INTERNATIONAL JOURNAL OF MATHEMATICAL SCIENCES AND APPLICATIONS
Volume 5 • Number 1 • January-June 2015

GENERIC ATTACKS ON ELLIPTIC CURVE DISCRETE LOG PROBLEM

Sanjay Kumar[1] and S.K. Pal[2]

SAG, DRDO, Delhi, [1]E-mail:skparsade@gmail.com,[2]E-mail:skptech@yahoo.com

Abstract*: The security of Elliptic Curve Cryptography relies on the difficulty of Elliptic Curve Discrete Logarithm Problem(ECDLP). ECC can yield the same level of security with a 164-bit key as RSA with 1024-bits. Because ECC helps to establish equivalent security with lower computing power and resources, it is suitable for constrained and lightweight environments, such as cellular phone, and smart card, RFID etc. where available resources are limited. However, a number of attacks on ECDLP have been proposed*

In this paper we analyze the generic attackson Elliptic Curve Discrete Log Problem. We present details of Pholig Hellman method and Pollard rho method and also mention the speed up by group automorphism, parallel methods and massive hardware exploration for the Pollard-Rho method. We also propose scope for improvements in the Pollard- Rho method for ECDLP.

Keywords*: Finite Field,Automorphism,Public Key Cryptography,DLP, Elliptic Curves.*

1. INTRODUCTION

In 1976 W.Diffie and M.Hellman [1] invented an agreement protocol that allows two users to exchange a secret key over an insecure channel without any prior contact. This event is commonly considered the birth of public-key cryptography. Elliptic Curve Cryptography (ECC), proposed independently in 1985 by Neal Koblitz [2] and Victor Miller [3]. ECC algorithms can achieve the same level of security with much shorter keys. Because of the shorter key length, ECC algorithms run faster, require less space, and consume less energy. These advantages make ECC a better choice for public-key cryptography, especially in resource constrained systems such as wireless and mobile devices for pervasive computing.

In 1997, Certicom issued several challenges [12] in order to "increase the cryptographic community's understanding and appreciation of the difficulty of the ECDLP." All challenges over fields of109 bits were solved between April 2000 and April 2004 in distributed efforts. The challengesover fields of 131 bits are open; in particular there are two challenges over $F_2{}^{131}$ and one over F_p, where p is a 131-bit prime. One of the curves over $F_2{}^{131}$ is a Koblitz curve [13]. Koblitz curves are of interest for implementations because they support particularly efficient scalar multiplication. NIST has standardized 15 elliptic curves of which 5 are Koblitz curves. The best known attacks on the elliptic curve discrete log problem are generic attacks based on Pollard's-Rho method. The improvements given by wiener and Zuccherato[9], van Oorchot and Wiener[10], and Gallant, lambert and Vanstone [11] showed how the computationcan be efficiently parallelized and how group automorphism can speed up the attack.

Remaining part of the paper is organized as follows. Section 2 describes the elliptic curve over the finite field. Section 3 defines the elliptic curve discrete log problem. Section 4 introduces ElGamal Scheme based on ECDLP.Sections 5 describe the various attacks on ECDLP. Section 6 shows the ECDLP challenges and Section 7 presents the scope of improvements in the Pollard-Rho method.The paper concludes in Section 8 showing the possibility of improvements for future research.

2. BASICS OF ELLIPTIC CURVES

Let F_q be the finite field, where q is a prime or power of prime. An elliptic curve $E(F_q)$ over the field F_q is a set of points which satisfy a (Weierstrass) equation of the form

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4\,x + a_6$$

where $a_1, a_2, a_3, a_4, a_6 \in F_q$. For simplicity assume $q = p$ prime ($p > 3$), then with an admissible change of variablesin the preceding equation we obtain a more familiar representation:

$y^2 = x^3 + a_x + b$, and satisfies the condition $4a^3 + 27b^2 \neq 0$

Choice of the coefficients $a, b \in F_q$ should be proper otherwise a poor choice could generate a curve which is easier to attack for the cryptanalyst. The number of points $\#E(F_q)$ in an elliptic curve satisfies Hasse's bound: $|\#E(F_q) - (q + 1)| \leq 2\sqrt{q}$.

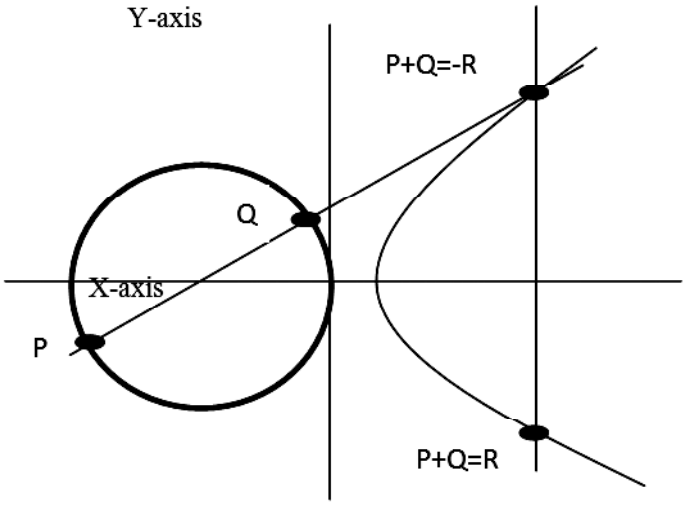

Figure1: Point Addition in Elliptic Curve

3. ELLIPTIC CURVE DISCRETE LOG PROBLEM

ECDLP is a particular case of DLP, there is no generic algorithm with sub-exponential running time that solves it. Let $E(F_q)$ be an elliptic curve over a finite field F_q, a point $P \in E(F_q)$ of order n, for any point $Q \in <P>$ (the subgroup generated by P), the problem is to find out the integer $k \in [1, n-1]$ such that $Q = k \cdot P$. the integer k is called the discreate log problem of Q to the base P.

4. SCHEMES BASED ON ECDLP

4.1 Elliptic Curve ElGamal Schemes

The Schemes consists of a key generation algorithm based on the elliptic curve analogue of Diffie-Hellman, and an encryption and decryption algorithm based on the elliptic curve analogue of ElGamal.

Public Parameters known to the publicly are given as follows:

1. The elliptic curve E.
2. A curve-point P, which generates a cyclic subgroup of E.

3. The order n of the cyclic subgroup of E.

P is a generator of the cyclic subgroup of E:

$$\langle P \rangle = \{\infty, P, 2P, ..., (n-1)P\}$$

Key-pair Generation: As input we give the public parameters from before. The procedure runsas follows:

1. Choose a random integerd $\in \{1, ..., n - 1\}$
2. Compute $Q = dP$.
3. The pair (Q, d). public key = Q, private key = d.

We observe that computing the private key d from a given public-key Q requires solving the discrete logarithm problem.

Encryption: Besides the public parameters, we are given the public-key Q. We want to encrypt a message M, which is encoded as a point on the elliptic curve (we elaborate how to do the encodingin the next section):

1. Choose a random integer $k \in \{1,..., n-1\}$
2. Compute $C_1 = kP$
3. Compute $C_2 = M + kQ$
4. Output ciphertext: (C_1, C_2)

One observation is that since k is chosen at random, $C_2 = M + kQ$ actually also appears to berandom.

Decryption: We are given the ciphertext-pair (C_1, C_2), and the private key d. The goal is to reconstruct the message-point M:

$$C_2 - dC_1 = M + kQ - dkP$$
$$= M + kQ - k(dP) = M + kQ - kQ = M$$

this is the plaintext message.

5. ATTACKS ON ECDLP

Elliptic curve discrete log problem is computational hard. There is no sub-exponenational algorithm. However the following exponenational algorithms for attacks on ECDLP are given as follows.

5.1 Generic Methods

The best known algorithms to solve the ECDLP are called the generic attacks. We are given general attacks as follows:

5.1.1 Exhaustive Search

In this method we compute successive multiples of P until point Q is found. This method can take upto n steps in the worst case. This attack is completely in efficient when dealing with concrete cryptographic situations.

5.1.2 Baby- Step, Giant -Step Algorithm

Similar in nature to the setting of the DLP, this attack uses a combination of computational power and storage in an attempt to solve the ECDLP. Let $E(F\_q)$ be an elliptic curve with generator P. Suppose that P has order n, and let

$Q \in \langle P \rangle$. Suppose that we want to solve $Q = kP$. Set $m = \lceil n \rceil$ and compute mP. We now make a list of iP for $0 \le i < m$, and store this list. We can now compute $Q–j(mP)$, for $0 \le j \le m^{-1}$ until we have found a match from the list that we have stored. Once we have a match we then have the following:

$$iP = Q - j(mP)$$

hence,

$$Q = iP + j(mP).$$

Therefore we have solved the *ECDLP* since $k \equiv i + jm \bmod n$.

Again this attack takes at mostn operations and storesn values in a list to check for a match. Thus the expected running time of this algorithm is $O(n)$ [7].

This algorithm is a time-memory trade-off of the method of exhaustive search. It requires storageabout ?n points, and its running time is roughly n steps in the worst case.

5.1.3 Pollard's λ- Method

This is randomized algorithm due to Pollard [4]. The parallelized λ-method is slightly slower than the parallelized rho method [5]. The λ-method is, however, faster in situations when the logarithm being sought is known to lie in a subinterval [0, b] of [0, n–1], where $b < 0.39n$ [5].

5.1.4 Pohlig- Hellman Method

The setup for this attack is similar to the setup in the case of the DLP. Suppose that we are given an elliptic curve $E(F_q)$, a point $P \in E(\text{Eq})$ of order n and $Q \in <P>$. We again want to solve for the unique integer k such that $Q = kP$. Suppose further that we know the factorization of n, say $n = \prod_{i=1}^{r} l_i^{ei}$, where each li is prime. Similar to the situation in the case of the DLP, we will now attempt to solve for k by reducing the problem to solve for values of $k_i \equiv k \bmod l_i^{ei}$ for $0 \le i \le r$, which gives us asystem of congruences modulo each prime l_i, namely

$$k \equiv k_1 \bmod l_1^{e1}$$

$$k \equiv k_2 \bmod l_2^{e2}$$

$$k \equiv k_3 \bmod l_3^{e3}$$

.

.

.

$$k \equiv k_r \bmod l_r^{e_r}.$$

The Chinese Remainder Theorem guarantees the existence of a unique solution, namely k.

Let's take a closer look at how this works. For the moment *flx* a prime say$l_1^{e_1}$. We compute $k_1$as follows.

We compute $k1$ as follows. We write the base-l_1 representation *of*k_1,

$$k \equiv a_0 + a_1 l_1 + a_2 l_1^2 + \ldots + a_{e_1-1} l_1^{e_1-1} \bmod l_1^{e_1}$$

where each $a_i \in [0, l_1 - 1]$.

We begin by computing a list of small values for each prime divisor l_i of n Set $T_i = \{ j([\frac{n}{l_i}]p) : 0 \le j \le l_i - 1\}$. . We will look for a match with these points andvalues that we will determine below. When we find a match we have solved for agiven coefficient in the base–l_1 expansion of k. We can now compute the following,

$$\left[\frac{n}{l_1}\right] Q = \left[\frac{n}{l_1}\right] ([a_0 + a_1 l_1^1 + ... + a_{e_1 - 1} l_1^{e_1 - 1}]P)$$

$$= [a_0] \left[\frac{n}{l_1}\right] P + ([a_1 + a_2 l_1 + ...])\,[n]\,P = [a_0] \left[\frac{n}{l_1}\right] P.$$

Thus we can now look in our list T_1, find the matching point in the list and read off the coefficienta a_0.

To solve for the next coefficient, a_1, we have to change our starting point which can be easily done. Since we have already solved for a_0 we can use it and set $Q_1 = Q - [a_0]P$ then perform the above calculation using Q_1 instead, and shifting by the proper quantity to isolate for a_1. If we multiply (4.2) by $\frac{n}{l_1^2}$, after a_0 has been removed this will then give us

$$\left[\frac{n}{l_1^2}\right] Q_1 = ([a_1 + a_2 l_1 + ...])\,[\frac{n}{l_1}]P = [a_1]\,([\frac{n}{l_1}]P),$$

and again we look in our list T_1 for a matching solution. This then gives us a result for a_1. We continue in this way until we have solved for each coefficient in the base–l_1 expansion of k_1. We then continue and solve for each k_i in the same manner. When this is done we solve the above system of equations and recover the original value of k in our original problem $Q = [k]P$.

The expected running time of this algorithm is $O(\sqrt{l^*})$ [7], where l^* is the largest prime divisor of n. In practice this attack becomes infeasible when n has a large prime divisor. If this is the case, it then becomes difficult to make and storethe list T to find matches, let alone attempting to solve for k^* in its base-l^* expansion.

This algorithm, due to Pohlig and Hellman [6], exploits the factorization of n, the order of the point P. The algorithm reduces the problem of recovering k to the problem of recovering lmodulo each of the prime factors of n; the desired number k can then be recovered by using the Chinese Remainder Theorem.

The implications of this algorithm are the following. To construct the most difficult instance of the ECDLP, one must select an elliptic curve whose order is divisible by a large prime n. preferably, this order should be a prime or almost a prime (i.e. a large prime n times a small integer h). The elliptic curves in the exercises and challenges posed here are all of this type.

5.1.5 Pollard- Rho Method

Let $E(F_q)$ be an elliptic curve and $P \in E(Fq)$. Suppose that P has order n, where n is prime, and let $Q \in <P>$. Suppose that we want to solve $Q = kP$. In this attack we will attempt to find distinct pairs of integers (a, b) and (a^*, b^*) modulo n such that $aP + bQ = a^*P + b^* Q$. Rearranging this we can obtain a solution for k, name $lyk \equiv (a - a^*)\,(b^* - b)^{-1} \bmod n$.

Method 1: for finding these pairs of integers is to simply select $a, b \in [0, n-1]$ uniformly at random, compute the point $[a]P + [b]Q$, and then store the triple $(a,b, aP + bQ)$. We continue to generate pairs (a, b) uniformly at random and check these against all previously stored triples until we find a pair (a^*, b^*) with $a^* P + b^* Q$ where $(a, b) \neq (a^*, b^*)$. When this happens we have solved the ECDLP and as mentioned above, we can rearrange $aP + bQ = a^* P + b^* Q$ as $[a - a^*]P = [b^* - b]Q = [b^* - b](kP)$, and thus $k \equiv (a - a^*)(b^* - b)^{-1} \bmod n$.

Again as in the setting of the DLP, the birthday problem governs the expected running time of this algorithm. This first method gives an expected running time of $O(\sqrt{\frac{\pi n}{2}})$ [8], but unfortunately requires approximately $O(\sqrt{\frac{\pi n}{2}})$ amount of storage for the triples that we have computed.

Method 2: A second approach that has roughly the same running time, but uses less storage. Instead of storing a list of triples, we define a function $f : \langle P \rangle \to \langle P \rangle$ so that for any $X \in \langle P \rangle$ and $a, b \in [0, n-1]$ with $X = aP + bQ$, we can easily compute $f(X) = X^*$ and $a^*, b^* \in [0, n-1]$ with $X^* = a^* P + b^*]Q$. One way to define such a function is to partition $\langle P \rangle$ into L sets of roughly equal size, say $\{S_1, S_2, S_L\}$.

We define a second function H so that $H(X) = j$ if $X \in S_j$. Then $a_j, b_j \in [0, n-1]$ are chosen uniformly at random for $1 \leq j \leq L$. Now our function $f : \langle P \rangle \to \langle P \rangle$ is defined by $f(X) = X + a_j P + b_j Q$, where $j = H(X)$. So, if $X = aP + bQ$, then $f(X) = X^* = a^* P + b^* Q$ where $a^* = a + a_j \bmod n$ and $b^* = b + b_j \bmod n$ [8]. This then determines a sequence of points in $\langle P \rangle$. Since $\langle P \rangle$ is finite we will eventually obtain a collision, thus obtaining our pairs of integers (a, b) and (a^*, b^*), and so enabling us to solve the ECDLP.

As mentioned, this approach has a similar running time to the first, but requires less storage, since we are no longer required to store ordered triples in order to find a collision. The strategy of the algorithm is to produce a sequence of randomly generated terms (R_i, a_i, b_i), where R_i a point on the curve is E and $a_i, b_i \in F_P$, over which the elliptic curve is defined. Since $E(F_P)$ is a finite group, the sequence eventually becomes periodic and loops back to an earlier term in the sequence. We use this periodicity to solve the ECDLP.

5.2 Speedingup Pollard-Rho Method: Speed up of Pollard-Rho Method Given by the Following Methods

5.2.1 Speeding up by Group Automorphism

The speed up Pollard's ρ-Method by using group automorphisms is such that Let $\psi : \langle P \rangle \to \langle P \rangle$ be an auto morphism of groups, where $P \in E(F_q)$ has order n. Suppose that ψ has order t, in other words, t is the smallest positive integer such that $\psi^t(R) = R$ for all $R \in \langle P \rangle$. In this way we can define an equivalence relation on $\langle P \rangle$ by the following. $R_1 \sim R_2$ iff $R_1 = \psi^j(R_2)$ for some $j \in [0, t-1]$. We now define the equivalence class [R] containing $R \in \langle P \rangle$, simply as powers of $\psi(R)$, that is,

$$[R] = \{R, \psi(R), \psi^2(R), ..., \psi^{l-1}(R)\},$$

where l is the smallest positive divisor of t such that $\psi^l(R) = R$.

We have to defined function f on the equivalence classes of $\langle P \rangle$ rather than just points, to speed up our calculations. To do this we have to choose representatives for each class [R], denote this by $\overline{R}$. Then we define a new function on our representatives by setting $g(R) = \overline{f(R)}$. So suppose we know an integer $\lambda \in [0, n-1]$, such that $\psi(P) = \lambda P$. Since ψ is an automorphism, $\psi(R) = \lambda R$ for all $R \in \langle P \rangle$. Thus if we know integers a and b such that $X = aP + bQ$, then we can efficiently determine integers a^* and b^* such that $\overline{X} = a^* P + b^* Q$. Since if we have that $\overline{X} = \psi^j(X)$, then $a^* = \lambda^j a \bmod n$ and $b^* = \lambda^j b \bmod n$ [8]. We can now use g and the equivalence classes

in the parallelized version of the algorithm above and obtain a speed up. If each equivalence class has size approximatelyt, then we've reduced the search space by a factor of approximately $\frac{n}{t}$, thus making the expected running time of this algorithm $O(\frac{1}{m}\sqrt{\frac{\pi n}{2t}+\frac{1}{\theta}})$ [8].

5.2.2 Speeding by Parallel Methods

A parallel version of the Pollard- Rho algorithm by van Oorchot and Wiener [10] provides speedup in the collision finds process. Suppose we have M processors. we have create a sequence of points in cyclic group generated by $\langle P\rangle$. Denote this sequence as $\{X_i\}_{i\geq 0}$. If two different processors collide, then the two sequences will be identical after words. This trick is to determine by an efficient method of finding a collision. One method is to establish a distinguishing property of a point. When this distinguished point is hit in the sequence, the processor can send the information back to the central server where it can be stored. When the server receives the distinguished point a second time, it can compute the discrete logarithm and terminate the *M* processors. Denote the proportion of points in $\langle P\rangle$ that have this distinguishing property by θ. The expected number of steps per processor before a collision occurs is $\frac{1}{M}\sqrt{\frac{\pi n}{2}}$, and a distinguished point is expected after $\frac{1}{\theta}$ steps. Thus the total expected running time before a collision ofdistinguished points is expected is $\frac{1}{M}\sqrt{\frac{\pi n}{2}}+\frac{1}{\theta}$[8].

5.2.3 Speeding by Massive Hardware

In order to improve the speedup of pollard- Rho attack on ECDLP beyond parallel processing on CPU high performance clusters, GPU based cards were used for optimization of computations for specific modules. GPU clusters and play station 3 units were also used to speed up ECC2K-130 computations by efficient implementation of the multiplication and squaring operations.

However, to attain faster speed of operations, customized hardware solutions on FPGA and ASICS have been proposed. FPGAs have been used for both fields of odd prime characteristics as well as characteristic two fields. This involves optimization of the finite fields operations. With proper designs, FPGAs were shown to outperform ECDLP computations on CPU/GPU clusters.

The best price vs performance results may be obtained for this case by use of ASICS. By carefully designing the iteration function, field arithmetic like polynomial multiplications and normal basis multiplications, communications etc. More practical attacks on ECDLP may be launched within a $1,00,000 budget.

6. CERTICOM CHALLENGES AND ATTACKS

Certicom published a list of ECDLP challenges and prizes in 1997[12]. These challenges range from very easy exercises, solved in 1997 and 1998 to serious cryptanalytic challenges. The last certicom challenge that was publically broken was a 109-bit ECDLP in 2004. Certicom had already predicted the lack of further progress: it had stated in[12] that the subsequent challenges were expected to be infeasible against realistic software and hardware attacks. After this, attacks on specific curves & fields by use of CPU, GPU clusters and FPGA were reported.

7. SCOPE FOR IMPROVEMENTS

There is much scope for improvements in the Pollard rho-method. There are two areas for improvements. In the first area we can improve the partition function such that partioned the function corresponding to the y-coordinate of the elliptic curve point and in the other hand partioned the function corresponding to x-coordinate and get the collision early from the general techniqueand the second area of improvement is in the collision technique using parallel methods. Massive parallelism for this part may be introduced by use of GPU clusters and special hardware FPGA designs.

8. CONCLUSION AND FUTURE WORK

In this paper we have discussed various generic attacks on the elliptic curve discrete log problem and mention the presently used algorithms like the pohlig- Helman and Pollard- Rho for attacks on ECDLP. We have also proposed the scope for improvements in the Pollard-Rho algorithm. Our future work would be towards improving attacks on ECDLP for generic and specific cases.

REFERENCES

[1] W. Diffie and M. E. Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, IT-22 (1976), pp. 644-654.

[2] N. Koblitz, "Elliptic Curve Cryptosystems", *Math. of Computation*, **48**, pp. 203-209, 1987.

[3] V. Miller, "Uses of Elliptic Curves in Cryptography", *CRYPTO'85*, LNCS 218, pp. 417-426, 1986.

[4] J. Pollard, "Monte Carlo Methods for Index Computation Mod p", *Mathematics of Computation*, **32**, pages 918-924, 1978.

[5] P. van Oorschot and M. Wiener, "Parallel Collision Search with Cryptanalytic Applications", *Journal of Cryptology*, **12** (1999), 1-28.

[6] S. Pohlig and M. Hellman, "An Improved Algorithm for Computing Logarithms Over GF(p) and its Cryptographic Significance", *IEEE Transactions on Information Theory*, **24**, pages 106-110, 1978.

[7] Lawrence C. Washington. "Elliptic Curves". *Discrete Mathematics and its Applications*. Chapman & Hall/ CRC, Boca Raton, FL, 2003. Number Theory and Cryptography.

[8] Darrel Hankerson, Alfred Menezes, and Scott Vanstone. "Guide to Elliptic Curve Cryptography". *Springer Professional Computing*. Springer-Verlag, New York,2004.

[8] Michael J. Wiener and Robert J. Zuccherato. "Faster Attacks on Elliptic Curve Cryptosystem". *In Selected Areas in Cryptography*, **1556** of LNCS, pages 190-200, 1998.

[10] Paul C. van Oorchot and Michael J. Wiener. "Parallel Collision Search with Cryptanalytic Applications". *Jouranal of Cryptology*, **12(1)**, 1-28, 1999.

[11] Robert P. Gallant, Robert J. Lambert, and Scott A. Vanstone. "Improving the Parallelized Pollard Lambda Search on Anomalous Binary Curves". *Mathematics of Computation*, **69(232)**: 1699-1705, 2000.

[12] Certicom. "Certicom ECC Challenge. http://www.certicom.com/images/pdfs/cert_ecc_challenge.pdf,1997.

[13] NealKoblitz. "CM-curves with Good Crypto Graphic Properties". *In Advances in Cryptology* - Crypto1991, volume 576 of Lecture Notes in Comput. Sci., pages 279-287. Springer-Verlag, Berlin, 1992.