International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:8, 2015

Solving 94-bit ECDLP with 70 Computers in Parallel

Shunsuke Miyoshi, Yasuyuki Nogami, Takuya Kusaka, Nariyoshi Yamai

Abstract—Elliptic curve discrete logarithm problem(ECDLP) is one of problems on which the security of pairing-based cryptography is based. This paper considers Pollard's rho method to evaluate the security of ECDLP on Barreto-Naehrig(BN) curve that is an efficient pairing-friendly curve. Some techniques are proposed to make the rho method efficient. Especially, the group structure on BN curve, distinguished point method, and Montgomery trick are well-known techniques. This paper applies these techniques and shows its optimization. According to the experimental results for which a large-scale parallel system with MySQL is applied, 94-bit ECDLP was solved about 28 hours by parallelizing 71 computers.

Keywords—Pollard's rho method, BN curve, Montgomery multiplication.

I. Introduction

RECENT Ate-based pairings such as Optimal ate[1] and Xate[2] on Barreto-Naehrig(BN) curve[3] have received much attention since they realize quite efficient pairing calculations. However, few researchers have tried to evaluate the security of pairings. Elliptic curve discrete logarithm problem(ECDLP)[4] is one of problems on which the security of pairing-based cryptography is based. Pollard's rho method[5] is known as an efficient algorithm to solve a large-scale ECDLP. This paper optimizes Pollard's rho method to solve ECDLP on $\mathbb{G}_1$ over Barreto-Naehrig(BN) curve. $\mathbb{G}_1$ is a rational point group on BN curve defined over prime field from which an input for pairing is chosen. The rho method basically consists of two steps: randomly generating points on the curve and detecting a collision from the generated points.

Some techniques such as Montgomery multiplication, Montgomery trick, distinguished point method, and the group structure on BN curve have been proposed to make generating random points step more efficient. Montgomery trick reduces the number of inversions required in the calculation procedure. Distinguished point method reduces the load of a collision detection at server. The group structure on BN curve reduces the size of ECDLP itself. This paper optimizes Montgomery multiplication to make the arithmetic operations on BN curve more efficient.

When attacking large size ECDLPs, the system for detecting collisions is important. When solving 110-bit ECDLP, the rho method requires about 1600[GB] storage area and it needs to detect a collision from about 1.8×10^{16} points even if using the distinguished point method. Thus, this paper considers about a client-server model, and then the server detects a collision using MySQL.

Then, this paper attacks 94-bit ECDLP for evaluating the security. This experiment uses 69 clients, and 2 servers in our university. We used these computers during about 2 days and the system solved a 94-bit ECDLP.

S. Miyoshi, Y. Nogami and T. Kusaka are with Okayama University, 3-1-1, Tsushima-naka, Kita-ku, Okayama-shi, Okayama, 700-8530 Japan (e-mail: shunsuke.miyoshi@s.okayama-u.ac.jp).
N. Yamai is with Tokyo University of Agriculture and Technology, 2-24-16, Nakacho, Koganei, Tokyo, 184-8588 Japan.

II. Preliminaries

This section introduces elliptic curve over finite field and some properties of rational point on elliptic curve. In addition, this section introduces Pollard's rho method.

A. Elliptic Curve

In what follows, $\mathbb{F}_p$ denotes a prime field. An elliptic curve E is generally defined as follows.

$$E : y^2 = x^3 + ax + b, \qquad a, b \in \mathbb{F}_p,\ x, y \in \mathbb{F}_{p^m}. \tag{1}$$

$E(\mathbb{F}_p)$ that is a set of rational points on E over $\mathbb{F}_p$, including the $infinity\ point$ $\mathcal{O}$, forms an additive Abelian group. r denotes the group order of $E(\mathbb{F}_p)$.

1) Elliptic Curve Addition: Elliptic curve addition is the addition between rational points. $Q_1(x_1, y_1) + Q_2(x_2, y_2) = Q_3(x_3, y_3)$ is defined as follows, where Q_1, Q_2 and Q_3 are rational points of elliptic curve $E(\mathbb{F}_p)$.

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & if\ Q_1 \neq Q_2\ and\ x_1 \neq x_2, \\ \dfrac{3x_1^2 + a}{2y_1} & elseif\ Q_1 = Q_2\ and\ y_1 \neq 0, \\ \phi & otherwise, \end{cases} \tag{2}$$

$$\begin{pmatrix} x_3 \\ y_3 \end{pmatrix} = \begin{cases} \begin{pmatrix} \lambda^2 - x_1 - x_2 \\ (x_1 - x_3)\lambda - y_1 \end{pmatrix} & if \lambda \neq \phi, \\ \mathcal{O} & otherwise. \end{cases} \tag{3}$$

2) Barreto-Naehrig Curve: Barreto-Naehrig(BN) curve[3] that is well known to realize an efficient pairing is defined in the form of

$$E : y^2 = x^3 + b, \qquad b \in \mathbb{F}_p, \tag{4}$$

together with the following parameter settings,

$$p = 36\ell^4 - 36\ell^3 + 24\ell^2 - 6\ell + 1, \tag{5}$$
$$r = 36\ell^4 - 36\ell^3 + 18\ell^2 - 6\ell + 1, \tag{6}$$

where ℓ is a certain integer and p is the characteristic of $\mathbb{F}_p$. The embedding degree k of BN curve is 12.

B. Ate Pairing

Ate pairing e is defined as follows, where $E(\mathbb{F}_{p^k})[r]$, Ker, and π denote the set of rational points of order r, kernel of homomorphism, and Frobenius mapping[4], respectively.

$$\mathbb{G}_1 = E(\mathbb{F}_{p^k})[r] \cap \mathrm{Ker}(\pi - [1]), \tag{7}$$
$$\mathbb{G}_2 = E(\mathbb{F}_{p^k})[r] \cap \mathrm{Ker}(\pi - [p]), \tag{8}$$
$$e(\cdot,\cdot) : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3 = \mathbb{F}^*_{p^k}/(\mathbb{F}^*_{p^k})^r. \tag{9}$$

In the case of BN curve, the above $\mathbb{G}_1$ is equal to $E(\mathbb{F}_p)$.

C. Twist and Skew-Frobenius Mapping

In order to improve Ate pairing with BN curve, the sextic-twist technique is available. Since the embedding degree of BN curve is 12 and BN curve is written as Eq.(4), sextic-twist curve E' is given by

$$E' : \; y^2 = x^3 + bv^{-1}, \tag{10}$$

where v is a cubic and quadratic non residue in $\mathbb{F}_{p^2}$. In this case, we have the following isomorphism.

$$\mathbb{G}'_1 = E'(\mathbb{F}_{p^{12}})[r] \cap \mathrm{Ker}(\pi^2 - [p^2]), \tag{11}$$
$$\begin{aligned}\psi_6 : \; &(x, y) \in \mathbb{G}_1 \\ &\longmapsto (v^{1/3}x, v^{1/2}y) \in \mathbb{G}'_1.\end{aligned} \tag{12}$$

$\mathbb{G}'_1$ has the following automorphism $\tilde{\pi}$, where Q is a rational point on $\mathbb{G}_1$. It is called skew-Frobenius mapping.

$$\begin{aligned}\tilde{\pi}(Q) &= \psi_6^{-1}(\pi^2(\psi_6(Q))) \\ &= (v^{\frac{p^2-1}{3}}x, v^{\frac{p^2-1}{2}}y).\end{aligned} \tag{13}$$

In this case, $\tilde{\pi}^6(Q) = \tilde{\pi}$. Thus, in what follows, it is denoted by $\tilde{\pi}_6$. $\tilde{\pi}_6$ is used for the grouping in the rho method.

D. Security of Pairing-based Cryptography

The security of pairing-based cryptography based on discrete logarithm problem on $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_3$, and pairing inversion problem. Thus, pairing-based cryptography requires all of these difficulties. This paper evaluates the security of elliptic curve discrete logarithm problem on $\mathbb{G}_1$ for which this section introduces Pollard's rho method.

1) ECDLP: ECDLP is the problem that calculates the scalar s only by using rational points P and Q on E such that Q=$[s]P$, where $[s]P$ means

$$[s]P = \underbrace{P + P + P + \cdots}_{s \text{ points}}. \tag{14}$$

2) Pollard's Rho Method: Pollard's rho method is known as an efficient technique for solving an ECDLP. The rho method consists of two steps. The first step randomly generates a lot of rational points as follows, where a_i, b_i are random numbers from 0 to $r - 1$.

$$T_i = [a_i]P + [b_i]Q. \tag{15}$$

The next step detects a collision among the generated points. When a collision is found such as $T_i = T_j (i \neq j)$, the ECDLP is solved by a simultaneous equation.

The original algorithm of rho method is given as **Alg**.1. In this paper, we precompute a table which consists of n rational points such as Eq.(15) and generate random points by using the table. In what follows, $\eta(T_i)$ is a function that decides the index of a rational point in the table corresponding to the input T_i. It is said that a collision occurs when $\sqrt{\pi r/2}$ points are generated according to the birthday paradox.

Algorithm 1: Pollard's Rho Method

Input: P, $Q(= [s]P) \in E(\mathbb{F}_p)$ $(0 \leq s < r)$
Output: s

```
for i = 0 to n − 1 do
    a_i, b_i are random elements (0 ≤ a_i, b_i < r),
    T_i ←[a_i]P + [b_i]Q.
a_n, b_n are random elements (0 ≤ a_n, b_n < r),
T_n ← [a_n]P + [b_n]Q.
for i = n + 1 to r − 1 do
    l ← η(T_{i−1}),
    a_i ← a_{i−1} + a_l, b_i ← b_{i−1} + b_l, T_i ← T_{i−1} + T_l,
    if T_i = T_j (0 ≤ j ≤ i) then
        go out this loop.
s ← −(a_i − a_j)/(b_i − b_j) (mod r).
```

III. Improvement of Rho Method

This section denotes some techniques for making the attack more efficient.

A. Using The Group Structure on BN Curve

The skew-Frobenius mapping $\tilde{\pi}_6$ is available for the grouping technique. In $\mathbb{G}_1$ of BN curve, suppose that the rho method detected a collision by T_i and T_j as $T_i = \tilde{\pi}_6^n(T_j)$ $(0 \leq n < 6)$. Because, from the Eq.(8), $\tilde{\pi}_6^n(T_j) = [p^{2n}]T_j$. Thus, this technique enables to reduce the average number for a collision from $\sqrt{\pi r/2}$ to $\sqrt{\pi r/12}$. In detail,

$$\begin{aligned} T_i &= \tilde{\pi}_6^n(T_j), \\ T_i &= [p^{2n}]T_j, \\ [a_i]P + [b_i]Q &= [p^{2n}]([a_j]P + [b_j]Q), \\ a_i + b_i \cdot s &\equiv p^{2n} \cdot a_j + p^{2n} \cdot b_j \cdot s \pmod r, \\ s &\equiv -\frac{(a_i - p^{2n} \cdot a_j)}{(b_i - p^{2n} \cdot b_j)} \pmod r. \end{aligned} \tag{16}$$

$\tilde{\pi}_6^n(T_j)(0 \leq n < 6)$ are thus conjugates to each other that helps efficiently solving ECDLPs since they are connected by not only efficient mapping $\tilde{\pi}_6^n$ but also scalar multiplications $[p^{2n}]$. In detail, they are given as follows, where $\epsilon = v^{\frac{p^2-1}{3}}$,

$$T_j = (x_j, y_j), \tag{17}$$
$$\tilde{\pi}_6(T_j) = (\epsilon x_j, y_j), \tag{18}$$
$$\tilde{\pi}_6^2(T_j) = (\epsilon^2 x_j, y_j), \tag{19}$$
$$\tilde{\pi}_6^3(T_j) = (x_j, -y_j), \tag{20}$$
$$\tilde{\pi}_6^4(T_j) = (\epsilon x_j, -y_j), \tag{21}$$
$$\tilde{\pi}_6^5(T_j) = (\epsilon^2 x_j, -y_j). \tag{22}$$

In order to apply this grouping technique to the rho method, a function $L(\cdot)$ that determines a representative of the conjugate points is required. Then, a random walk is slightly changed to $T_i = L(T_{i-1}) + T_l$, where $l = \eta(L(T_{i-1}))$. In this procedure, this technique causes a fruitless cycle[7]. When a fruitless cycle occurs, this paper escapes the cycle by adding another rational point.

B. Montgomery Trick

The rho method calculates a lot of elliptic curve additions for the random walk step. An elliptic curve addition costs $6A + 3M + I$, where A, M, and I denote calculation costs of addition, multiplication, and inversion over $\mathbb{F}_p$, respectively. When attacking a large size ECDLP, I is much larger than M, i.e., $I > 5M$ in the case that p is a 94 bits prime. Montgomery trick[6] is a technique to reduce the number of inversions for which many elliptic curve additions are preferred to be calculated in parallel. When using Montgomery trick, an elliptic curve addition cost is $6A + 6M + I/cn$, where cn is the number of elliptic curve additions calculated in parallel.

C. Distinguished Point Method

The original rho method averagely needs to store $\sqrt{\pi r/2}$ points, and detect a collision among the generated points. Distinguished point method is proposed to reduce the number of storing points and the time for a collision detection. This method stores only the $distinguished\ points$. In this paper, a distinguished point means that its x-coordinate is divisible by a number parameter θ.

D. Parallelization

Suppose that a collision $T_i = U_j$ was detected. Then, their following points similarly collide as follows,

$$T_{\eta(T_i)} = U_{\eta(U_j)},\ T_{\eta(T_{\eta(T_i)})} = U_{\eta(U_{\eta(U_j)})},\ \dots \tag{23}$$

where η is used for uniquely determining a certain point in the random walk table. Thus, it is found that a parallelization is effective for the rho method.

IV. Optimization

This section shows an optimization for the rho method. The target of this paper is about 100-bit ECDLP and this paper uses 64-bit word length computer. The compiler is gcc version 4.5.4, this version supports 128-bit integer. Thus, an addition in $\mathbb{F}_p$ is easily implemented when using 128-bit integer, however a multiplication and inversion in $\mathbb{F}_p$ require additional ideas.

A. Montgomery Multiplication

This paper applies Montgomery multiplication for a multiplication in $\mathbb{F}_p$. Montgomery multiplication calculates a multiplication in $\mathbb{F}_p$ using only additions and bit shift operations. The algorithm is shown as **Alg**.2,

Algorithm 2: Montgomery Multiplication

Input: $X = (x_{m-1} \cdot 2^{w\cdot(m-1)}, ..., x_1 \cdot 2^{w\cdot 1}, x_0 \cdot 2^{w\cdot 0})$,
$Y = (y_{m-1} \cdot 2^{w\cdot(m-1)}, ..., y_1 \cdot 2^{w\cdot 1}, y_0 \cdot 2^{w\cdot 0})$,
$N = (n_{m-1} \cdot 2^{w\cdot(m-1)}, ..., n_1 \cdot 2^{w\cdot 1}, n_0 \cdot 2^{w\cdot 0})$,
$W = -N^{-1} \bmod 2^w$
Output: $Z = XY2^{-m\cdot w} (\bmod N)$
1 $Z = 0$
2 **for** $i = 0$ **to** $m-1$ **do**
3   $C = 0$
4   $t_i = (z_0 + x_i y_0)W \bmod 2^w$
5   **for** $j = 0$ **to** $m-1$ **do**
6     $Q = z_j + x_i y_j + t_i n_j + C$
7   **if** $j \neq 0$ **then**
      $z_{j-1} = Q \bmod 2^w$
8     $C = Q/2^w$
9   $z_{m-1} = C$
10 **if** $Z \geq N$ **then**
  $Z = Z - N$

This paper considers the optimization of **Alg**.2. j at line 5 equals to 0 or 1. When $j = 0$, it is same value of "$z_0 + x_i y_0$" at line 4 and "$z_j + x_i y_j$" at line 6. However, Q at line 6 maybe 129-bit. This paper deals with this carry by comparison. In line 6, store $x_i y_j$ first. Next, calculate Q and compare with the stored $x_i y_j$. If $Q < x_i y_j$, a carry occurs. The optimized Montgomery multiplication algorithm is shown in **Alg**.3.

Algorithm 3: Optimized Montgomery Multiplication

Input: $X = (x_1 \cdot 2^{64}, x_0)$, $Y = (y_1 \cdot 2^{64}, y_0)$,
$N = (n_1 \cdot 2^{64}, n_0)$, $W = -N^{-1} \bmod 2^{64}$
Output: $Z = XY2^{-128} (\bmod N)$
$xy = x_0 \times y_0$
$t = xy \times W \bmod 2^{64}$
$Q = xy + (t \times n_0)$
$carry = Q < xy$
$Q = x_0 \times y_1 + (t \times n_1) + Q/2^{64}$
if $carry$ **then**
  $Q = Q + 2^{64}$
$Z = Q/2^{64}$
$xy = x_1 \times y_0 + Q \bmod 2^{64}$
$t = xy \times W \bmod 2^{64}$
$Q = xy + (t \times n_0)$
$carry = Q < xy$
$Z = Z + x_1 \times y_1 + t \times n_1 + Q/2^{64}$
if $carry$ **then**
  $Z = Z + 2^{64}$
if $Z \geq N$ **then**
  $Z = Z - N$

B. Collision Detection at Server

This paper considers about a model such that many clients generate random rational points by the optimized rho method in parallel and then send the generated and distinguished points to the server, and the server detects a collision.

There are a few rational points on the server when using the technique described in **Sec**.III-C, however it is difficult that

International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:8, 2015

all points are stored on memory. For example, when solving 110-bit ECDLP and setting the parameter θ by 2^{20}, about $\sqrt{\pi \times 2^{110}/12}/2^{20} \approx 1.8 \times 10^{16}$ points need to be stored. If the size of 1 data is 100[Byte], it requires 1600[GB] storage area.

Therefore this paper stores rational points on the storage such as HDD (Hard Disk Drive). When rational points are stored in HDD, the number of storing points are increased, however, the performance degradation is caused by disk IO. This paper avoids the performance degradation by controlling the size of θ in **Sec**.III-C.

V. Experiment

This paper implemented the optimized rho method described in **Chap**.III and **Chap**.IV and evaluated the security of ECDLP. In this experiment, the target is 94-bit ECDLP, where r = 9401882419968856913336171017(94-bit). Then average number for detecting a collision is $\sqrt{\pi r/12} \approx 49599992270415$ according to the birthday paradox on the technique described in **Sec**.III-A. This paper tried to find the scalar s for the following P and $Q(=[s]P)$.

$$P = (7262408195386367430506168279, \\ 5909492387210969059012426224) \quad (24)$$

$$Q = (8830658031211912159902020265, \\ 7909569409614716533720230088) \quad (25)$$

Table I shows the computational environment. **Table** II shows the result. The scalar value s was 4960155460405181786913975321 from solving result. The number of generated points in **Table** II is given by multiplying the number of stored points and θ for the distinguished points. According to the result, 94-bit ECDLP was solved in 100779[sec] by generating about 42069934473216 points. Since, the average number for a collision is $\sqrt{\pi r/12} \approx 49599992270415$, it is estimated that 94-bit ECDLP is averagely solved by 100779[sec] $\times \frac{49599992270415}{42069934473216} \approx$ 118818[sec].

TABLE I
Computational Environment

Client	The number of computers	69
	OS	Windows7 Professional (64-bit)
	CPU	Intel Core 2 Duo (3.06GHz)
Server	The number of computers	2
	OS	CentOS 6.5 (64–bit)
	CPU	Intel Core 2 Duo (2.80GHz) Intel Core i5-4670K (3.40GHz)
	Database	MySQL ver. 5.1.73

TABLE II
Result

The number of stored points	10030254
θ for distinguished point	2^{22}
The number of generated points	42069934473216
Time for solving an ECDLP[sec]	100779
Average time for solving an ECDLP[sec]	118818

VI. Conclusion

This paper has evaluated the security of $\mathbb{G}_1$ in pairing-based cryptography on BN 94-bit curve by optimized the rho method. The results of the experiment of a 94-bit ECDLP show that the problem was solved in 28 hours with 71 computers. Since the number of the generated points in the experiment is 42069934473216, the number is much smaller than the estimated average number of the rho method.

References

[1] F. Vercauteren, “Optimal pairings,” IEEE Transactions on Information Theory, Vol. 56, No.1, pp. 455-461, 2010.
[2] Y. Nogami, Y. Sakemi, H. Kato, M. Akane, and Y.Morikawa, “Integer Variable χ-based Cross Twisted Ate Pairing and Its Optimization for Barreto-Naehrig Curve,” IEICE Transactions on Fundamentals of Electronics, IEICE Transactions, Vol. 2009, No. 8, pp. 1859-1867, 2009.
[3] P. S. L. M. Barreto and M. Naehrig,“Pairing–Friendly Elliptic Curves of Prime Order,” SAC 2005, LNCS, vol. 3897, pp. 319–331, 2006.
[4] H. Cohen and G. Frey ed., “Handbook of Elliptic and Hyperelliptic Curve Cryptography, ” Chapman & Hall, 2006.
[5] J. Pollard, “Monte Carlo Methods for Index Computation (mod p),” Math. Comp, vol. 32, pp. 918–924, 1978.
[6] P. L. Montgomery, “Speeding the Pollard and Elliptic Curve Methods of Factorization,” MATH. OF COMPUT., vol. 48, no. 177, pp. 243–264, 1987.
[7] Joppe W. Bos, Craig Costello, and Andrea Miele, “Elliptic and Hyperelliptic Curves: a Practical Security Analysis,” PKC 2014, vol. 8383, pp. 203–220, 2014