Journal of Math-for-Industry, Vol. 3 (2011B-3), pp. 107–112

Math-for-industry
Education & Research Hub

On random walks of Pollard's rho method for the ECDLP on Koblitz curves

Masaya Yasuda, Tetsuya Izu, Takeshi Shimoyama and Jun Kogure

Received on August 24, 2011

Abstract. Pollard's rho method is the asymptotically fastest known attack for the elliptic curve discrete logarithm problem (ECDLP) except special cases. It works by giving a pseudo-random sequence defined by an iteration function and then detecting a collision in the sequence. We note that the number of iterations before obtaining a collision is significant for the running time of the rho method and depends on the choice of an iteration function. For many iteration functions suitable for the ECDLP on elliptic curves except Koblitz curves, the number of iterations before obtaining a collision had been investigated. In this paper, we propose a new iteration function on Koblitz curves which is an extension of the iteration function proposed by Gallant et al. and analyze the performance on our iteration function experimentally.

Keywords. Pollard's rho method, ECDLP, Koblitz curves, Frobenius map

1. Introduction

In 1985, Neal Koblitz and Victor Miller independently proposed using elliptic curves to design public-key cryptographic systems (see [4, 9, 12]). The hardness of the elliptic curve discrete logarithm problem (ECDLP) is essential for the security of all elliptic curve cryptographic schemes. The ECDLP is as follows: given an elliptic curve E defined over a finite field $\mathbb{F}_q$, a point $S \in E(\mathbb{F}_q)$ of order n, and a point $T \in \langle S \rangle$, find the integer $k \in [0, n-1]$ with $T = kS$. Although a number of ways to solve the ECDLP is known, Pollard's rho method [14] is the asymptotically fastest known attack for the ECDLP except special cases including the supersingular cases and the anomalous cases (see [9, 13, 15, 16, 18]).

In the rho method, an *iteration function* $f : \langle S \rangle \to \langle S \rangle$ is used to define a sequence $\{X_i\}$ by $X_{i+1} = f(X_i)$ for $i = 0, 1, 2, \ldots$ with a starting point X_0. Furthermore, f should have the characteristic of a random function. Since the set $\langle S \rangle$ is finite, the sequence will eventually meet a point that has occurred before, which is called a *collision*, and then cycle forever. Since a collision gives the solution of the ECDLP with high probability, the number of iterations before obtaining a collision is significant for the running time of the rho method. Improved by Wiener and Zuccherato [22], van Oorschot and Wiener [21], and Gallant, Lambert and Vanstone [8], the rho method can be efficiently parallelized and can be sped up using group automorphisms. In particular, the rho method can be sped up by a factor of $\sqrt{2m}$ for Koblitz curves over $\mathbb{F}_{2^m}$ using the Frobenius map and the Negation map. By the birthday paradox, the expected number of iterations on Koblitz curves before obtaining a collision is approximately $\frac{1}{2}\sqrt{\pi n/m}$ (see [9] for details). However, since the rho method is a probabilistic algorithm, the number of iterations before obtaining a collision is dependent on the choice of an iteration function f and a starting point X_0.

In this paper, we propose a new iteration function on Koblitz curves which is an extension of the iteration function proposed by Gallant, Lambert and Vanstone [8], and analyze the performance of our iteration function experimentally. To analyze the performance of an iteration function f on elliptic curves, we consider the value

$$\delta(f) := \begin{pmatrix}\text{The number of iterations } f\\ \text{before obtaining a collision}\end{pmatrix} \Big/ \text{Exp},$$

where 'Exp' is the expected number of iterations before obtaining a collision. For many iteration functions f suitable for solving the ECDLP on elliptic curves except Koblitz curves, the average value of $\delta(f)$ was investigated by Teske [19, 20], and Bai and Brent [1]. By many experiments of solving the ECDLP on Koblitz curves of relatively short parameters, we estimate the average value of $\delta(f)$ with our iteration function f. In particular, we estimate the average value of $\delta(f)$ with the iteration function f proposed by Gallant, Lambert, and Vanstone [8] experimentally.

2. Review on Pollard's rho method for the ECDLP

To fix our notation, we here review on the rho method for the ECDLP due to [9].

2.1. The basic idea of the rho method for the ECDLP

Definition 1. The elliptic curve discrete logarithm problem (ECDLP) is as follows: given an elliptic curve E defined over a finite field $\mathbb{F}_q$, a point $S \in E(\mathbb{F}_q)$ of order n, and a point $T \in \langle S \rangle$, find the integer $k \in [0, n-1]$ with $T = kS$.

We define an *iteration function* $f : \langle S \rangle \to \langle S \rangle$ such that it is easy to compute $X' = f(X)$ and $c', d' \in [0, n-1]$ with $X' = c'S + d'T$ for given $X = cS + dT$. For a starting point $X_0 = c_0 S + d_0 T$ with randomly chosen $c_0, d_0 \in [0, n-1]$, we define a sequence $\{X_i\}$ by $X_{i+1} = f(X_i)$ for $i \geq 0$. It follows from the definition of iteration functions that we can compute $c_i, d_i \in [0, n-1]$ with $X_i = c_i S + d_i T$. Since the set $\langle S \rangle$ is finite, the sequence will eventually meet a point that has occurred before, which is called a *collision*, and then cycle forever. A collision $X_i = X_j$ with $i \neq j$ gives the equation

$$c_i S + d_i T = c_j S + d_j T.$$

Since we have $(c_i - c_j)S = (d_j - d_i)T = (d_j - d_i)kS$, we can compute the solution

$$k = (c_i - c_j) \cdot (d_j - d_i)^{-1} \bmod n$$

of the ECDLP if $d_j - d_i \in (\mathbb{Z}/n\mathbb{Z})^*$. This is the idea of the rho method for the ECDLP (see [9, pp. 157–158] for details).

For solving the ECDLP efficiently, an iteration function f should have the characteristic of a random function, and the expected number of iterations before obtaining a collision is approximately $\sqrt{\pi n/2} \approx 1.2533\sqrt{n}$ by the birthday paradox if f is a random function [9, p. 157]. However, in fact, the number of iterations before obtaining a collision is heavily dependent on the choice of an iteration function.

Remark 1. A typical iteration function is as follows: Let $\{H_1, H_2, \cdots, H_L\}$ be a random partition of the set $\langle S \rangle$ into L sets of roughly the same size. We write $H(X) = j$ if $X \in H_j$ and call H the *partition function*. For $a_j, b_j \in_R [0, n-1]$ for $1 \leq j \leq L$, set $M_j = a_j S + b_j T \in \langle S \rangle$. Then we can give an iteration function $f : \langle S \rangle \to \langle S \rangle$ defined by

$$f(X) = X + M_j, \text{ where } j = H(X).$$

For given $X = cS + dT$, we can compute $X' = f(X) = c'S + d'T$ with $c' = c + a_j \bmod n$ and $d' = d + b_j \bmod n$. This iteration function is called an *L-adding walk* proposed by Teske (see [1, 19, 20]). For many iteration functions suitable for the ECDLP on elliptic curves except Koblitz curves, the number of iterations before obtaining a collision was investigated by Teske [19, 20], and Bai and Brent [1]. For example, we give Table 1: All the data in Table 1 denote the average number of iterations before obtaining a collision by solving the ECDLP on prime fields of 5–13 digits.

Table 1: Performance of iteration functions on elliptic curves on prime fields

Iteration functions	f_P	f_{PG}	$f_{TA[20]}$	$f_{TM[16:4]}$
(average of iterations)/$\sqrt{n}$	1.60	1.62	1.29	1.30
(average of iterations)/Exp	1.28	1.29	1.03	1.04

Exp $= \sqrt{\pi n/2}$ (in this case), f_P: Pollard's original iteration function [14], f_{PG}: Pollard's iteration function generalized by Teske, $f_{TA[20]}$: Teske's L-adding walk with $L = 20$ explained in Remark, and $f_{TM[16:4]}$: Teske's mixed-walk with 16 multipliers and 4 squarings (see [19] for details)

2.2. Improving Pollard's rho method

2.2.1. Parallelized Pollard's rho method:

Van Oorshot and Wiener [21] proposed a variant of Pollard's rho method that yields a factor M speed up when M processors are employed. The idea is to allow the sequences $\{X_i\}$ generated by the processors to collide with one another. More precisely, each processor randomly selects its own starting point X_0, but all processors use the same iteration function f to compute subsequent points X_i.

2.2.2. Collision detection:

Floyd's cycle-finding algorithm [11] finds a collision in the sequence generated by a single processor. The following strategy enables efficient finding of a collision in the sequences generated by different processors. An easy testable *distinguishing property* of points is selected. For example, a point may be *distinguished* if the leading t bits of its x-coordinate are zero. Let $0 < \theta < 1$ be the proportion of points in the set $\langle S \rangle$ having this distinguishing property. Whenever a processor encounters a distinguished point, it transmits the point to a central server which store it in a sorted list. When the server receives the same distinguished point for the second time, it computes the desired logarithm and terminates all processors. The expected number of iterations per processor before obtaining a collision is $(\sqrt{\pi n/2})/M$, when M processors are employed. A subsequent distinguished point is expected after $1/\theta$ iterations. Hence the expected number of elliptic curve operations performed by each processor before observing a collision of distinguished points is

$$\frac{1}{M}\sqrt{\frac{\pi n}{2}} + \frac{1}{\theta}.$$

We note that the running time of $1/\theta$ iterations after a collision occurs is negligible for the total running time if we select θ such that $1/\theta$ is small enough compared to the order n of the point S.

3. Speeding Pollard's rho method for Koblitz curves

Koblitz curves were first suggested for use in cryptography by Koblitz [12]. The defining equation for a Koblitz curve

E is

$$y^2 + xy = x^3 + ax^2 + b,$$

where $a, b \in \mathbb{F}_2$ with $b \neq 0$. The *Frobenius map* $\phi : E(\mathbb{F}_{2^m}) \to E(\mathbb{F}_{2^m})$ is defined by

$$\phi : (x, y) \mapsto (x^2, y^2) \text{ and } \phi : \mathcal{O} \mapsto \mathcal{O},$$

where $\mathcal{O}$ is the point of infinity. We note that the Frobenius map is a group homomorphism of order m and can be efficiently computed since squaring in $\mathbb{F}_{2^m}$ is relatively inexpensive (see [9] for details). Using the Frobenius map and the Negation map, the rho method for the ECDLP on Koblitz curves E is sped up. The relation $\sim$ on the set $\langle S \rangle$ defined by

$$P \sim Q \text{ if and only if } P = \pm\phi^j(Q) \text{ for some } j \in [0, m-1]$$

is an equivalence relation. We denote the set of equivalence classes by $E/\sim$, and let $[P]$ denote the equivalence class containing a point P. The idea behind the speedup is to modify an iteration function on E so that it is defined on $E/\sim$. Since most equivalence classes have size $2m$, then the collision search space has size approximately $n/2m$. Hence the expected running time of the rho method accelerated by Frobenius map is

$$\frac{1}{2}\sqrt{\frac{\pi n}{m}},$$

which is a speed up by a factor of $\sqrt{2m}$.

3.1. A typical iteration function on Koblitz curves

Gallant, Lambert and Vanstone in [8] proposed an iteration function suitable for the rho method with speedup by the Frobenius map as follows. We define an iteration function on E

$$g : R \to R + \phi^t(R), \text{ where } t = \mathrm{hash}_m(\mathcal{L}(R)),$$

where hash_m is a conventional hash function (in the computer science) having range $[0, m-1]$ and $\mathcal{L}$ is a *labelling function* from the equivalence classes in $E/\sim$ to some set of representatives. For example, the labelling function $\mathcal{L}$ takes the lexicographically least x-coordinate of the elements of the equivalent class. We note that computing $R + \phi^t(R)$ is about the same work as a point addition on E since computing $\phi^t(R)$ is reasonably easy. Using the iteration function g, we give an iteration function f on $E/\sim$ defined by

$$f([R]) = [g(R)], \text{ for } R \in E,$$

which is the iteration function proposed by Gallant, Lambert and Vanstone [8]. We note that the iteration function f is a well-defined map on $E/\sim$.

Remark 2. For solving the ECC2K-130 challenge problem [5, 6], Bailey et al. in [3] proposed an iteration function on $E/\sim$ as follows:

$$f([R]) = [g(R)],\ g(R) = R + \phi^t(R),$$
$$t = ((\mathrm{HW}(x_R)/2 \bmod 8) + 3),$$

where $\mathrm{HW}(x_R)$ is the Hamming weight of the x-coordinate x_R of a point $R \in E$. Compared to the iteration function proposed by Gallant, Lambert, and Vanstone, this iteration function may reduce the randomness of the walk in the rho sequence since the index t satisfies $3 \leq t \leq 10$ (see [3] for details). We note that this iteration function has an advantage of the computational speed.

3.2. Our iteration function

For $0 \leq s \leq m$, we define an iteration function on E given by

$$g_s(R) = \begin{cases} 2R & \text{if } 0 \leq t < s, \\ R + \phi^t(R) & \text{otherwise,} \end{cases}$$

where $t = \mathrm{hash}_m(\mathcal{L}(R)) \in [0, m-1]$. Our iteration function f_s on $E/\sim$ is defined by $f_s([R]) = [g_s(R)]$. Clearly, our iteration function f_s with $s = 0$ is the same as the iteration function proposed by Gallant, Lambert and Vanstone. Hence our iteration function is an extension of the iteration function proposed by Gallant, Lambert and Vanstone, based on the Teske's idea [19]. We note that our iteration function is also a well-defined map on $E/\sim$. The cost $t(g_s)$ of computing g_s is

$$t(g_s) = \frac{s}{m} \cdot S + \frac{m-s}{m} \cdot M,$$

where M is the cost of a point addition on E and S is the cost of a point doubling on E, if we neglect the cost of computing $\phi^t(R)$. Since we must compute g_s and $\mathcal{L}$ in computing f_s, we see that the cost $t(f_s)$ of computing f_s is equal to $t(g_s) + t(\mathcal{L})$ where $t(\mathcal{L})$ is the cost of computing $\mathcal{L}$. In the case of $s = 0$, Gallant, Lambert and Vanstone in [8, p. 1702] estimate $t(\mathcal{L}) = 0.2 \cdot t(g_0)$ and hence $t(f_0) = 1.2 \cdot t(g_0)$. For the further comparison in this paper, we choose $t(f_s) = 1.2 \cdot t(g_s)$ for any s, which is the highest cost estimation for our proposed scheme. Therefore we can compute g_s and f_s with high speed as the parameter s become large if a point doubling is faster than a point addition on E.

4. Experimental investigation

In this section, we analyze the performance on our iteration function f_s by many experiments of solving the ECDLP on Koblitz curves of relatively small parameters.

4.1. Description of experiments

To analyze the performance on our iteration function f_s, we solved the ECDLP on Koblitz curves under the following conditions and collected the data of the number of iterations before obtaining a collision:

- We used parallelized Pollard's rho method with $M = 10$ processors, collision detection using distinguished points and our iteration function f_s for $s = 0, m/5, m/3, m/2$.

- For each parameter of the ECDLP on Koblitz curves and each iteration function, we solved the ECDLP for 100 times with randomly chosen starting points.
- The solving ECDLP parameters are denoted by ECC2K-41, ECC2K-53, ECC2K-83 and ECC2K-89 (see below).

4.2. Parameters of the ECDLP on Koblitz curves

We describe the parameters of the ECDLP on Koblitz curves as follows:

- Description of each parameter
  - m: the order of the finite field is 2^m.
  - a, b: the field elements in $\mathbb{F}_2$ which define the elliptic curve $E : y^2 + xy = x^3 + ax^2 + b$.
  - f: the reduction polynomial which defines the polynomial basis representation of $\mathbb{F}_{2^m}$ (f is represented in hexadecimal).
  - n: the order of the point S; n is a prime number.
  - h: the co-factor (the number of points in $E(\mathbb{F}_{2^m})$ divided by n).
  - x_S, y_S: the x- and y- coordinates of the point S.
  - x_T, y_T: the x- and y- coordinates of the point T.
- List of the parameters of the ECDLP on Koblitz curves
  - ECC2K-41

$$\begin{cases} m &= 41,\ a = 0,\ b = 1 \\ f &= \texttt{20000000009} \\ n &= \texttt{800008CE1F}\ (\approx 39\text{bit}) \\ h &= 4 \\ x_S &= \texttt{11CA4FC0912} \\ y_S &= \texttt{14C103D4BF1} \\ x_T &= \texttt{173AE716000} \\ y_T &= \texttt{1F8511BA580} \end{cases}$$

  - ECC2K-53

$$\begin{cases} m &= 53,\ a = 0,\ b = 1 \\ f &= \texttt{20000000000047} \\ n &= \texttt{1323E34C2FD1}\ (\approx 44\text{bit}) \\ h &= \texttt{1AC} \\ x_S &= \texttt{D1372FE8A9D59} \\ y_S &= \texttt{817E6D0A220B9} \\ x_T &= \texttt{AEEEFC38145B6} \\ y_T &= \texttt{1AE8D2DD5175D6} \end{cases}$$

  - ECC2K-83

$$\begin{cases} m &= 83,\ a = 1,\ b = 1 \\ f &= \texttt{80000000200000000007} \\ n &= \texttt{1E722CD0C26963}\ (\approx 53\text{bit}) \\ h &= \texttt{434442CA} \\ x_S &= \texttt{14B4C858FD773EFE30A41} \\ y_S &= \texttt{22F9797E1981BD7750CBC} \\ x_T &= \texttt{5374B930E6A5A75E910C9} \\ y_T &= \texttt{2E7F26B3A696E98555167} \end{cases}$$

  - ECC2K-89

$$\begin{cases} m &= 89,\ a = 0,\ b = 1 \\ f &= \texttt{200000000000004000000001} \\ n &= \texttt{61D035AC14F7357}\ (\approx 58\text{bit}) \\ h &= \texttt{53C05A24} \\ x_S &= \texttt{E757BC1D481ED8AAE2F53F} \\ y_S &= \texttt{1C4EDA2EA43480E1820958} \\ x_T &= \texttt{5BF5E00A5CF3D704FEDAF3} \\ y_T &= \texttt{F6DA9E382DD26B77E43856} \end{cases}$$

Remark 3. The Koblitz curve used in cryptography has a cofactor $h = 2, 4$. Since the defining equation for a Koblitz curve is special, there exist few Koblitz curves with $h = 2, 4$ for $m < 100$. We here choose Koblitz curves with small cofactor h as the parameters of the ECDLP for Koblitz curves. Moreover, we used parallelized Pollard's rho method with collision detection using distinguished points having $1/\theta$ small enough compared with the order n of the point S, where θ is the proportion of points in $\langle S \rangle$ having the distinguishing property ($1/\theta : 8 \sim 12$bit). Therefore $1/\theta$ iterations after a collision occurs is negligible.

4.3. Our experimental results

In Table 2, we summarize the performance of our iteration function f_s for $s = 0, m/5, m/3, m/2$ on each parameter of the ECDLP on Koblitz curves. In Table 2, we give the following data:

- μ: the average number of iterations before a collision with f_s.
- $\delta(f_s)$: the value given by μ/Exp, where $\mathrm{Exp} = \frac{1}{2}\sqrt{\pi n/m}$ is the expected number of iterations before obtaining a collision (see §3).
- σ/Exp: standard deviation σ divided by Exp.

In Table 3, we summarize the average value of $\delta(f_s)$ from Table 2. From Table 3, we can see the followings in our experiments:

- For a random function f, we have $\delta(f) = 1$. Since the average value of $\delta(f_s)$ with $s = 0$ is 1.05, the iteration function proposed by Gallant, Lambert and Vanstone has a performance almost equal to a random function on $E/\sim$.
- The average number of iterations f_s before obtaining a collision increases as the parameter s become large. Therefore it follows from §3.2 that there is a relation of trade-off between the computational speed of f_s and the number of iterations before obtaining a collision with f_s, if a point doubling is faster than a point addition on E.
- From our assumption $t(f_s) = 1.2 \times t(g_s)$ in §3.2, the average running time of the rho method with our iteration function f_s is upper-bounded by

$$\Delta(s) = (\text{average value of } \delta(f_s)) \times t(f_s).$$

Table 2: Performance of our iteration function on ECC2K-41, 53, 83, 89

Parameter of the ECDLP	Iteration function f_s	Av. of the number of iterations (μ)	Av. value for $\delta(f_s)$ (μ/Exp)	St. deviation (σ/Exp)
ECC 2K-41	$s = 0$	109789	1.06	0.52
	$s = m/5$	117708	1.14	0.64
	$s = m/3$	133100	1.29	0.64
	$s = m/2$	120467	1.17	0.62
ECC2K-53	$s = 0$	616273	1.10	0.56
	$s = m/5$	540220	0.96	0.50
	$s = m/3$	628533	1.12	0.59
	$s = m/2$	706519	1.26	0.65
ECC2K-83	$s = 0$	9362605	1.03	0.49
	$s = m/5$	8979741	0.99	0.56
	$s = m/3$	11278465	1.25	0.59
	$s = m/2$	10531561	1.16	0.63
ECC2K-89	$s = 0$	64270064	1.01	0.48
	$s = m/5$	74819712	1.20	0.62
	$s = m/3$	67569759	1.08	0.58
	$s = m/2$	80869338	1.29	0.60

Exp $= \frac{1}{2}\sqrt{\frac{\pi n}{m}} = 102620$ (ECC2K-41), 558438 (ECC2K-53), 905047 (ECC2K-83), 62348200 (ECC2K-89).

Table 3: Average value of $\delta(f_s)$ for $s = 0, m/5, m/3, m/2$ on ECC2K-41, 53, 83, 89.

Parameter of the ECDLP	Iteration function f_s			
	$s = 0$	$s = m/5$	$s = m/3$	$s = m/2$
ECC2K-41	1.06	1.14	1.29	1.17
ECC2K-53	1.10	0.96	1.12	1.26
ECC2K-83	1.03	0.99	1.25	1.16
ECC2K-89	1.01	1.20	1.08	1.29
average	1.05	1.07	1.18	1.22

In the case $S = 0.8M$, we have

$$\Delta(s) = \begin{cases} 1.05 \cdot 1.2M \approx 1.26M & (s = 0) \\ 1.07 \cdot 1.15M \approx 1.23M & (s = m/5) \\ 1.18 \cdot 1.12M \approx 1.32M & (s = m/3) \\ 1.22 \cdot 1.08M \approx 1.32M & (s = m/5) \end{cases}$$

from Table 3 and §3.2. Therefore we see that our iteration function f_s with $s = m/5$ is the most suitable for solving the ECDLP on Koblitz curves in the case $S = 0.8M$ on average.

5. Conclusion

Gallant, Lambert and Vanstone in [8] proposed an iteration function suitable for solving the ECDLP on Koblitz curves. In this paper, we proposed a new iteration function f_s on Koblitz curves which is an extension of their iteration function based on the Teske's idea [19] and analyzed the performance of our iteration function experimentally. By many experiments of solving the ECDLP on Koblitz curves of relatively small parameters (ECC2K-41, ECC2K-53, ECC2K-83, ECC2K-89), we obtained the average value of the number of iterations before obtaining a collision in Table 3. We showed the followings in our experiments: The iteration function proposed by Gallant, Lambert and Vanstone has a performance almost equal to a random function on $E/\sim$. Furthermore, in the case $S = 0.8M$, our iteration function f_s with $s = m/5$ is more suitable for solving the ECDLP on Koblitz curves than the iteration function proposed by Gallant, Lambert and Vanstone.

References

[1] S. Bai and R. P. Brent, "On the efficiency of Pollard's rho method for discrete logarithms", In Proceedings of CATS '2008, pp. 125–131, (2008).

[2] D. V. Bailey et. al., "The Certicom Challenges ECC2-X", available at `http://binary.cr.yp.to/ecc2x-20090901.pdf`, (2009).

[3] D. V. Bailey et. al., "Breaking ECC2K-130," available at `http://eprint.iacr.org/2009/541.pdf`, (2009).

[4] I. Blake, G. Seroussi and N. Smart, Elliptic Curves in Cryptography, Cambridge University Press, (1999).

[5] Certicom, Certicom ECC Challenge, available at `http://www.certicom.jp/images/pdfs/cert_ecc_challenge.pdf`, (1997).

[6] Certicom, Curves List, available at `http://www.certicom.jp/index.php/curves-list`, (1997).

[7] J. Delescaille and J. Quisquater, "How easy is collision search? Application to DES", *Advances in Cryptology-EUROCRYPT 1989*, LNCS **434**, pp. 429–434, (1990).

[8] R. Gallant, R. Lambert and S. Vanstone, "Improving the Parallelized Pollard Lambda Search on Binary Anomalous Curves," Mathematics of Computation **69**, pp. 1699–1705, (2000).

[9] D. Hankerson, A. Menezes and S. Vanstone, Guide to Elliptic Curve Cryptography, Springer Professional Computing, (2004).

[10] R. Harley, Elliptic curve discrete logarithms project, available at `http://pauillac.inria.fr/~harley/ecdl/`.

[11] D. Knuth, The art of computer programming, Seminumerical Algorithms, vol. II, Addison-Wesley. Reading (1969).

[12] N. Koblitz, "Elliptic curve cryptosystems," Mathematics of Computation **48**, pp. 203–209, (1987).

[13] A. Menezes, T. Okamoto and S. Vanstone, "reducing elliptic curve logarithms to logarithms in a finite field," IEEE Transactions on Information Theory **39**, pp. 1639–1646, (1993).

[14] J. Pollard, "Monte Carlo methods for index computation mod p," Mathematics of Computation **32**, pp. 918–924, (1978).

[15] T. Satoh and K. Araki, "Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves," Commentarii Mathematici Universitatis Sancti Pauli **47**, pp. 81–92, (1998).

[16] I. Semaev, "Evaluation of discrete logarithms in a group of p-torsion points of an elliptic curve in characteristic p," Mathematics of Computation **67**, pp. 353–356, (1998).

[17] J. H. Silverman, The Arithmetic of Elliptic Curves, Graduate Texts in Math. Springer-Verlag, Berlin-Heidelberg-New York, (1986).

[18] N.P. Smart, "The discrete logarithm problem on elliptic curves of trace one," Journal of Cryptology **12**, pp. 110–125, (1999).

[19] E. Teske, "Speeding up Pollard's rho method for computing discrete logarithms", *Algorithmic Number Theory-ANTS III*, LNCS **1423**, pp. 541–554, (1998).

[20] E. Teske, "On random walks for Pollard's rho method", Mathematics of Computation **70**, pp. 809–825, (2001).

[21] P. C. van Oorschot and M. J. Wiener, "Parallel collision search with cryptanalytic applications", Journal of Cryptology **12**, pp. 1–28, (1999).

[22] M. J. Wiener and R. J. Zuccherato, "Fast attacks on elliptic curve cryptosystems", *Selected Areas in Cryptology-SAC '98*, LNCS **1556**, pp. 190–200, (1999).

Masaya Yasuda, Tetsuya Izu, Takeshi Shimoyama and Jun Kogure
FUJITSU LABORATORIES LTD., 1–1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki, 211-8588, Japan
E-mail: Not available