of each partition of B. We restart until all elements of each partitions have been processed.

The number of time we have to restart is bounded by the size of the largest partition of B. If the x-coordinated are uniformly distributed over $\mathbb{F}_p$, we can estimate this size, using maximum-load results [Mit96]. We obtain that the number of time we have to restart is with high probability bounded by a constant times $\frac{\log p}{\log \log p}$. Replacing ω_2 by the best known bound 3.257 [LG14], and for $\epsilon < 10^{-4}$, its follows that $T_E = \tilde{\mathcal{O}}\left(p^{1.314}\right)$.

We may assume that for each evaluated value $E[j]$ we kept track of the corresponding (x_j, y_j). Then building B' from E requires only to scan through each entry of E once, and add (x_j, y_j) in B' for all $E[j] = 0$ that are encountered. The runtime of this step is thus linear in $|E| = |B|$, and therefore $T_{B'} = \mathcal{O}\left(p\right)$.

The last step consists in evaluating all f_i simultaneously in all the point of B'. We proceed in a naive way by taking all the polynomials, one by one, and evaluating each of them simultaneously in all the point of B'. This results in the two entwined for-loops in line 6 and 7. The first one iterates over all the polynomials and thus is iterated $\sqrt{p}$ times. The second one iterates over all the

points of B' and thus is iterated $|B'|$ times. Each iteration of the second loop can be performed in $\mathcal{O}(1)$ as all the f_i are of constant degree. It follows that $T_C = \mathcal{O}\left(\sqrt{p}|B'|\right)$.

It remains to estimate $|B'|$. From Lemma 5, for any random $(x, y) \in \mathbb{F}_p^2$

$$\mathbb{P}[\mathsf{F}(x, y) = 0] \leq \frac{\sqrt{p}}{p}.$$

Assuming that the points of B are uniformly distributed over $\mathbb{F}_p^2$, it follows that the expected size Z of B' satisfies

$$\mathbb{E}[Z] = \sqrt{p},$$

and thus the expected time complexity of this third step is $\mathcal{O}(p)$. From Equation 4, it follows that

$$T = \mathcal{O}\left(\max(p, p^{1.314})\right) = \mathcal{O}\left(p^{1.314}\right).$$

Building the list $\bar{A}$ of the polynomials f_i from the list A of points (x_i, y_i) and the polynomial f can be done in time linear in $|A| = \sqrt{p}$, since f has constant total degree. Thus the `ZJ-Problem` can be solved in time T too. From Theorem 1, we can conclude that `RepECDLP` solves p^2-`ECDLP` in time $\mathcal{O}\left(p^{1.314}\right)$. □

We hope that the result of Theorem 2 may be further improved. A possible direction is to replace the (unnecessary) multi-evaluation of F by some presumably more efficient zero testing method. An other research direction, as pointed out by one of the reviewers, could be to have a look at a different elliptic curve model than the one of Weierstraß (e.g. Edwards Curve). However, we are not sure whether the problem would become easier in this case.

Acknowledgement The authors would like to thank the reviewers for their comments, as well as Andre Esser for proof reading this paper.

References

BCJ11. Anja Becker, Jean-Sébastien Coron, and Antoine Joux. Improved generic algorithms for hard knapsacks. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 364–385, Tallinn, Estonia, May 15–19, 2011. Springer, Heidelberg, Germany.

FG98. Gerhard Frey and Herbert Gangl. How to disguise an elliptic curve (Weil descent). *Talk at ECC*, 98:128–161, 1998.

Gau09. Pierrick Gaudry. Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. *Journal of Symbolic Computation*, 44(12):1690 – 1702, 2009. Gröbner Bases in Cryptography, Coding Theory, and Algebraic Combinatorics.

GG16. Steven D Galbraith and Pierrick Gaudry. Recent progress on the elliptic curve discrete logarithm problem. *Designs, Codes and Cryptography*, 78(1):51–72, 2016.

GHS02a. Steven D. Galbraith, Florian Hess, and Nigel P. Smart. Extending the GHS Weil descent attack. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 29–44, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Heidelberg, Germany.

GHS02b. Pierrick Gaudry, Florian Hess, and Nigel P. Smart. Constructive and destructive facets of Weil descent on elliptic curves. *Journal of Cryptology*, 15(1):19–46, January 2002.

GS99. Steven D. Galbraith and Nigel P. Smart. A cryptographic application of Weil descent. In Michael Walker, editor, *7th IMA International Conference on Cryptography and Coding*, volume 1746 of *Lecture Notes in Computer Science*, pages 191–200, Cirencester, UK, December 20–22, 1999. Springer, Heidelberg, Germany.

HGJ10. Nick Howgrave-Graham and Antoine Joux. New generic algorithms for hard knapsacks. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 235–256, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany.

KT19. Taechan Kim and Mehdi Tibouchi. Equidistribution among cosets of elliptic curve points in intervals, 2019. Accepted at NutMic 2019.

LG14. François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303. ACM, 2014.

Mit96. Michael David Mitzenmacher. *The power of two random choices in randomized load balancing*. PhD thesis, PhD thesis, Graduate Division of the University of California at Berkley, 1996.

NZ04. Michael Nüsken and Martin Ziegler. Fast multipoint evaluation of bivariate polynomials. In *European Symposium on Algorithms*, pages 544–555. Springer, 2004.

PKM16. Christophe Petit, Michiel Kosters, and Ange Messeng. Algebraic approaches for the elliptic curve discrete logarithm problem over prime fields. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016: 19th International Conference on Theory and Practice of Public Key Cryptography, Part II*, volume 9615 of *Lecture Notes in Computer Science*, pages 3–18, Taipei, Taiwan, March 6–9, 2016. Springer, Heidelberg, Germany.

SA$^+$98. Takakazu Satoh, Kiyomichi Araki, et al. Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. *Rikkyo Daigaku sugaku zasshi*, 47(1):81–92, 1998.

Sch85. René Schoof. Elliptic curves over finite fields and the computation of square roots mod p. *Mathematics of computation*, 44(170):483–494, 1985.

Sch95. René Schoof. Counting points on elliptic curves over finite fields. *Journal de théorie des nombres de Bordeaux*, 7(1):219–254, 1995.

Sem98. Igor Semaev. Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p. *Mathematics of Computation of the American Mathematical Society*, 67(221):353–356, 1998.

Sem04. Igor Semaev. Summation polynomials and the discrete logarithm problem on elliptic curves. Cryptology ePrint Archive, Report 2004/031, 2004. http://eprint.iacr.org/2004/031.

Sha71. Daniel Shanks. Class number, a theory of factorization, and genera. In *Proc. of Symp. Math. Soc., 1971*, volume 20, pages 41–440, 1971.

Sma99. Nigel P. Smart. The discrete logarithm problem on elliptic curves of trace one. *Journal of Cryptology*, 12(3):193–196, June 1999.