A Las Vegas algorithm to solve the elliptic curve discrete logarithm problem

Ayan Mahalanobis[1], Vivek Mohan Mallick[2], and Ansari Abdullah[3]

[1] IISER Pune, Pune INDIA ayan.mahalanobis@gmail.com
[2] IISER Pune, Pune INDIA vmallick@iiserpune.ac.in
[3] Savitribai Phule Pune University, Pune, INDIA abdullah0096@gmail.com

Abstract. In this paper, we describe a new Las Vegas algorithm to solve the elliptic curve discrete logarithm problem. The algorithm depends on a property of the group of rational points on an elliptic curve and is thus not a generic algorithm. The algorithm that we describe has some similarities with the most powerful index-calculus algorithm for the discrete logarithm problem over a finite field. The algorithm has no restriction on the finite field over which the elliptic curve is defined.

Keywords: Elliptic curve discrete logarithm problem

1 Introduction

Public-key cryptography is a backbone of this modern society. Many of the public-key cryptosystems depend on the *discrete logarithm problem* as their cryptographic primitive. Of all the groups used in a discrete logarithm based protocol, the group of *rational points of an elliptic curve* is the most popular. In this paper, we describe a **Las Vegas algorithm** to solve the elliptic curve discrete logarithm problem.

There are two kinds of attack on the discrete logarithm problem. One is generic. This kind of attack works in any group. Examples of such attacks are the baby-step giant-step attack [8, Proposition 2.22] and Pollard's rho [8, Section 4.5]. The other kind of attack depends on the group used. Example of such an attack is the index-calculus attack [8, Section 3.8] on the multiplicative group of a finite field. An attack similar to index-calculus for elliptic curves, known as xedni calculus, was developed by Silverman [9, 13]. However, it was found to be no better than exhaustive search. Another simailar work in the direction of ours is Semaev [11] which has given rise to index-calculus algorithms for elliptic curves. A curious reader can consult Amadori et. al. [1] for a list of references. Our approach to solve the elliptic curve discrete logarithm problem is completely different and has **no restriction on the finite field** on which the elliptic curve is defined.

The main algorithm is divided into two algorithms. The first one reduces the elliptic curve discrete logarithm problem to a problem in linear algebra. We call the linear algebra problem, Problem L. This reduction is a Las Vegas algorithm with **probability of success** 0.6 and is **polynomial** in both time and space complexity. The second half of the algorithm is solving Problem L. This is the current bottle-neck of the whole algorithm and better algorithms to solve Problem L will produce better algorithms to solve elliptic curve discrete logarithm problem. The success of the main algorithm is $0.6 \times (\log p)^2 / p$ where every pass is polynomial in time and space complexity. This also shows that our algorithm is worse than Pollard's rho or other square root attack for sufficiently large p.

1.1 Notations

All elliptic curves in this paper are non-singular curves defined over a finite field of arbitrary charactersitic.

All curves are projective plane curves. We do not deal with the affine case though that can be achieved with minor modification.

The group of rational points of the elliptic curve is assumed to be of prime order [8, Remark 2.33] and we reserve p for that prime.

We denote by $\mathbb{P}^2(\mathbb{F})$ the projective plane over the field $\mathbb{F}$.

1.2 The central idea behind our attack

Let G be a cyclic group of prime order p. Let P be a non-identity element and $Q(= mP)$ belong to G. The *discrete logarithm problem* is to compute the m mod p. One way to find m is the following: fix a positive integer k; for $i = 1, 2, \ldots, k$ find positive integers n_i, $1 \leq n_i < p$ such that $\sum_{i=1}^{k} n_i = m \bmod p$. The last equality is hard to compute because we do not know m. However we can decide whether

$$\sum_{i=1}^{k} n_i P = Q \tag{1}$$

and based on that we can decide if $\sum_{i=1}^{k} n_i = m \bmod p$. Once the equality holds, we have found m and the discrete logarithm problem is solved.

The number of possible choices of n_i for a given k that can solve the discrete logarithm problem is the number of partitions of m into k parts modulo a prime p. The applicability of the above method depends on, how fast can one decide on the equality in the above equation and on the probability, how likely is it that a given set of positive integers n_i sums to m mod p?

2 The elliptic curve discrete logarithm problem

The elliptic curve discrete logarithm problem (ECDLP) is an important problem in modern public-key cryptography. This paper is about a new probabilistic algorithm to solve this problem. We denote by $\mathcal{E}(\mathbb{F}_q)$ the group of rational points of the elliptic curve $\mathcal{E}$ over $\mathbb{F}_q$. It is well known that there is an isomorphism $\mathcal{E}(\mathbb{F}_q) \to \mathrm{Pic}^0(\mathcal{E})$ given by $P \mapsto [P] - [\mathcal{O}]$ [10, Proposition I.4.10].

Theorem 1. *Let $\mathcal{E}$ be an elliptic curve over $\mathbb{F}_q$ and $P_1, P_2, \ldots, P_k$ be points on that curve, where $k = 3n'$ for some positive integer n'. Then $\sum_{i=1}^{k} P_i = \mathcal{O}$ if and only if there is a curve $\mathcal{C}$ over $\mathbb{F}_q$ of degree n' that passes through these points. Multiplicities are intersection multiplicities.*

Proof. Assume that $\sum_{i=1}^{k} P_i = \mathcal{O}$ in $\mathbb{F}_q$ and then it is such in the algebraic closure $\bar{\mathbb{F}}_q$. From the above isomorphism, $\sum_{i=1}^{k} P_i \mapsto \sum_{i=1}^{k}[P_i] - k[\mathcal{O}]$. Then $\sum_{i=1}^{k}[P_i] - k[\mathcal{O}]$ is zero in the Picard group $\mathrm{Pic}^0_{\bar{\mathbb{F}}_q}(\mathcal{E})$. Then there is a rational function $\dfrac{\phi}{z^{n'}}$ over $\mathbb{P}^2\left(\bar{\mathbb{F}}_q\right)$ such that

$$\sum_{i=1}^{k}[P_i] - k[\mathcal{O}] = \operatorname{div}\left(\frac{\phi}{z^{n'}}\right) \tag{2}$$

Bezout's theorem justifies that $\deg(\phi) = n'$, since ϕ is zero on $P_1, P_2, \ldots, P_k$. We now claim, there is ψ over $\mathbb{F}_q$ which is also of degree n' and passes through $P_1, P_2, \ldots, P_k$. First thing to note is that there is a finite extension of $\mathbb{F}_q$, $\mathbb{F}_{q^N}$(say) in which all the coefficients of ϕ lies and $\gcd(q, N) = 1$. Let G be the Galois group of $\mathbb{F}_{q^N}$ over $\mathbb{F}_q$ and define

$$\psi = \sum_{\sigma \in \mathcal{G}} \phi^{\sigma}. \tag{3}$$

Clearly $\deg(\psi) = n'$. Note that, since P_i for $i = 1, 2, \ldots, k$ is in $\mathbb{F}_q$ is invariant under σ. Furthermore, σ being a field automorphism, P_i is a zero of ϕ^{σ} for all $\sigma \in G$. This proves that P_i are zeros of ψ and then Bezout's theorem shows that these are the all possible zeros of ψ on $\mathcal{E}$. The only thing left to show is that ψ is over $\mathbb{F}_q$. To see that, lets write $\phi = \sum_{i+j+k=n'} a_{ijk} x^i y^j z^k$. Then $\psi = \sum_{i+j+k=n'} \sum_{\sigma \in G} a_{ijk}^{\sigma} x^i y^j z^k$. However, it is well known that $\sum_{\sigma \in G} a^{\sigma} \in \mathbb{F}_q$ for all $a \in \mathbb{F}_{q^N}$.

Conversely, suppose we are given a curve $\mathcal{C}$ of degree n' that passes through $P_1, P_2, \ldots, P_k$. Then consider the rational function $\mathcal{C}/z^{n'}$. Then this function has zeros on $P_i,\ i = 1, 2, \ldots, k$ and a pole of order k at $\mathcal{O}$. The above isomorphism says $\sum_{i=1}^{k} P_i = \mathcal{O}$. □

2.1 How to use the above theorem in our algorithm

We choose k such that $k = 3n'$ for some positive integer n'. Then we choose random points $P_1, P_2, \ldots, P_s$ and $Q_1, Q_2, \ldots, Q_t$ such that $s + t = k$ from $\mathcal{E}$ and check if there is a homogeneous curve of degree n' that passes through these points where $P_i = n_i P$ and $Q_j = -n'_j Q$ for some integers n_i and n'_j. If there is a curve, the discrete logarithm problem is solved. Otherwise repeat the process by choosing a new set of points $P_1, P_2, \ldots, P_s$ and $Q_1, Q_2, \ldots, Q_t$. To choose these points P_i and Q_j, we choose random integers n_i, n'_j and compute $n_i P$ and $-n'_j Q$. We choose n_i and n'_j to be distinct from the ones chosen before. This gives rise to distinct points P_i and Q_j on $\mathcal{E}$.

The only question remains, how do we say if there is a homogeneous curve of degree n' passing through these selected points? One can answer this question using linear algebra.

Let $C = \sum_{i+j+k=n'} a_{ijk} x^i y^j z^k$ be a *complete* homogeneous curve of degree n'. We assume that an ordering of i, j, k is fixed throughout this paper and C is presented according to that ordering. By complete we mean that the curve has all the possible monomials of degree n'. We need to check if P_i for $i = 1, 2, \ldots, s$ and Q_j for $j = 1, 2, \ldots, t$ satisfy the curve C. Note that, there is no need to compute the values of a_{ijk}, just mere existence will solve the discrete logarithm problem.

Let P be a point on $\mathcal{E}$. We denote by $\overline{P}$ the value of C when the values of x, y, z in P is substituted in C. In other words, $\overline{P}$ is a linear combination of a_{ijk} with the fixed ordering. Similarly for Qs. We now form a matrix $\mathcal{M}$ where the rows of $\mathcal{M}$ are $\overline{P_i}$ for $i = 1, 2, \ldots, s$ and $\overline{Q_j}$ for $j = 1, 2, \ldots, t$. If this matrix has a non-zero left-kernel, we have solved the discrete logarithm problem. By *left-kernel* we mean the kernel of $\mathcal{M}^{\mathrm{T}}$, the transpose of $\mathcal{M}$.

2.2 Why look at the left-kernel instead of the kernel

In this paper, we will use the left-kernel more often than the (right) kernel of $\mathcal{M}$. We denote the left-kernel by $\mathcal{K}$ and kernel by $\mathcal{K}'$. We first prove the following theorem:

Theorem 2. *The following are equivalent:*

(a) $\mathcal{K} = 0$.
(b) $\mathcal{K}'$ *only contain curves that are a multiple of* $\mathcal{E}$.

Proof. The proof uses a simple counting argument. First recall the well-known fact that the number of monomials of degree d is $\binom{d+2}{2}$. Furthermore, notice two things – all multiples of $\mathcal{E}$ belongs to $\mathcal{K}'$ and the dimension of that vector-space (multiples of $\mathcal{E}$) is $\binom{n'-1}{2} = \dfrac{(n'-2)(n'-1)}{2}$, where n' is as defined earlier.

Now, $\mathcal{M}$ is as defined earlier, has $3n'$ rows and $\dfrac{(n'+1)(n'+2)}{2}$ columns. Then $\mathcal{K} = 0$ means that the row-rank of $\mathcal{M}$ is $3n'$. So the dimension of the $\mathcal{K}'$ is

$$\frac{(n'+1)(n'+2)}{2} - 3n' = \frac{(n'-2)(n'-1)}{2}.$$

This proves (a) implies (b).

Conversely, if $\mathcal{K}'$ contains all the curves that are a multiple of $\mathcal{E}$ then its dimension is at least $\dfrac{(n'-2)(n'-1)}{2}$, then the rank is $3n'$, making $\mathcal{K} = 0$. □

It is easy to see, while working with the above theorem $\mathcal{M}$ cannot repeat any row. So from now onward we would assume that $\mathcal{M}$ has no repeating rows. For all practical purposes this means that we are working with distinct partitions (also known as unique partitions). By distinct partition we mean those partitions which has no repeating parts.

A question that becomes significantly important later is, instead of choosing k points from the elliptic curve what happens if we choose $k+l$ points for some positive integer l. The answer to the question lies in the following theorem.

Theorem 3. *If $l \geq 1$, the dimension of the left kernel of $\mathcal{M}$ is l.*

Proof. First assume $l \geq 1$. In this case, any non-trivial element of $\mathcal{K}'$ will define a curve which passes through more than $3n'$ points of the elliptic curve. Since the elliptic curve is irreducible, it must be a component of the curve. Thus the equation defining the curve must be divisible by the equation defining the elliptic curve. Thus, the dimension of $\mathcal{K}'$ is the dimension of all degree n' homogeneous polynomials which are divisible by the elliptic curve. This is the same as the dimension of all degree $n'-3$ homogeneous polynomials. Thus, we get

$$\dim(\mathcal{K}') = \frac{(n'-2)(n'-1)}{2}.$$

On the other hand, by rank-nullity theorem, it follows:

$$\begin{aligned} \dim(\mathcal{K}') + \dim(\text{image}(\mathcal{M})) &= \tfrac{(n'+2)(n'+1)}{2} \\ \dim(\mathcal{K}) + \dim(\text{image}(\mathcal{M}^{\mathrm{T}})) &= 3n' + l. \end{aligned}$$

Thus, since row rank and the column rank of a matrix are equal,

$$\dim(\mathcal{K}) = 3n' + l - \frac{(n'+2)(n'+1)}{2} + \dim(\mathcal{K}') = l.$$

Corollary 1. *Assume that $\mathcal{M}$ has $3n'+l$ rows, computed from the same number of points of the elliptic curve $\mathcal{E}$. If there is a curve $\mathcal{C}$ intersecting $\mathcal{E}$ non-trivially in $3n'$ points among $3n'+l$ points, then there is a vector v in $\mathcal{K}$ with at least l zeros. Conversely, if there is a vector v in $\mathcal{K}$ with at least l zeros, then there is a curve $\mathcal{C}$ passing through those $3n'$ points that correspond to the non-zero entries of v in $\mathcal{M}$.*

Proof. Assume that there is a non-trivial curve $\mathcal{C}$ intersecting $\mathcal{E}$ in $3n'$ points. Then construct the matrix $\mathcal{M}'$ whose rows are the points of intersection. Then from the earlier theorem we see that $\mathcal{K}$ for this matrix $\mathcal{M}'$ is non-zero. In all the vectors of $\mathcal{K}$ if we put zeros in the place where where we deleted rows then those are element of the left kernel of $\mathcal{M}$. It is clear that these vectors will have at least l zeros.

Conversely, if there is a vector with at least l zeros in $\mathcal{K}$, then by deleting l zeros from the vector and corresponding rows from $\mathcal{M}$ we have the required result from the theorem above. □

The algorithm that we present in this paper has two parts. One reduces it to a problem in linear algebra and the other solves that linear algebra problem which we call Problem L. The first algorithm, Algorithm 1, is Las Vegas in nature with high success probability and is polynomial in both time and space complexity.

3 The main algorithm – reducing ECDLP to a linear algebra problem (Problem L)

Algorithm 1: Reducing ECDLP to a linear algebra problem (Problem L)

Data: Two points P and Q, such that $mP = Q$

Result: m

Select a positive integers, n' and $l = 3n'$. Initialize a matrix with $3n' + l$ rows and $\binom{n'+2}{2}$ columns. Initialize a vector $\mathcal{I}$ of length $3n' - 1$ and another vector $\mathcal{J}$ of length $l + 1$. Initialize integers $A, B = 0$.

```
repeat
    for i = 1 to 3n' − 1 do
        repeat
            choose a random integer r in the range [1, p)
        until r is not in I
        I[i] ← r
        compute rP
        compute \overline{rP}
        insert \overline{rP} as the i^th row of the matrix M
    end
    for i = 1 to l + 1 do
        repeat
            choose a random integer r in the range [1, p)
        until r is not in J
        J[i] ← r
        compute −rQ
        compute \overline{−rQ}
        insert \overline{−rQ} as the (3n' + i − 1)^th row of the matrix M
    end
    compute K as the left-kernel of M
until K has a vector v with l zeros (Problem L)
for i = 1 to 3n' − 1 do
    if v[i] ≠ 0 then
        A = A + I[i]
    end
end
for i = 3n' to 3n' + l do
    if v[i] ≠ 0 then
        B = B + J[i − 3n' + 1]
    end
end
return A × B^{−1} mod p
```

Why is this algorithm better than exhaustive search In the exhaustive search we would have picked a random set of $3n'$ points and then checked to see if the sum of those points is Q. In the above algorithm we are taking a set of $3n'+l$ points and then checking all possible $3n'$ subsets of this set simultaneously. There are $\binom{3n'+l}{l}$ such subsets. This is one of the main advantage of our algorithm.

Probability of success of the above algorithm To compute the probability, we need to understand the number of unique partitions of an integer m modulo a prime p. For our definition of partition, order of the parts does not matter. The number of partitions is proved in the following theorem:

Theorem 4. *Let $2 < k \leq p/2$ be an integer. The number of k unique partitions of m modulo a odd prime p is at least*

$$\frac{(p-1)(p-2)\dots(p-k+2)(p-2k+1)}{k!}.$$

Proof. The argument is a straight forward counting argument. We think of k parts as k boxes. Then the first box can be filled with $p-1$ choices, second with $p-2$ choices and so on. The last but one, $k-1$ box can be filled with $p-k+1$ choices. When all $k-1$ boxes are filled then there is only one choice for the last box, it is m minus the sum of the other boxes. So it seems that the count is $(p-1)(p-2)\dots(p-k+1)$ choices.

However there is a problem, the choice in the last box might not be different from the first $k-1$ choices. To remove that possibility we remove $k-1$ choices from the last but one box. Furthermore, the choice in the last box can not be zero that removes one more choice from the last but one box.

In some pathological cases, the above argument might remove more choices than necessary. Thus the above formula gives a lower bound for the number of k distinct partitions.

Since order does not matter, we divide by $k!$. □

Consider the event, m is fixed, we pick k integers less than $p/2$. What is the probability that those numbers form a partition of m. From the above theorem, number of favorable events is

$$\frac{(p-1)(p-2)\dots(p-k+2)(p-2k+1)}{k!}$$

and the total number of events is $\binom{p}{k}$. Since for all practical purposes k is much smaller than p, we approximate the probability to be $\frac{1}{p}$.

Now we look at the probability of success of our algorithm. In our algorithm we choose $3n'$ points from $3n'+l$ points. This can be done in $\binom{3n'+l}{l}$ ways. Then the probability of success of the algorithm is $1-\left(1-\frac{1}{p}\right)^{\binom{3n'+l}{l}}$.

Let us first look at the $\left(1-\frac{1}{p}\right)^{p}$. It is well known that $\left(1-\frac{1}{p}\right)^{p}$ tends to $\frac{1}{e}$ when p tends to infinity. So if we can make $\binom{3n'+l}{l}$ close to p, then we can claim the asymptotic probability of our algorithm is $1-\frac{1}{e}$ which is greater than $\frac{1}{2}$.

Since we are dealing with matrices, it is probably the best that we try to keep the size of it as small as possible. Note that the binomial coefficient is the biggest when it is of the form $\binom{2n}{n}$ for some positive integer n. Furthermore, from Stirling's approximation it follows that for large enough n, $\binom{2n}{n} \approx \frac{4^n}{\sqrt{\pi n}}$.

So, when we take $3n' = l$ and such that $\binom{3n'+l}{l} = p$ then l is the solution to the equation $l = O(1) + \log l + \log p$.

To understand the time complexity of this algorithm (without the linear algebra problem), the major work done is finding the kernel of a matrix. Using Gaussian elimination, there is an algorithm to compute the kernel which is cubic in time complexity. Thus we have proved the following theorem:

Theorem 5. *When p tends to infinity, the probability of success of the above algorithm is approximately $1 - \frac{1}{e} \approx 0.6321$. The size of the matrix required to reach this probability is $O(\log p)$. This makes our algorithm polynomial in both time and space complexity.*

3.1 Few comments

Accidentally solving the discrete logarithm problem It might happen, that while computing rP and rQ in our algorithm, it turns out that for some r_1 and r_2, $r_1 P = r_2 Q$. In that case, we have solved the discrete logarithm problem. We should check for such accidents. However, in a real life situation, the possibility of an accident is virtually zero, so we ignored that in our algorithm completely.

On the number of Ps and Qs in our algorithm The algorithm will take as input P and Q and produce different Ps and Qs and the produce a vector v with l many zeros. If all of these l zeros fall either in the place of Ps or Qs exclusively, then we have not solved the discrete logarithm problem. To avoid this, we have chosen Ps and Qs of roughly same size, with one more P than Q. This way the vector v will have atleast one non-zero in the place of both P and Q.

Allowing, detecting and using multiple intersection points in our algorithm One obvious idea to make our algorithm slightly faster: allow multiplicities of intersection between the curve C and the elliptic curve $\mathcal{E}$. This will increase the computational complexity. Since the elliptic curve is smooth at the points one is interested in, one observes that with high probability the multiplicity of intersection will coincide with the multiplicity of the point in C. This reduces to checking if various partial derivatives are zero. This can easily be done by introducing extra rows in the matrix $\mathcal{M}$. Then the algorithm reduces to finding vectors with zeroes in a particular pattern. This is same as asking for special type of solutions in Problem L. However, this has to be implemented efficiently as probability of such an event occurring is around $1/p$ for large primes p.

3.2 Choosing Ps and Qs uniformly random

We raise an obvious question, can one choose the set of n_i and n'_j (see Section 2.1) which give rise to P_i and Q_j respectively in such a way that the probability of solving the discrete logarithm problem is higher than the uniformly random selection? In this paper we choose P_i and Q_j uniformly random.

4 Dealing with the linear algebra problem

This paper provides an efficient algorithm to reduce the elliptic curve discrete logarithm problem to a problem in linear algebra. We call it the Problem L.

At this stage we draw the attention of the reader to some similarities that emerge between the most powerful attack on the discrete logarithm problem over finite fields, the index-calculus algorithm, and our algorithm. In an index-calculus algorithm, the discrete logarithm problem is reduced to a linear algebra problem. Similar is the case with our algorithm. However, in our case, the linear algebra problem is of a different genre and not much is known about this problem. In this paper, we have not been able to solve the linear algebra problem completely. However, we made some progress and we report on that in this section.

Problem L. Let W be a l-dimensional subspace of a n-dimensional vector space V. The vectors in the vectors space are presented as linear sum of some fixed basis of V. The problem is to determine, if W contains a vector with l zeros. If there is one such vector, find that vector.

This problem is connected with the earlier algorithm in a very straightforward way. We need to determine if the left-kernel of the matrix $\mathcal{M}$ contains a vector with l zeros and that is where Problem L must be solved efficiently for the overall algorithm to run efficiently. As is customary, we would assume that the kernel $\mathcal{K}$ is presented as a matrix of size $l \times (3n' + l)$, where each row is an element of the basis of $\mathcal{K}$. Recall that we chose $3n' = l$.

A algorithm that we developed, uses Gaussian elimination algorithm multiple times to solve Problem L. In particular we use the row operations from the Gaussian elimination algorithm. Abusing our notations slightly, we denote the basis matrix of $\mathcal{K}$ by $\mathcal{K}$ as well. Now we can think of $\mathcal{K}$ to be made up of two blocks of $l \times l$ matrix. Our idea is to do Gaussian elimination to reduce each of these blocks to a diagonal matrix one after the other. The reason that we do that is, when the first block has been reduced to diagonal, every row of the matrix has at least $l - 1$ zeros. So we are looking for another zero in some row. The row reduction that produced the diagonal matrix in the first block might also have produced that extra zero and we are done. However, if this is not the case, we go on to diagonalize the second block and check for that extra zero like we did for the first block.

Algorithm 2: Multiple Gaussian elimination algorithm

Data: The basis matrix $\mathcal{K}$

Result: Determine if Problem L is solved. If yes, output the vector that solves Problem L

for *i=1 to 2* **do**

  row reduce block i to a lower triangular block

  check all rows of the new matrix to check if any one has l zeros

  if *there is a row with l zeros* **then**

    STOP and return the row

  end

  row reduce the lower-triangular block to a diagonal block

  check all rows of the new matrix to check if any one has l zeros

  if *there is a row with l zeros* **then**

    STOP and return the row

  end

end

STOP (Problem L not solved)

5 Complexity, implementation and conclusion

5.1 Complexity

We describe the complexity of the whole algorithm in this section. First note that the whole algorithm is the composition of two algorithms, one is Algorithm 1, which has success probability 0.6 and the other is the linear algebra problem. It is easy to see from conditional probability that the probability of success of the whole algorithm is the product of the probability of success of Algorithm 1 and Algorithm 2.

Let us now calculate the probability of Algorithm 2 under the condition that Algorithm 1 is successful. In other words, we know that Algorithm 1 has found a $\mathcal{K}$ whose span contains a vector with l zeros. What is the probability that Algorithm 2 will find it?

Notice that Algorithm 2 can only find zero if they are in certain positions and the number of such positions is l^2. Total number of ways that there can be l zeros in a vector of size $3n' + l$ is $\binom{3n'+l}{l}$. In our setting we have already assumed that $\binom{3n'+l}{l} \approx p$. Then the probability of success of the whole algorithm is

$$0.6 \times \frac{(\log p)^2}{p}.$$

Which is a significant improvement over exhaustive search!

One thing to notice, the probability of success is $1 - \left(1 - \frac{1}{p}\right)^{\binom{3n'+l}{l}}$ and in the probability estimate we have $\binom{3n'+l}{l}$ in the denominator. Furthermore, one observes that in this paper we have taken $\binom{3n'+l}{l}$ to approximately equal the prime p. One can now question our choice and argue, if we took $\binom{3n'+l}{l}$ to be much smaller than p, we might get a better algorithm. Alas, this is not the case, $1-\left(1-\frac{1}{p}\right)^{p^{\frac{1}{n}}}$ tends to 0 as p tends to infinity for $n \geq 2$.

5.2 Implementation

The aim of our implementation is to determine an average number of tries required by the Las Vegas algorithm to solve a elliptic curve discrete logarithm problem. However to generate real life data that solves discrete logarithm problems is very time consuming. So we set a cut off, if the number of steps taken by the program is more than $\sqrt{p}$ where p is the order of P, we stop the program. This way we deal with the black swan situation that normally happens in any Las Vegas algorithm. In each step we generate points on the elliptic curve, form the matrix $\mathcal{M}$ and then the left kernel $\mathcal{K}$, perform two row-operations on $\mathcal{K}$ and see if there are l zeros in a row after each row-reduction of $\mathcal{K}$. The algorithm was implemented using NTL [12].

The left-kernel $\mathcal{K}$ of M is a matrix of size $l \times 2l$. Thus $\mathcal{K}$ consists of two matrices of size $l \times l$ stacked sideways. Both these matrices in $\mathcal{K}$ are row-reduced to a diagonal matrix using only row operations in our experiment. The first attempt reduces columns 1 to l in $\mathcal{K}$ and checks for a row with l zeros. If a row with l zeros is present, the discrete logarithm is solved. If the first attempt fails to reduce $\mathcal{K}$ which contain a row with l zeros, second row reduction is applied. The second attempt reduces columns $l+1$ to $2l$ and checks for a row with l zeros. If this reduced form contains a row with l zeros DLP is solved. If the first as well as the second row reduction does not yield a row with l zeros the algorithm is restarted with a fresh choice of random points on the elliptic curve.

The Las Vegas algorithm was executed 80 times and try-count, the number of tries, for each execution was recorded. The data is presented in Table 1. Each of the 80 executions resulted in either a value for try-count if the DLP was solved before the try-count reached $\sqrt{p}$ or no value for try-count if DLP was not solved in less than $\sqrt{p}$ steps.

Table 1. No. of steps required to solve ECDLP using our algorithm

Field Size	$\sqrt{p}$	Solved in less than $\sqrt{p}$ tries	Can't solve in less than $\sqrt{p}$ tries
2^{17}	256	80	00
2^{19}	512	80	00
2^{23}	2048	77	03
2^{29}	16384	39	41
2^{31}	32768	26	54
2^{37}	262144	07	73

5.3 Conclusion

We conclude this paper by saying that we have found a new genre of attack against the elliptic curve discrete logarithm problem. This attack has some similarities with the well-known index-calculus algorithm. In an index-calculus algorithm, the discrete logarithm problem is reduced to a problem in linear algebra and then the linear algebra problem is solved. However, the similarities are only skin deep as our linear algebra problem is completely new.

5.4 Acknowledgment

We are indebted to the anonymous referees for their careful reading of the manuscript and detail comments. Due to lack of time, we were not able to incorporate the new research directions suggested. However, those comments have certainly piqued our interest and we thank the referees for those.

References

[1] Alessandro Amadori, Federico Pintore, and Massimiliano Sala. On the discrete logarithm problem for prime-field elliptic curve. *Finite Fields and Their Applications*, 51:168–182, 2018.

[2] D. Bernstein and T. Lange. Non-uniform cracks in the concerete: the power of free precomputation. In *Advances in Cryptology – ASIACRYPT 2013*, volume 8270 of *LNCS*, pages 321–340, 2013.

[3] William Fulton. *Algebraic Curves.* self-published, 2008.

[4] S. Galbraith and P. Gaudry. Recent progress on the elliptic curve discrete logarithm problem. *Designs, Codes and Cryptography*, 78:51–78, 2016.

[5] S. Galbraith and S. Gebregiyorgis. Summation polynomial algorithms for elliptic curves in characteristic two. In *Progress in Cryptology – INDOCRYPT 2014*, volume 8885 of *LNCS*, pages 409–427, 2014.

[6] Pierrick Gaudry. Index calculus for abeian varieties of small dimension and the elliptic curve discrete logarithm problem. *Journal of Symbolic computation*, 44:1690–1702, 2009.

[7] Joe Harris. *Algebraic Geometry.* Springer, 1992.

[8] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. *An introduction to mathematical cryptography.* Springer, 2008.

[9] Michael J. Jacobson, Neal Koblitz, Joseph H. Silverman, Andreas Stein, and Edlyn Teske. Analysis of the xedni calculus attack. *Designs, Codes and Cryptography*, 20(1), 41-64 2000.

[10] J. S. Milne. *Elliptic Curves.* BookSurge Publishers, 2006.

[11] Igor Semaev. Summation polynomials and the discrete logarithm problem on elliptic curves. https://eprint.iacr.org/2004/031, 2004.

[12] Victor Shoup. NTL: A library for doing number theory. http://www.shoup.net/ntl, 2016.

[13] Joseph H. Silverman. The xedni calculus and the elliptic curve discrete logarithm problem. *Designs, Codes and Cryptography*, 20(1):5–20, 2000.

[14] M. Wiener and R. Zuccherato. Faster attacks on elliptic curve cryptosystems. In *Selected Areas in Cryptography – SAC '98*, volume 1556 of *LNCS*, pages 190–200, 1999.