IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 2, Ver.VI (Mar-Apr. 2016), PP 01-11
www.iosrjournals.org

A Review on Solving ECDLP over Large Finite Field Using Parallel Pollard's Rho (ρ) Method

Kaushal A. Chavan[1], Dr. Indivar Gupta[2], Dr. Dinesh B. Kulkarni[3]
[1-3]Department of Information Technology, Walchand College of Engineering, Sangli, MS, India
[2]SAG, DRDO, New Delhi, India

Abstract: *Elliptic Curve Discrete Log Problem (ECDLP) is the underlying basis of many popular Public Key Scheme like Diffie-Hellman and ElGamal. The strength of such public key schemes is based on the difficulty of solving the ECDLP. The best method for solving the ECDLP has time complexity exponential in the size of the underlying field. ECDLP based cryptosystems are popular because they provide good security at key sizes much smaller than number theoretical Public Key Schemes like RSA cryptosystem. Elliptic curve cryptosystem based on ECDLP are also present in the list of recommended algorithms for use by NIST and NSA. Since ECDLP based cryptosystems are in wide-spread use, continuous efforts on monitoring the effectiveness of new attacks or improvements to existing attacks on ECDLP over large field is important. Using the parallel Pollard's method to solve the ECDLP efficiently is one of the prime concerns. Use of different parallel architectures like cluster computing (MPI), GPGPU, FPGA cluster increases the effectiveness of attack.*
This article covers various aspects of finite field, Elliptic Curve Cryptography (ECC), ECDLP, methods for solving ECDLP along with emphasis on parallel Pollard's methods using CPU cluster.
Keyword: *ECDLP, Parallel Pollard's Rho method, Cluster computing, Elliptic Curves*

I. Introduction

Elliptic curves play a very important role in the field of cryptography. To achieve desired security level, key size requirement in the traditional public key cryptographic systems such as RSA and ElGamal, which are based on difficulty of solving Integer Factorization Problem or Discrete Logarithm Problem (DLP) in Natural Group, are required to be significantly high. However, in the case of elliptic curve based cryptosystems, which are based on the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP), we can work with a smaller key size in comparison to RSA or ElGamal to achieve same security level. Due to smaller key size, implementation of elliptic curve based cryptosystems requires small silicon area. Therefore elliptic curve based cryptosystems have become popular in small devices such as Smart cards, PDA providing good amount of security. To cryptanalyze elliptic curve based systems, one needs to solve ECDLP in reasonable amount of time. In mathematics, an elliptic curve *(E)* is a smooth projective algebraic curve of genus one. An elliptic curve is in fact an abelian variety – that is, it has multiplication, defined algebraically, with respect to which it is a (necessarily commutative) group. The elliptic curve cryptography is based on the operations performed on the point present on the curve.

The point operation obeys the chord and tangent law (Fig. 1) [1]. The elliptic curves are built on a finite field. A finite field is a family of elements but necessarily finite in number. The efficient implementation of finite field arithmetic is an important prerequisite in implementing elliptic curve systems because curve operations are performed using arithmetic operations in the underlying field.

Solving any elliptic curve cryptosystem directly relates to the efficiency of solving the ECDLP problem based on that curve. The strength of elliptic curve cryptosystem is directly proportional to the order (size) of the underlying finite field. As the field order increases the hardness of the cryptosystem increases. Traditions methods (naïve methods) are not capable to solve the ECDLP in reasonable amount of time. Pollard's rho algorithm, proposed by J. M. Pollard [3], gives a good tradeoff between time and space to solve the ECDLP problem. Solving the ECDLP problems whose underlying field order is large, is very difficult using sequential version of Pollard's Rho algorithm in reasonable amount of time. Growth in the parallel computing technology enables us to make the use of parallelized versions of algorithm.

The parallel Pollard's rho algorithm can be used to solve the ECDLP problems over large fields, which gives almost a linear speed-up. This algorithm makes the use of the parallel technologies like GPGPU, Cluster computing, etc. to solve the ECDLP.

The paper is organized as follows. The second section covers the literature review of the work done in solving the ECDLP problem. Third section covers the background of elliptic curve cryptography which includes brief overview of finite fields, ECDLP problem, and elliptic curves based on the finite fields. Fourth section gives abstract information of the work done in solving the ECDLP problem; it includes different algorithms which are used to solve the ECDLP. Fifth section takes overview of the sequential Pollard's Rho algorithm describing the algorithm. Sixth section covers the details of Parallel Pollard's rho algorithm and discusses the factors in implementing on a CPU cluster. The last section covers different parallel technologies which has been used prior and the results of our implementation.

II. Literature Review

To check the strength of cryptographic systems a lot of attacks have been attempted on the systems. Large amount of work has been carried out in solving the integer factorization problem (IFP), discrete logarithm problem in multiplicative group of finite field (DLP) and in the group points on an elliptic curve (ECDLP). The concept of elliptic curve cryptography was put forward by V. Miller in 1986 in the paper [12]. After that N. Koblitz, A. Menezes and S. Vanstone elaborated the concept in 2000 in the paper [11]. In these papers they proposed a system which uses the elliptic group $E(F_q)$, defined over a finite field F_q as an arithmetic base of cryptosystem.

The strongest known attacks against the ECDLP are generic attacks based on Pollard's Rho method. J. M. Pollard first proposed the Pollard's Rho method in the paper [3], in 1978. Certain improvements were suggested and showed by researchers in various papers. In 1998, M. Wiener and J. Zuccherato [7], in 1999 van Oorschot and Wiener [4], and in 2000 Galleant, Lambert and Vanstone [8] showed how the computation can be efficiently parallelized. In these papers, variants in implementation of the Pollard's Rho method were showed which were able to exploit the use of parallel architectures.

CERTICOM [10] introduced ECC challenges in 1997 to increase industry acceptance of cryptosystems which are based on elliptic curve discrete logarithm problem (ECDLP). It publishes system parameters and challenges for different security levels. Since then, solving the ECDLP problem based on large field is of great interest to the researchers in the world. Up till now hardest CERTICOM challenge of prime field based ECDLP is solved by Monico et al. [10] which was ECC p-109 by using a cluster of 10,000 machines in 549 days. Another hard ECDLP was solved by Bos et al. [26] which was over 112-bit prime field using 200 PlayStation 3 in 6 months.
Parallel pollard's rho algorithm [4] has been used by almost allchallengers to solve these problems.

III. Background

3.1 Finite Field:

Fields are abstraction of number systems (such as the rational number *Q*, the real numbers *R*, and the complex numbers *C*) and their essential properties. They consists of set *F* together with two operations, addition (+) and multiplication (.) that usually satisfy the group properties and are abelian in nature. A finite field is a family of finite number of elements which obey certain rules. Three kinds of finite field are important for efficient implementation of elliptic curve cryptography - the prime field, binary field and optimal extension field. The order of a finite field is total number of elements present in the field. A finite field *F* of order *q*exist if and only if *q* is a prime power, i.e., $q=p^m$ where *p* is a prime number which is called the characteristic of *F* and *m* is a positive integer. If $m=1$, then *F* is called a prime field. If $m>=2$, then field (*F*) is called an extension field.
Let *p* be a prime number. The integer modulo *p*, consisting of the integers $\{0, 1, 2,\ldots, p-1\}$ with addition operation and multiplication operation performed modulo *p*, is a finite field of order *p*. Efficient implementation of finite field arithmetic is an important affecting factor in elliptic curve cryptosystems [1].

3.2 Elliptic curves over finite field:

Elliptic curve (*E*) defined over a finite field (F_q) satisfies the following equation [1]-

$$E : y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a_6^3 \quad \ldots.(1)$$

where, the coefficients of the equation are taken from the underlying finite field. The general curve equation when the underlying finite field has characteristic not equal to 2 or 3 is given by-

$$y^2=x^3+ax+b \qquad \ldots\ldots (2)$$

where $a, b \in F_q$ and $4a^3+27b^2 \neq 0$, together with the special point *O* at the infinity serving as the identity point for the group.
Let *E* be an elliptic curve defined over F_q. Let $E(F_q)$be the group of points over the elliptic curve. The number of points in $E(F_q)$, denoted as $\#E(F_q)$, is called the order of *E* over F_q. Hasse's theorem is used to give the order of group.
Let *P* and *Q* be two points on the curve.
For all $P, Q \in E(K)$following group laws are defined.

- $P + O = O + P = P$
- $-O = O$
- If $P= (x_1,y_1) \neq O$ then $-P = (-x_1, -y_1)$
- If $Q = -P$ then $P + Q = O$
- If *P* and *Q* belong to *E* then $P + Q$ also does.

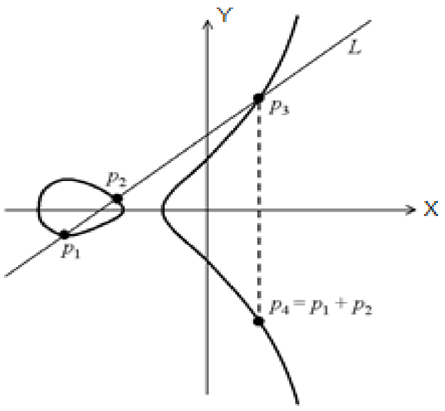


Fig 1: Point addition (Chord- Tangent rule)

Let $P = (x_1, y_1)$ and $Q = (x_2,y_2)$ be two points on curve, then addition of P and Q can be described with the help of chord – tangent law (Fig. 1) as-

$$P + Q = (x_3, y_3)$$

where,

$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = \lambda(x_1 - x_3) - y_1$$

and

$$\lambda = (y_2 - y_1) / (x_2-x_1) \qquad \text{if } P \neq Q$$
$$\lambda = (3x_1^2 + a) / 2y_1 \qquad \text{if } P = Q$$

where, λ is slope of the line passing through point *P* and *Q* and slope of tangent passing through *P* or *Q*.

3.3 Elliptic Curve Cryptography

Let *E* be an elliptic curve over a finite field F_q. Let *P* be a point in $E(F_q)$, and suppose that *P* has order *n*. Then the cyclic subgroup of $E(F_q)$ generated by *P* is-

$$<P> = \{P, 2P, 3P, ..., (n-1)P\}.$$

The prime *q*, the equation of the elliptic curve *E*, and the point *P* and the point order *n*, are the public domain parameters. A private key is an integer *l* that is selected uniformly at random from the interval *[1, n-1]*, and the corresponding public key is *Q=lP*.

The elliptic curve cryptosystem based on ElGamal scheme is presented below: A plaintext *M* is first represented as a point *M*, and then encrypted by adding (using Chord and Tangent rule as shown in fig. 1) it to *k Q* where *k* is a randomly selected integer, and *Q* is the intended receiver's public key. The sender transmits the points $C_1 = kP$ and $C_2 = M + kQ$ to the recipient who uses her private key *l* to compute

$$l\, C_1 = l\,(k\,P) = k\,(l\,P) = k\,Q ....(3)$$

and thereafter recovers $M= C_2 - kQ$.

3.4 ECDLP:

The discrete logarithm problem in a group *G* is as follows- given $y \in G$, find an integer *x* such that,

$$g^{x} = y \;...... (4)$$

An integer *x* exists if and only if $y \in <g>$, where $<g>$ is the cyclic subgroupof *g* generated by *g*. Consider an elliptic curve $E(F_q)$ defined over a finite field F_q of order *p*. Consider two points on that elliptic curve *P* and *Q*, such that $P, Q\in E(F_q)$. Point*P* is the generator point and forms a cyclic group $<P>$. Point*Q* is the point in the group formed by *P* such that $Q \in <P>$. The elliptic curve discrete logarithm problem (ECDLP) problem [1] is to find an integer *l* such a that,

$$Q = lP = (P + P + ... + l \text{ times}) .... (5)$$

The addition operation and the point doubling operations are defined over the elliptic curve of finite field F_q.

Many attacks had been proposed for solving the ECDLP problem. The most basic will be the naïve approach also known as exhaustive search. Here one computes the sequence of points *P, 2P, 3P, 4P, ...*, until it encounters Q. The running time is approximately *n* steps in the worst case and *n/2* steps in average case. Therefore, exhaustive search can be circumvented by selecting elliptic curve parameters sufficiently large.

IV. Methods To Solve ECDLP

Many attacks have been proposed for solving Discrete Log Problem (DLP). In this section we discuss some of the attacks that work on arbitrary groups including elliptic curve. These attacks can be applied for solving ECDLP.

- Pohlig-Hellman Reduction
- Baby-Step Giant-Step
- Pollard's Rho Attack
- Pollard's Lambda Attack
- Index Calculus Algorithm
- Xedni Calculus Algorithm
- Weil Descent and Cover Attack
- Some attacks for ECDLP on particular curve

4.1 Pohlig- Hellman Reduction:

This method requires the knowledge of the factorization of-

$$n=|E(F_q)|=\pi_{i=1}^{k} p_i^{ei}.$$

This method reduces DLP from higher order group to small subgroup of prime order (prime factor of n). To solve DLP over small subgroup, some other method is required. Reduction carried out in two steps: reduction from n to p_i^{ei} and reduction from p_i^{ei} to p_i. After the reduction, we solve DLP over subgroup of prime order (p_i) and lift on subgroup of prime power (p_i). We then solve DLP for each subgroup of order p_i^{ei} and apply CRT to solve DLP on actual group of order n.

4.2 Baby-Step Giant-Step:

This method is proposed by Shanks D [15]. Aim is to find discrete log of Q with respect to P. Let $Q = xP$ where x be discrete log of Q w.r.t. P. Assume that $m = |\sqrt{n}|$. Applying Euclidean Division, it is known that there exists $q, r \in Z$: $x = qm + r$, with $0 \leq r < m$. The principle idea is find q and r (hence x) using $(Q - rP) = (qm)P$. Thus we have:

1. *Baby Step:*

Compute a table of Baby steps $B = \{(Q-rP, r)\}_{0 \leq r < m}$. If there exists $r \in [0, m]$ such that $(O, r) \in B$, then $x = r$.

2. *Giant Step:*

Let $R = m P$. For $q = 1, 2 \ldots$, compute $S_q = q R$ and check the table B. If there exists $r \in [0, m]$ such that $(q R, r) \in B$, then $x = qm + r$.

This method terminates in $|\sqrt{n}|$ steps. The complexity of this method is roughly $O(\sqrt{n})$ as this much time is required to compute the baby steps and maximum amount to compute the giant steps. The main problem with themethod is that it requires the storage$|\sqrt{n}|$ group elements.

4.3 Pollard's Rho Algorithm:

The general idea of Pollard ρ Algorithm is as follows. Let S be a set $|S| = n$ and $f: S \rightarrow S$ be a random mapping. Starting with random value $x_0 \in S$, we compute
$x_{i+1} = f(x_i)$for $i \geq 0$.

It is clear that the sequence $x_0, x_1, x_2, \ldots,$ is deterministic random walk. Since S is finite we must eventually obtain $x_i = x_j$ which implies $x_{i+1} = f(x_i) = f(x_j) = x_{j+1}$. Hence sequence $x_0, x_1, x_2, \ldots$ becomes cyclic and a picture of sequence looks like the Greek letter ρ. The rho-shape has an initial tail and cyclic part. The expected length of both, the tail and the cyclic part is $\sqrt{\pi n/8}$. If we use Pollard's Rho method in a naive way then it also requires $O(\sqrt{n})$ memory to store the elements as in Baby-Step-Giant-Step algorithm. Therefore, to overcome from storage problem, Floyds cycle finding algorithm has been suggested. This algorithm detect a collision with $O(1)$ storage. Floyds Cycle Finding Algorithm is described below:

1. Given (x_1, x_2), compute (x_2, x_4), then (x_3, x_6) and so on.
2. Given the pair (x_i, x_{2i}) we compute $(x_{i+1}, x_{2i+2}) = (f(x_i), f(f(x_{2i})))$.
3. We stop when we find $x_m = x_{2m}$.
4. If tail has length λ and cycle length μ then

$$m = \mu . \frac{\lambda}{\mu}$$

5. Since $\lambda \leq m \leq \mu + \lambda$, then we see that $m = O(\sqrt{n})$.

Detailed Pollard's Rho algorithm for ECDLP has been discussed in section 5.

4.4 Pollard's Lambda Algorithm:

Pollard's λ method works similar to Pollard's ρ method. However this method uses two starting points. Therefore we have two random walks. Eventually they will have points in common, and therefore they will coincide thereafter. The picture of this process appears like Greek letter λ, hence the name. In this method we assume that the discrete logarithm lies in a certain interval $x \in [a \;....\; b]$. We let $w = b - a$ denote length of interval and set $N = \sqrt{w}$. The algorithm is described below:

1. Divide G up into $k = log_2(w)/2$ subsets S_i for $0 \leq i < k$.
2. Let $s_i = 2^i$ for $0 \leq i < k$ and define random walk $X_{i+1} = X_i + s_jP$ if $X_i \in S_j (0 \leq i < k)$.
3. For first random walk we let $R_0 = bP$ and $R_{i+1} = R_i + s_jP$ for $i = 1,...N-1$.
4. Let $c_0 = b$ and $c_{i+1} = c_i + s_j (mod\ n)$, then $log_P(R_N) = c_N$.
5. For second random walk, let $H_0 = Q = xP$ and compute $H_{i+1} = H_i + s_jP$.
6. Set $x_0 = 0$ and $x_{i+1} = x_i + s_j\ (mod\ n)$, then $log_P(H_i) = x + x_i$.
7. Once path of H_i meets that of R_i it follows it.
8. Iterate M times until $H_M = R_N$, then $c_N = log_P(R_N) = log_P(H_M) = x + x_M (mod\ n)$

Expected running time of this algorithm is $O(\sqrt{w})$ and require $O(log(w))$ memory to store elements.

4.5 Index Calculus Algorithm for ECDLP:

Index calculus method works effectively for Discrete Logarithm Problems (DLP) of multiplication group F^*_p. This method makes use of the fact that $F^*_p$is the reduction modulo p of the group Q^*. We usually use a small numbers of prime elements in Q^* called factor base. It is noted that there are lot of these to choose from as Q^* has infinitely many generators and they have an obvious ordering, with smaller generators being easier to handle in the computer.

Now the question is that whether this method can be extended to solve ECDLP efficiently. To apply the index calculus for the DLP to the ECDLP, one would first lift the elliptic curve E/F_p to an elliptic curve E/Q, next we lift various points from E/F_p to E/Q. Finally we use relationship among these lifts to solve ECDLP.
There are two difficulties in the index calculus algorithm for ECDLP. First of all, one needs to lift $E(F_p)$ to $E(Q)$ having many independent rational points of small height. Secondly one needs to lift points from $E(F_p)$ to $E(Q)$. Both these problem are very difficult, probably more difficult than the original ECDLP. As it is a well-known fact that $E(K)$ is finitely generated group for any number field K, it is difficult to find curve $E(Q)$ of high rank. Furthermore, even if one could find curve $E(Q)$ of very high rank, generators of $E(Q)$ would never be small enough. Thus can't allow the lifting problem to be solved in sub exponential time.

A conclusion made by Silverman and Suzuki [16] is that the index calculus will not work for solving ECDLP because it is not possible to lift $E(F_p)$ to the curve $E(Q)$ having many independent points of sufficiently small height.

4.6 Xedni Calculus for ECDLP:

In 1998, J Silverman proposed a new type of algorithm to attack the ECDLP [17]. He called it Xedni Calculus because it *"stands index calculus on its head"*". Core idea of Xedni calculus algorithm is as follows: first we choose points in $E(F_p)$ and lift them to points in Z^2. Subsequently we choose a curve $E(Q)$ containing the lift points using Mestre's method [18] in reverse to make rank $E(Q)$ small. However index calculus follows reverse approach. Xedni calculus for ECDLP has not been tested in practice.

It is important to note that soon after Silverman proposed the xedni algorithm, Koblitz showed that a modified version of Silverman's xedni algorithm could be used to attack both the DLP and Integer Factorization Problem (IFP). Practically it is difficult to implement modified version of Index calculus algorithm. If this algorithm turned out to be practical, all form of public key cryptographic systems being used currently can be broken easily in comparison to existing algorithms to break the systems.

4.7 Weil Descent and Cover Attack:

Weil descent has been first introduced in cryptography by Frey [19]. The idea is to view an abelian variety A of dimension $d$which is defined over an extension field K/k as an abelian variety $W_{K/k}$ of dimension d $.[K : k]$ over k. If $W_{K/k}$ turnsout to be the Jacobian of a curve $C_{|k}$ or can be mapped into such a Jacobian, then the discrete logarithm in *A(K)* can be transferred to $Jac_C(k)$, where due to the existence of efficient index calculus algorithms it maybecome much weaker. The main difficulty of this Weil descent method is to find the curve *C*.

4.8 Some attacks for ECDLP on particular curve

Some attacks have been proposed for ECDLP on particular types of curves. Elliptic curve $E(F_p)$ over finite field F_p of order *p* (where *p* is prime) is called anomalous curve. The attack on "anomalous" curves was proposed by Smart [20], and Satoh and Araki [21]. A similar attack has been proposed by Semaev [22] to solve the ECDLP in subgroups of order *p*, where *p* is the characteristic of the field of definition of the curve. In these attacks they use the theory of elliptic curves defined over p-adic numbers, *E(Q)*.

Menezes, Okamoto and Vanstone [23] proposed an idea that uses the Weil Pairing to convert a discrete log problem in $E(F_q)$ $(q = p^m)$ to one in ${F^*_q}^m$. This idea is known as MOV attack. Since discrete log problems in finite fields can be attacked by index calculus methods, they can be solved faster than elliptic curve discrete log problems, as long as the field ${F^*_q}^m$ is not much larger than F_q. For supersingular curves (curves with *t* equivalent to *0 mod p*, characteristic of the field F_q), we can usually take $m=2$, so discrete logarithm problems can be computed more easily for such types of curves than for arbitrary elliptic curves.

V. Pollard's Rho Algorithm For ECDLP

The Pollard's Rho algorithm was first proposed by mathematician J. M. Pollard in 1978[3]. The general pollard's rho algorithm was designed to work for the DLP problem; after it has been opted for ECDLP problem.

Let *P* and *Q* be two points on elliptic curve *E,* and *Q* be in the cyclic subgroup generated by *P*, $Q \in \langle P \rangle$. Let *n* be the order of the generator point *P*. We have to find the positive integer *l* such that $Q= l P$.
The algorithm is given below. In the algorithm we make the use of an iteration function to iterate from one point to another in subgroup.

Iteration Function:

Iteration function defines the random cyclic walk of points over the subgroup [2]. Many researchers have presented different iteration functions. Teske [6], Weiner and Zuccherato [7], Gallant et al. [8], and Bailey [9] have presented different forms of iteration functions with their advantages. Motive of all iteration functions is that they update a state of triplet *(c, d, X)*.

***Algorithm**:*

```
Input: $P \in E(F_q)$ of prime order $n$, $Q \in \langle P \rangle$.
Output: The discrete logarithm $l = \log_P Q$.
1. Consider that the subgroup is partitioned into three sets.
2. Start with initial triple $(X_0, c_0, d_0) = (O, 0, 0)$, where
                              $X_0 = c_0P + d_0Q$.
3. Set $X_0'' = X_0' = X_0$,
       $c_0'' = c_0' = c_0$,
       $d_0'' = d_0' = d_0$.
4. Repeat the following
    a. Check in which set point $X'_l$ belongs out of three.
    b. Use the following iteration function to iterate from $X'_i$ to $X'_{i+1}$.
```

$$X'_{i+1} = \begin{cases} Q + X'_i, & X'_i \in S_1 \\ 2X'_i, & X'_i \in S_2 \\ P + X'_i, & X'_i \in S_3 \end{cases}$$

$$c'_{i+1} = \begin{cases} c'_i, & X'_i \in S_1 \\ 2c'_i \bmod n, & X'_i \in S_2 \\ c'_i + 1 \bmod n, & X'_i \in S_3 \end{cases}$$

$$d'_{i+1} = \begin{cases} d'_i + 1 \bmod n, & X'_i \in S_1 \\ 2d'_i \bmod n, & X'_i \in S_2 \\ d'_i, & X'_i \in S_3 \end{cases}$$

```
    c. For $i$ from $1$ to $2$ do
        i. Check in which set point $X''_i$ belongs out of three.
        ii. Use the following iteration function to iterate from $X''_i$ to $X''_{i+1}$.
```

$$X''_{i+1} = \begin{cases} Q + X''_i, & X''_i \in S_1 \\ 2X''_i, & X''_i \in S_2 \\ P + X''_i, & X''_i \in S_3 \end{cases}$$

$$c''_{i+1} = \begin{cases} c''_i, & X''_i \in S_1 \\ 2c''_i mod\ n, & X''_i \in S_2 \\ c''_i + 1\ mod\ n, & X''_i \in S_3 \end{cases}$$

$$d''_{i+1} = \begin{cases} d''_i + 1\ mod\ n, & X''_i \in S_1 \\ 2d''_i mod\ n, & X''_i \in S_2 \\ d''_i, & X''_i \in S_3 \end{cases}$$

Until $X_m' = X_m''$ for some m.

5. If $d_m' = d_m''$ then failure
   Else compute,
   $l = (c_m' - c_m'')(d_m'' - d_m')^{-1} \bmod n.$

The main idea behind the Pollard's rho algorithm is to define an iteration function that defines a random cyclic walk over a graph. Floyd's cycle detection algorithm is used to find the collision of points. Use of Floyd's cycle detection algorithm avoids the need of excessive storage of triplet *(X, c, d)*. If Floyd's cycle detection algorithm is not used then every tripletneeds to be stored in memory and have to do comparison every time a new value of *X* is generated. Using the Floyd's algorithm, we try to find the colliding points $(c_m'P + d_m'Q)$ and $(c_m''P + d_m''Q)$ for some scalar *m* and then we can find the value of the scalar '*l*' as follows

$$c_m'P + d_m'Q = c_m''P + d_m''Q$$

Then,

$$(c_m' - c_m'')P = (d_m'' - d_m')Q = (d_m'' - d_m')l\,P$$

And so

$$l = (c_m' - c_m'')\,(d_m'' - d_m')^{-1} \bmod n \text{.........} (6)$$

By the birthday paradox, the collision is obtained in approximately $\sqrt{\pi n/2}$ number of iterations. Use of Floyd's detection algorithm reduces the storage requirement, keeping the time requirement unchanged.
The above algorithm has been devised in such a way that it will require negligible amount of memory. At any instance only the current values of X'_i and X''_i will be in memory for comparison. But the total computation has been increased, due to which it is not possible to solve large ECDLP problems.

VI. Parallel Pollard's Rho Algorithm

The technology enhancement in parallel computing allows to take the advantage of available resources more efficiently. Thus resulting into great speedup in run time. CPU clusters allowsrunning the code indistributed manner on multiple machines in MIMD structure. MPI technology provides a sophisticated way of communication between these machines. As the arithmetic operations in the group of elliptic curve require more computational cost, specialized libraries can be used to perform those operations. GMP, GP- PARI, rosing, miracl etc. are some of the examples of such libraries. In our implementation we had made the use of GP- PARI [14] library which is based on the GMP library for multi-precision operations. Use of such libraries also helps in dealing with the large underlying finite fields.

The idea of parallel Pollard's rho algorithm was first put forward by Van Oorschot and Wiener [4] in 1999. The idea yields a factor *M* speedup when *M* processors are employed. The idea is to allow the sequences $\{X_i\}_{i>=0}$to collide with one another necessarily generated by different processors. More precisely, each processor randomly selects its own starting point X_0, but all use the same iteration function *f* to compute subsequent points X_{i+1}. Thus, if the sequences generated from two different processors (processes) ever collide, then, two sequences will be identical from that point on.

Iteration function which is responsible for a pseudo random walk in group and the distinguishing point property are the important part of the algorithm. The iteration function is explained in detailed in section 5.

Distinguished Points:

In order to parallelize the attack efficiently, distinguished points concept is used. The concept of distinguished point has been proposed in the same paper by Van Oorschot and Weiner [4]. Distinguished points are a subset of points, which satisfy a particular condition. Any such condition can be specific number of leading zero digits of a point's x-coordinate or a particular bounded range of Hamming weights in normal basis representation. Those distinguished points are stored in a central database, which are computed in parallel by processors.

Parallelized Pollard's rho algorithm for the ECDLP:

```
Input: $P \in E(F_q)$ having prime order $n$, $Q \in <P>$.
Output: The discrete logarithm $l = \log_P Q$.
1.  Consider that the subgroup is divided into three sets.
2.  Select a distinguished property for points in $<P>$.
3.  Each of the $M$ processors does the following:
    a.  Select $c_0, d_0 \in [0, n-1]$ and compute $X_0 = c_0P + d_0Q$ as initial starting point.
    b.  Repeat the following:
        i.   If $X_i$ is distinguished then send $(c_i, d_i, X_i)$ to the central server.
        ii.  Check in which set point $X_i$ belongs out of three.
        iii. Use the same iteration function to iterate from $X_i$ to $X_{i+1}$ explained in sequential version in section 5.
    Until the server receives some distinguished point $X_m$ for the second time ($m$ be any scalar).
4.  Let the two triples associated with $X_m$ be $(c_m', d_m', X_m)$ and $(c_m'', d_m'', X_m)$.
5.  If $d_m' = d_m''$ then return failure
    Else compute $l = (c_m' - c_m'')(d_m'' - d_m')^{-1} \bmod n$.
```

The use of PARI library [14] on each processor will help in the operations performed in the elliptic curve group. Each node iterates through points in the cyclic subgroup and sends only those points to root that satisfy the distinguished property. Thus, the storage requirement on the central server can be reduced by hardening the property, but which will eventually reduce the chances of collision. Early collision can be found if the distinguishing property allows sending all points to central server, thus resulting in consequence of high storage requirement.

VII. Observation And Results

In modern days, the major cryptographic and cryptanalytic techniques require high computational power. If these techniques are executed on singlestand-alone PC, then it will take a lot of time in completing the task. Therefore for executing these techniques within the time frame, we can use multiple parallel platforms which include cluster of standard CPUs, FPGAs and Graphics Processing Unit (GPUs). The idea of the parallel Pollard's Rho algorithm was put forward by P. C. van Oorshant and M. J. Wiener in paper [4]. In that paper authors showed that if *M* processors are used to solve the ECDLP using parallel Pollard's rho algorithm, then a speedup of *M* can be achieved. In that paper author also explained that the expected number of steps before collision is encountered will be- $\frac{\sqrt{\pi n}}{2M}$ where *n* is the order of field and *M* are the number of processors used.

Use of proper distinguished point property also affects the overall performance of the algorithm. Let θ be the proportion of points in $<P>$ having distinguishing property. Then before collision, the expected number of elliptic curve operations performed by each processor is - $\frac{1}{M}\sqrt{\frac{\pi n}{2}} + \frac{1}{\theta}$.

1. GPGPU Based Computing:

Graphics Processing Unit (GPU) can be used for general purpose computing also. The technique which uses GPUs to perform computation traditionally handled by the CPU (i.e general purpose computation), is referred as General-purpose graphics processing units (GPGPU).NVIDIA developed parallel programming architecture for GP-GPU on its graphics processing unit. This architecture is known as CUDA (Computing United Device Architecture). CUDA is accessible to software developers as variants of many industry standard programming languages. For execution on GPU, programmer may use C for CUDA (compiler developed by NVIDIA which is an extension of C with certain restrictions).

Significant work has been done in solving the ECDLP using the power of GPGPU. Lei Xu and Dongdai Lin [24] have shown that using GPU technology, speedup of more than one hundred times can be achieved. Scalar multiplication is one of the costly operation in elliptic group. Eric M. Mahe and Jean-Marie Chauvet [25] have shown that the scalar multiplication on a 255-bit prime elliptic curve can be done using GPGPU technology in microseconds. Also, S. B. Khemnar, R. Chaudhary and D. B. Kulkarni [13] have shown that the inversion operation can be performed using GPGPU in an efficient way resulting in overall speedup of attack. In that work they were able to run the algorithm for small field size.

2. FPGA Based Computing:

A Field –Programmable Gate Array (FPGA) is an integrated circuit which can be configured by the consumer or designer using the Hardware Description Language (HDL) as per their need. The programmable logic blocks can be connected together to form different configurations which can be used as logic gates or combinational functions or a memory block. FPGA's have large resources of logic gates and RAM blocks allowing them to perform very fast I/O and bidirectional data transfer.

The ECDLP problems based on Koblitz curve or binary curves can be efficiently solved using FPGA cluster as the group operations can be performed using bit shifting. Erich Wenger and Paul Wolfger [2] presented a FPGA architecture that can be used to solve the ECDLP of Weierstrass curves defined over Koblitz curve and binary curves.They have solved the discrete logarithm problem of the elliptic curve *sect113r1*, a previously standard binary curve.

3. Computing Using Computer Cluster:

The availability of low cost microprocessors, high speed networks and software for high-performance distributed computing has result in the emergence of the computer cluster technology. Set of interconnected computers working as a single unit is called as computer cluster. Also development of communication technologies such as MPI, OpenMP has resulted in wide boosting of this computing.

The ECDLP problem using parallel Pollard's Rho method can be very efficiently solved using the computer cluster. Most of the challenges presented by the CERTICOM [10] has been solved using cluster computing. Joppe W. Bos and his colleagues [26] have solved the 112-bit prime elliptic curve discrete logarithm problem using the combination of computer cluster and FPGA architecture, which is the largest ECDLP problem solved in prime field elliptic curve.

4. PARI/GP Implementation:

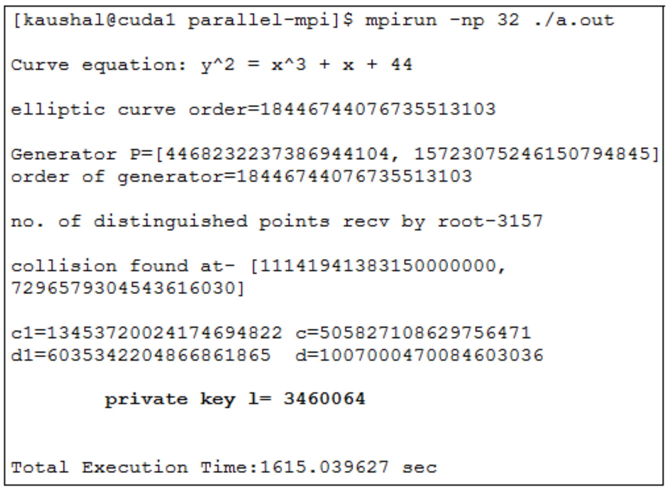
PARI/GP is a widely used computer algebra system [14] designed for fast computation in number theory (factorization, algebraic number theory, elliptic curves, etc.). PARI/GP is also available as a C library to allow faster computation. PARI/GP uses GMP library as its base library. GMP is a free library for arbitrary precision arithmetic. It is useful while operating on signed integers, rational numbers, and floating-point numbers.

While dealing with large fields, algebra of large number (more than 64 bit) has to be taken into consideration. PARI/GP provides functionality for such issue. In our implementation, we had made extensive use of PARI/GP functions. MPI technology has been used to parallelize the algorithm. MPI technology allows to run the program on cluster of machine.

In parallel Pollard's Rho method, as mentioned before, each of the processor starts with different initial point and use the same iteration function to generate a pseudo random sequence. The iteration function used in our implementation is devised such that it takes negligible amount of memory, but at the cost of more cycles (iterations). If we try to reduce the number of iterations then we can go further in terms of size but it will require approximately $\sqrt{n}$ number of point storage for n problem size.The implementation is explained in following example.

Let us consider ECDLP problem in an elliptic group of 64 bit prime. Figure shown below (Fig. 2) gives the output screen of the implementation on a single machine (GHz capacity).

Each of the 31 nodes (1 node is root node) will start from different starting point and iterate through group, if the distinguished property is satisfied by the point, then that point along with the values of c and d are sent to central server, which saves those points in an array. When some same point is received by the central server second time, collision is said to be occurred. After the collision is occurred, respective values of c and d are taken and the private key value is calculated. In the above figure we can see that due to hard distinguish point property, total numbers of points received by root node are very less.



```
[kaushal@cuda1 parallel-mpi]$ mpirun -np 32 ./a.out

Curve equation: y^2 = x^3 + x + 44

elliptic curve order=18446744076735513103

Generator P=[4468232237386944104, 15723075246150794845]
order of generator=18446744076735513103

no. of distinguished points recv by root-3157

collision found at- [11141941383150000000,
7296579304543616030]

c1=13453720024174694822 c=505827108629756471
d1=6035342204866861865  d=1007000470084603036

        private key l= 3460064


Total Execution Time:1615.039627 sec
```

Fig 2: Results on single machine

We have implemented the algorithm on a cluster of 256 octa-core computers having PFLOPs capacity. We have solved the ECDLP problem based on prime field up to 85-bit. It can be solved further on the same infrastructure using all computation power of 256 machines. Following table shows the implementation results on a cluster-

Prime Field Size (bits)	Total processes	Time (sec)
70	4*8=32	4330.040
75	16*8=128	7225.267
80	16*8=128	15880.519
85	16*8=128	190189.061

Table 1: Results obtained on cluster

VIII. Conclusion And Future Work

This paper gives a brief review of the work related with Pollard's rho algorithm and parallel implementation of the Pollard's Rho algorithm. After the introduction of the CERTICOM [10] challenges, several researchers have tried to solve them using parallel pollard's rho algorithm on various parallel platforms [4] [7] [8]. Parallel pollard's rho algorithm makes the efficient use of available resources and can give a speedup of *M* when *M* processors are employed.

The selection of the iteration function and the distinguishing property affects the performance of the algorithm. Also using specialized libraries to perform multi-precision operation also helps in achieving good speedup. This paper discusses the background of elliptic curve cryptography in which finite fields are the basic underlying fields on which the cryptosystem is build. The solvability of the elliptic curve cryptosystem reduces to the solvability of the ECDLP problem in that group. Thus different attacks on the ECDLP have been studied. Dealing with the ECDLP problem based on the large underlying finite field requires large computational resources. Using Parallel Pollard's rho algorithm on CPU cluster is a better approach to such instances.

Solving ECDLP based on large finite field requires huge resources and special algorithm (or hardware) have to be considered in order to perform arithmetic's on large integers. Making the use of arithmetic libraries helps in such scenarios. The parallel Pollard's rho algorithm can be modified to run on a hybrid cluster of CPU – GPU combination. Also, selecting proper iteration function and proper distinguished property as per the parallel platform can also result in good speedup.

Acknowledgment

Authors express our sincere thanks to all the authors, whose papers are in the area of elliptic curve cryptography, ECDLP and Parallel Pollard's Rho. Also authors would like to thanks Director of SAG at DRDO, New Delhi who permitted to work at their premise.

References

[1]. A. J. Menezes, D. Hankerson and S. Vanstone, "Guide to Elliptic Curve Cryptography", Springer-verlag New York, inc., 2003.

[2]. E. Wenger and P. Wolfger, "Solving the Discrete Logarithm of a 113- bit Koblitz Curve with an FPGA cluster", SAC, volume 8781 of LNCS, pages 363-379, Springer, 2014.

[3]. J. M. Pollard, "Monte Carlo Methods for Index Computation", Mathematics of Computation, Vol. 32, No. 143, American Mathematical Society Jun 1978.

[4]. P. C. van Oorshant and M. J. Wiener, "Parallel Collision Search with Cryptanalytic Applications", Journal of Cryptology, volume 12 No. 1, pages 1-28, 1999.

[5]. R. Harley, "Elliptic Curve Discrete Logarithms: ECC2K-108", 2000.

[6]. E. Teske, "Speeding up Pollard's Rho Method", In Algorithmic Number Theory, volume 1423 of LNCS, pages 541-554, Springer, 1998.

[7]. M. J. Wiener and R. J. Zuccherato, "Faster Attacks on Elliptic Curve Cryptosystems", In selected areas in Cryptography- SAC, volume 1556 of LNCS, pages 190-200, Springer, 1999.

[8]. R. Gallant, R. Lambert and S. Vanstone, "Improving the parallelized Pollard lambda search on anomalous binary curves", Mathematics of Computation of the American Mathematical Society, volume 69 No. 232, pages 19-46, 2002.

[9]. D. V. Bailey, B. Baldwin, L. Batina, D. J. Bernstein, P. Birkner, "The certicom challenges ECC 2-X", IACR Cryptology ePrint Archive, Report 2009/466, 2009.

[10]. Certicom Research, "The Certicom ECC Challenges", https://www.certicom.com/index.php/the-certicom-ecc-challenge, Nov 1997.

[11]. N. Koblitz, A. Menezes, S. Vanstone, "The State of Elliptic Curve Cryptography", Springer, 2000.

[12]. V. S. Miller, "Use of Elliptic Curves in Cryptography", Advances in Cryptography- CRYPTO' 85, LNCS 218, pp. 417-416, 1986.

[13]. S. B. Khemnar, R. Chaudhary, D. B. Kulkarni, "CUDA based Implementation of Parallelized Pollard's Rho Algorithm for ECDLP: A Review", Data Mining and Knowledge Engineering, Vol 7, No.5, 2015.

[14]. http://pari.math.u-bordeaux.fr

[15]. D. Shanks,"Class Number, a Theory of Factorization, and Genera",Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Book, NY, 1969), pages 415-440. Amer. Math. Soc., Providence, RI, 1971.

[16]. J. H. Silverman and J. Suzuki,"Elliptic Curve Discrete Logarithms and the Index Calculus", Advance in Cryptology-ASIACRYPT' 98. LNCS, vol 1514, pp. 110-125. Springer-Verlag, Berlin, 1998.

[17]. J. H. Silverman,"The Xedni Calculus and the Elliptic Curve Discrete Logarithm Problem",Design Codes and Cryptography, 20(1):5-40, 2000.

[18]. J. F. Mestre, "Formules Explicites et Minoration de Conducteurs de Varietes Algebriques",Compositio Mathematica, 58, 209-232, 1986.

[19]. G. Fery, "How to disguise an elliptic curve (Weil descent)", Talk at the 2nd Elliptic Curve Cryptography Workshop (ECC), 1998.

[20]. Smart N.P,"The Discrete Logarithm Problem on Elliptic Curves of Trace One"Cryptology Journey, 12(3):193-196, 1999.

[21]. Satoh T. and Araki,"Fermat Quotient and the Polynomial Time Discrete Logarithm for Anomalous Elliptic Curve", Comm. Math. Univ. Sancti. Pauli, 47:81-92, 1998.

[22]. Semaev I. A,"Evaluation of Discrete Logarithms on Some elliptic Curves",Math. Comp., 67: 353-356, 1998.

[23]. Menezes A. J., Okamoto T. , Vanstone S. A,"Reducing Elliptic Curve Logarithms to Finite field",IEEE Trans. Info. Theory, 39: 1639-1646, 1993.

[24]. Lei Xu and Dongdai Lin, "ECDLP on GPU", IACR Cryptology ePrint archeive, 2011.

[25]. Eric M. Mahe and Jean-Marie Chauvet, "Fast GPGPU-based Elliptic Curve Multiplication", IACR Cryptology ePrint archeive, 2014.

[26]. Joppe W. Bos, Marcelo E. Kaihara, Thosten K leinjung, Arjen K. Lenstra and Peter L. Montgomery, "Solving a 112-bit prime Elliptic Curve Discrete Logarithm Problem on game Consoles using Sloppy Reduction", International Journal of Applied Cryptography, Vol. 2:Issue 3: Pages 212-228, 2012.