Implementation of Schoof’s Algorithm on the Windows Platforms

Huiting Hsieh, Chunghuang Yang

Graduate Institute of Information and Computer Education
National Kaohsiung Normal University, Taiwan
hsieh.angle@gmail.com, chyang@nkucc.nknu.edu.tw

Abstract. In FIPS 186-2, NIST (National Institute of Standards and Technology) recommends ten secure elliptic curve parameters over binary field. If the user can randomly produce parameters over binary field, the cryptosystem will be more secure. One of the important steps is to count the number of points on an elliptic curve over binary field and find the curve of prime order. The study designed the graphic user’s interface to implement Schoof’s algorithm on Windows platforms. The system uses C++ Builder to design the interface and MIRACL as embedded function library. Besides counting points on an elliptic curve, the system can search out secure parameters of the elliptic curve over binary field. This can provide users convenient way for building up secure elliptic curve cryptosystems.

Key words: Elliptic Curve Cryptosystem (ECC), Schoof’s Algorithm, Binary Field, Cryptography

1 Introduction

In 1985, Neil Koblitz [1] and Victor Miller [2] proposed the elliptic curve cryptosystem (ECC). It can apply to encryption, digital signature, key exchange and integer factorization. Because the length of key is smaller than other public key cryptosystem on the same secure condition, it has advantages of quick calculation and small storing space. Table 1 [3] is the comparison of key length. Now ECC had been formulated in many international standards.

The security of ECC is based on the difficulty of elliptic curve discrete logarithm problem (ECDLP). Finding the elliptic curve of prime order is important for its security. At present, there are two kinds of methods to solve the points counting problem.

1. Complex multiplication method (CM method): Determine the order of elliptic curve to find suitable curve of this order, and then confirm the point of big prime order on this curve. For elliptic curve over prime fields, the CM method is also called the Atkin-Morain method; for binary fields it is called the Lay-Zimmer method [4].
Table 1. The comparison of key length [3]

<table>
<thead>
<tr>
<th>Symmetric key size (bits)</th>
<th>$2^{80}$</th>
<th>$2^{112}$</th>
<th>$2^{128}$</th>
<th>$2^{192}$</th>
<th>$2^{256}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA and DH key size (bits)</td>
<td>1024 2048 3072 7680 15360</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elliptic curve key size (bits)</td>
<td>160 224 256 384 512</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ratio of DH cost : EC cost</td>
<td>3:1 6:1 10:1 32:1 64:1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Random selection: Choose randomly an elliptic curve parameters and count the rational number of points on this curve until the order is prime number [5]. This method can be categorized into $l$-adic method and $p$-adic method.

(1) $l$-adic method: The $l$-adic method is efficient when characteristic is a big prime number [5]. The following are two algorithms.

A. Schoof’s algorithm: Proposed in 1985, R. Schoof gave a polynomial-time algorithm [6], which uses Forbenius endomorphism and division polynomial. The time complexity is $O(\log^8 q)$. But the degree of division polynomial is too huge; therefore, it requires a lot of space for storing.

B. SEA algorithm: In order to improve the efficiency of Schoof’s algorithm, Elkies and Atkin proposed a modified method, called SEA algorithm (Schoof-Elkies-Atkin algorithm). The time complexity in SEA algorithm is $O(\log^6 q)$. Afterwards, Morain gave the method of using isogeny cycle, and some people, Couveignes, Lercier, etc., optimized it to advance its efficiency [5], which enables SEA algorithm to be more valuable to build up the secure elliptic curve cryptosystem.

(2) $p$-adic method: In 1999, Satoh proposed a new method for counting points over finite fields of small characteristic [4]. Variants of Satoh’s method are Satoh-Skjernaa-Taguchi (SST) algorithm and Arithmetic-Geometric-Mean (AGM) algorithm.

2 Related Works

2.1 Elliptic Curve Cryptography (ECC)

A binary field is the field $F_{2^n}$ which contains $2^n$ elements for integer $n$, and $n$ is called the degree of the field. The elliptic curve $E$ over binary field has the form $E : y^2 + xy = x^3 + ax^2 + b$. And curve $E$ is non-supersingular with $j$-invariant $= 1/b$ and discriminant $\Delta = b$ is nonzero [4]. $E = \{P(x, y)|x, y \in F_q\}$ is satisfied $y^2 + xy = x^3 + ax^2 + b \cup \{O\}$, where $q = 2^n$ and the extra point $\{O\}$ is called the point at infinity. The points on elliptic curve $E$ over binary field form an Abelian group, and the definitions are described later. Suppose that $P$, $Q$ and $S$ are three points on the elliptic curve $E$. The group law of points on the elliptic curve describes as the following.
1. Addition identity: $P + O = O + P = P$ for all $P \in E$
2. Addition inverses: $P + (-P) = O$ for all $P \in E$
3. Associativity: $(P + Q) + S = P + (Q + S)$ for all $P, Q, S \in E$
4. Commutativity: $P + Q = Q + P$ for all $P, Q \in E$

Let three points $P = (x_1, y_1), Q = (x_2, y_2)$ and $R = (x_3, y_3)$ on $E(F_q)$. The addition inverse $-P = (x_1, y_1 + x_1)$. If $R = P + Q$, then

$$
x_3 = \begin{cases} 
\lambda^2 + \lambda + a + x_2 + x_1 + x_1, & \text{if } P \neq Q \\
\lambda^2 + \lambda + a = x_1^2 + \frac{b}{x_1^2}, & \text{if } P = Q 
\end{cases}
$$

$$
\lambda = \begin{cases} 
\frac{y_2 + y_1}{x_1 + x_2}, & \text{if } P \neq Q \\
\frac{x_1 + y_1}{x_1}, & \text{if } P = Q 
\end{cases}
$$

### 2.2 Elliptic Curve Discrete Logarithm Problem (ECDLP)

Let $E$ be an elliptic curve over finite field, and $n$ is the order of $E$. Let two points $P, Q \in E$, and $Q \in \langle P \rangle$. The elliptic curve discrete logarithm problem (ECDLP) on $E$ is to find an integer $x$ s.t. $Q = [x]P$ [1, 7]. And $[m]P = P + P + \cdots + P$ ($m$ times) is called the multiplication-by-$m$ map. The security of the elliptic curve cryptosystem is based on the difficulty level of ECDLP. In order to assure the security of the cryptosystem, the chosen curve must avoid the known attacks for elliptic curve, such as baby-step giant-step, Pollard $\rho$ method, Pollard $\lambda$ method, Pohlig-Hellman method, MOV attack, etc.

**Shanks’s Method (Baby-step Giant-step, BSGS).** Proposed by Shanks in 1971, the method is to search $m = \lfloor \sqrt{n} \rfloor a + b$ s.t. $Q = [m]P$, with $P, Q \in G$ and $0 \leq a, b < \lfloor \sqrt{n} \rfloor - 1$. The equation $Q = \lfloor \sqrt{n} \rfloor a + b P$ can be transposed to $Q - [b]P = [a]([\lfloor \sqrt{n} \rfloor]P)$. Then $R_b = Q - [b]P$ ($0 \leq b < \lfloor \sqrt{n} \rfloor - 1$) is called baby-step, and the computed values are stored in the table. $S_a = [a]([\lfloor \sqrt{n} \rfloor]P)$ is called the giant-step, on which every computed value must be contrasted to see if there is the same value in the table of baby-step. If there is the same value, it means $m$ can be computed by suitable $a$ and $b$ [7, 8].

### 2.3 Schoof’s Algorithm

The safety of elliptic curve cryptosystem is based on the difficulty of elliptic curve discrete logarithm problem. Hence counting the number of points on an elliptic curve is important for its security. Schoof’s algorithm is the method of solving the points counting problem.

**Hasse’s Theorem.** The number of the points on elliptic curve $E : y^2 + xy = x^3 + ax^2 + b$ over $F_q$ is $\#E(F_q) = q + 1 - t$, where $|t| \leq 2\sqrt{q}$.
**Torsion Point.** Let $K$ be a finite field $F_q$ and $E(K)$ is a torsion group, which means that every point on elliptic curve $E$ has finite order. $E(m)$ is the set of $m$-torsion points of $E$ defined by $E(m) = \{ P \in E(K) | [m]P = O \}$ [7].

Let $P$ be a point on $E(K) \setminus O$ and $m \geq 1$. Then $P \in E[m] \iff \psi_m(P) = 0$ [7]. And the division polynomial is the polynomial with the roots which is the $x$-coordinate of every point in $E(m)$ except for the point at infinite $O$ [7].

**Division Polynomial.** (Over binary field) Let $E: y^2 + xy = x^3 + ax^2 + b$ be an elliptic curve over $F_q$, and the $m$-th division polynomial $\psi_m \in F_q[x,y]$ is defined by the following recursion [7]:

- $\psi_0 = 0, \psi_1 = 1, \psi_2 = x, \psi_3 = x^4 + x^3 + b, \psi_4 = x^6 + bx^2$
- $\psi_{2m+1} = \psi_m^3 \psi_{m+2} + \psi_{m-1} \psi_2^3, m \geq 2$
- $\psi_{2m} = (\psi_{m-1} \psi_{m+2} + \psi_{m-2} \psi_{m+1}) \psi_m/x, m \geq 3$. Let point $P = (x, y) \in E(K) \setminus E[m]$ and an integer $m \geq 2$. Then $[m]P = \left( x + \frac{\psi_{m-1} \psi_{m+1}}{\psi_m^2}, x + \frac{(x^2 + x + y) \psi_{m-1} \psi_{m+1} + \psi_{m-2} \psi_{m+2}}{x \psi_m^3} \right)$.

**Forbenius Endomorphism.** Let $\varphi$ be the Frobenius endomorphism on the curve $E: y^2 + xy = x^3 + ax^2 + b$ over algebraic closure field $\overline{F}_q$ of $F_q$. Then $\varphi$ is defined by the mapping $\varphi: (x, y) \to (x^q, y^q)$, with $x, y \in \overline{F}_q$, and the points $(x, y)$ and $(x^q, y^q)$ are on $E$. And it is satisfied the equation $F_1: \varphi^2 - t \varphi + q = 0$ [8]. By Hasse’s theorem, $\#E(F_q) = q + 1 - t$ is the number of the points on elliptic curve $E(F_q)$, and in $\#E(F_q) = q + 1 - t$, an integer $t$ is called the trace of Frobenius endomorphism $\varphi$.

**The Idea on Schoof’s Algorithm.** Schoof [9] gave a polynomial-time algorithm to count points on elliptic curve in 1985. The main idea on Schoof’s algorithm is to collect a set $S = \{2, 3, \cdots, L\}$ of primes s.t. $\prod_{l \in S} l > 4\sqrt{q}$, and to calculate $t \mod l$ for every prime $l$ in $S$. Followed by Chinese Remainder Theorem (CRT), $t$ can be calculated, and Hasse’s Theorem, $\#E(F_q) = q + 1 - t$ is the order of the elliptic curve [7].

Finally, it is the Schoof’s algorithm.
Algorithm: Schoof’s algorithm \[7\]
Input: An elliptic curve \(E\) over finite field \(F_q\)
Output: The order of \(E(F_q)\)

Step:  
1. \(M \leftarrow 2, l \leftarrow 3, S \leftarrow \{(t \mod 2, 2)\}\)
   
2. While \(M < 4\sqrt{q}\) do:
   
   - For \(\tau = 0, \cdots , \frac{l-1}{2}\) do:
     
     - Check exact one \(\tau\) can pass the test \(\varphi^2(P) + [q]P = \pm [\tau]\varphi(P),\) for \(P \in E[l]\)
     
     - \(S \leftarrow S \cup \{(\tau, l)\}\) or \(S \leftarrow S \cup \{(-\tau, l)\}\)
     
     - \(M \leftarrow M \times l\)
     
     - \(l \leftarrow \text{nextprime}(l)\)
   
3. Calculate \(t\) by using CRT after collecting enough \(t\) in set \(S\)

4. Return \(#E(F_q)\) \(\leftarrow q + 1 - t\)

2.4 Optimization of Schoof’s Algorithm

The Early Abort Strategy. To randomly search the elliptic curve of prime order, it requires to choose the random parameters \(a\) and \(b\), calculate the number of points on the elliptic curve, and determine whether the order is prime number. If the order is not prime, the curve needs to be chosen again. The steps above should be repeated until the order of curve is prime number. Before counting points, if the order \(#E(F_q) = q + 1 - t\) can be judged that it is not a prime, which means that an integer \(l\) exists and \(q + 1 - t\) is divisible by \(l\), then the order of this curve doesn’t have to be calculated. Hence, the calculation will be aborted early, and the next curve will be selected.

Counting Optimization of Point Addition and Point Doubling. The point \((x, y)\) on the elliptic curve is called the affine coordinate. The formula of point addition and doubling of elliptic curve \(E: y^2 + xy = x^3 + ax^2 + b\) over binary field in affine coordinate is shown in section 2.1. Table 2 \[7, 10\] contrasts the cost of point addition and doubling among different coordinates. And the multiplication, squaring, and inversion operation are considered and marked with M, S, and I respectively. The computational cost of point addition and doubling over binary field is \(2M + 1S + I\). The field inversion is used in point addition and doubling over binary field, which takes much time. Therefore, in order to reduce the computational cost, the affine coordinate can transform into projective coordinate or Jacobian coordinate. The computational costs of point addition and point doubling in Jacobian coordinate are \(15M + 5S\) and \(5M + 5S\). The cost of point addition can be reduced to \(14M + 4S\) with \(a = 0\). In addition, the cost of \(14M + 4S\) can be decreased to \(10M + 3S\) as \(z_1 = 1\).

Choice of Underlying Field \(q\). There are two types of bases for binary field, including polynomial basis and normal basis. The polynomial basis is an
Table 2. Comparison computational cost [7, 10]

<table>
<thead>
<tr>
<th></th>
<th>Coordinate Point addition ($a \neq 0$)</th>
<th>Point addition ($a = 0$)</th>
<th>Point doubling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Affine</td>
<td>2M+1S+1I</td>
<td>2M+1S+1I</td>
<td>2M+1S+1I</td>
</tr>
<tr>
<td>Jacobian</td>
<td>15M+5S</td>
<td>14M+4S</td>
<td>5M+5S</td>
</tr>
<tr>
<td>As $z_1 = 1$</td>
<td>11M+4S</td>
<td>10M+3S</td>
<td></td>
</tr>
</tbody>
</table>

basis can be showed the form as irreducible binary trinomial $t^m + t^k + 1$ and pentanomial $t^m + t^a + t^b + t^c + 1$. In addition, for each $m$ not divisible by 8 is given the minimal type among Gaussian normal bases for $F_{2^m}$.

Choice of Curve. Two kinds of curves over binary field are given:

1. Pseudo-random curves: The pseudo-random curve has the form $E : y^2 + xy = x^3 + x^2 + b$. The cofactor of pseudo-random curve is $h = 2$.

2. Special curves: The special curve over $F_{2^m}$ is called a Koblitz curve or anomalous binary curve. The Koblitz curve has the form $E_a : y^2 + xy = x^3 + ax^2 + 1$, where $a = 0$ or 1. The cofactor $h$ of Koblitz curve is $h = \begin{cases} 4, & \text{if } a = 0 \\ 2, & \text{if } a = 1 \end{cases}$.

3 System Implementation

3.1 Brief Introduction of MIRACL

MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library) [11], which is a multiprecision big number function library, has been developed by the Shamus Software Ltd since 1988. MIRACL supports many cryptographic system implementations, such as RSA public key cryptography, Diffie-Hellman key exchange, DSA digital signature and elliptic curve cryptography. MIRACL offers full support for elliptic curve cryptography over $F_p$ and $F_{2^m}$. The latest version is 5.4.1, which is released on Feb. 26, 2010. MIRACL provides the source code of complex multiplication method, Schoof’s algorithm, and SEA algorithm for solving the point counting problem of elliptic curve over finite field. This study adopts the source code of Schoof’s algorithm over binary field in MIRACL to implement system.

Function and Command of Execution File. Here we introduce the function and command of execution about implementing Schoof’s algorithm. The file schoof2.exe is the program to count the number of points on an elliptic curve $E : y^2 + xy = x^3 + Ax^2 + B$ over the binary field $F_{2^m}$.

For example, the command “schoof2 1 52 191 9 -o result.ecs” is to count points on the elliptic curve $E : y^2 + xy = x^3 + x^2 + 52$ over the underlying field
\[ t^{191} + t^3 + 1, \text{ and to output the outcome into the file result.ecs. Figure 1 is the screenshot of executing the program schoof2.exe to calculate the order of the curve on the cmd environment. If the command uses the flag “-s”, the execution will fix the coefficient } A \text{ and increase the coefficient } B \text{ to calculate the order of an elliptic curve until its order is near-prime. For instance, the command “schoof2 1 52 -s 191 9” is to search the curve of prime order with increasing the coefficient } B. \]

![Fig. 1. Program schoof2.exe calculates order of curve](image)

### 3.2 System Design and Implementation

Because MIRACL[11] provides the source code and execution to implement Schoof’s algorithm over binary field. We design the graphic user’s interface to implement Schoof’s algorithm over binary field on Windows platforms. The program uses Borland C++ Builder 6 to design the interface and MIRACL as embedded function library. Figure 2 is the procedure of system implementation. First, the prime number \( m \) is chosen as the degree of the binary field. Second, the irreducible polynomial basis is generated with this prime \( m \). Third, for every polynomial basis the system calculated the order of the Koblitz curve \( y^2 + xy = x^3 + ax^2 + 1 \) where \( a = 0 \) or 1. If the order is prime, the parameters of the secure curve are recorded on the system. Besides the Koblitz curve, the system can count points on the pseudo-random curve \( E : y^2 + xy = x^3 + x^2 + b \). The system can generate random number \( b \) and calculate the order. Together with the primality test, the system can judge whether the order is prime or not. Therefore, the system can not only count points on curve but also search for
Related Strategies. We use some strategies to design the system, including selection of underlying fields and early abort strategy.

In order to avoid the attacks, the field \( q = 2^m \) of elliptic curve must satisfy that \( m \) is a prime. Table 3 and Table 4 are the underlying fields used in this system implementation. The degrees of underlying fields which are recommended by NIST in FIPS 186-2 are 163, 233, 283, 409 and 571. Since most numbers of the order are not primes, these curves can’t be used in secure cryptosystems. So the early abort strategy is adopted. That is to say, if the numbers of the order can be clearly judged that they are not primes, the system won’t count the points. And the fixed parameter \( a \) together with the increasing parameter \( b \) will do the next point-counting until the system searches out the curve of prime order.

Figure 3 is the interface of the implementation of Schoof’s algorithm on Windows platforms. The system has two main functions. One is “Counting points on elliptic curve”. Figure 3 (a) is the screenshot of counting points on pseudo-random curve. The other is “Increasing B to search for secure elliptic curve”, and this function plays a very important role in the system. Figure 3 (b) is the screenshot of providing the parameters of the secure Koblitz curve. User can choose the underlying field based on their own need before counting. If users don’t decide the underlying field, the default of the system will be \( q = t^{163} + t^2 + t^6 + t^3 + 1 \) of No. B-163.
Table 3. The underlying field $q$ of Koblitz curve

<table>
<thead>
<tr>
<th>No.</th>
<th>Underlying field</th>
<th>Parameter $a$</th>
<th>Large prime</th>
<th>Time (sec.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-163</td>
<td>$t^{163} + t^7 + t^6 + t^3 + 1$</td>
<td>1</td>
<td>31</td>
<td>100</td>
</tr>
<tr>
<td>K-233</td>
<td>$t^{233} + t^{74} + 1$</td>
<td>0</td>
<td>53</td>
<td>240</td>
</tr>
<tr>
<td>K-239</td>
<td>$t^{239} + t^{36} + 1$</td>
<td>0</td>
<td>53</td>
<td>422</td>
</tr>
<tr>
<td>K-277</td>
<td>$t^{277} + t^{12} + t^6 + t^3 + 1$</td>
<td>0</td>
<td>67</td>
<td>782</td>
</tr>
<tr>
<td>K-283</td>
<td>$t^{283} + t^{12} + t^7 + t^5 + 1$</td>
<td>0</td>
<td>71</td>
<td>733</td>
</tr>
<tr>
<td>K-311</td>
<td>$t^{311} + t^7 + t^5 + t^3 + 1$</td>
<td>1</td>
<td>79</td>
<td>1072</td>
</tr>
<tr>
<td>K-331</td>
<td>$t^{331} + t^{10} + t^6 + t^2 + 1$</td>
<td>1</td>
<td>83</td>
<td>2019</td>
</tr>
<tr>
<td>K-347</td>
<td>$t^{347} + t^{11} + t^{10} + t^3 + 1$</td>
<td>1</td>
<td>97</td>
<td>3146</td>
</tr>
<tr>
<td>K-349</td>
<td>$t^{349} + t^6 + t^5 + t^2 + 1$</td>
<td>0</td>
<td>97</td>
<td>3456</td>
</tr>
<tr>
<td>K-359</td>
<td>$t^{359} + t^{68} + 1$</td>
<td>1</td>
<td>97</td>
<td>4009</td>
</tr>
<tr>
<td>K-409</td>
<td>$t^{409} + t^{87} + 1$</td>
<td>0</td>
<td>109</td>
<td>6672</td>
</tr>
</tbody>
</table>

Table 4. The underlying field $q$ of pseudo-random curve

<table>
<thead>
<tr>
<th>No.</th>
<th>Underlying field</th>
<th>Digit of $B$</th>
<th>Number of curve</th>
<th>Large prime</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-163</td>
<td>$t^{163} + t^7 + t^6 + t^3 + 1$</td>
<td>50</td>
<td>10</td>
<td>31</td>
</tr>
<tr>
<td>B-179</td>
<td>$t^{179} + t^4 + t^3 + t + 1$</td>
<td>54</td>
<td>10</td>
<td>37</td>
</tr>
<tr>
<td>B-191</td>
<td>$t^{191} + t^9 + 1$</td>
<td>58</td>
<td>10</td>
<td>41</td>
</tr>
<tr>
<td>B-233</td>
<td>$t^{233} + t^{74} + 1$</td>
<td>71</td>
<td>8</td>
<td>53</td>
</tr>
<tr>
<td>B-239</td>
<td>$t^{239} + t^{96} + 1$</td>
<td>72</td>
<td>8</td>
<td>53</td>
</tr>
<tr>
<td>B-257</td>
<td>$t^{257} + t^{12} + 1$</td>
<td>78</td>
<td>8</td>
<td>61</td>
</tr>
<tr>
<td>B-283</td>
<td>$t^{283} + t^{12} + t^7 + t^5 + 1$</td>
<td>86</td>
<td>5</td>
<td>71</td>
</tr>
<tr>
<td>B-307</td>
<td>$t^{307} + t^8 + t^4 + t^3 + 1$</td>
<td>93</td>
<td>3</td>
<td>79</td>
</tr>
<tr>
<td>B-331</td>
<td>$t^{331} + t^{10} + t^6 + t^2 + 1$</td>
<td>100</td>
<td>3</td>
<td>83</td>
</tr>
<tr>
<td>B-359</td>
<td>$t^{359} + 1$</td>
<td>109</td>
<td>2</td>
<td>97</td>
</tr>
<tr>
<td>B-367</td>
<td>$t^{367} + t^{21} + 1$</td>
<td>111</td>
<td>2</td>
<td>101</td>
</tr>
<tr>
<td>B-401</td>
<td>$t^{401} + t^{152} + 1$</td>
<td>121</td>
<td>2</td>
<td>109</td>
</tr>
<tr>
<td>B-409</td>
<td>$t^{409} + t^{87} + 1$</td>
<td>124</td>
<td>2</td>
<td>109</td>
</tr>
<tr>
<td>B-431</td>
<td>$t^{431} + t^{120} + 1$</td>
<td>130</td>
<td>2</td>
<td>127</td>
</tr>
<tr>
<td>B-571</td>
<td>$t^{571} + t^{10} + t^5 + t^2 + 1$</td>
<td>172</td>
<td>2</td>
<td>173</td>
</tr>
</tbody>
</table>
3.3 Implementation Data

In Figure 3 (b), only one curve of prime order can be searched in every execution in this system. In order to collect a large number of experimental data, the system of Figure 3 (b) is modified to Figure 3 (c). Users can make multiple choice of underlying fields they want to search in this system. Table 4 is the default of data record implementation. And after the searching, the system will automatically record every operational outcome for the users in the cryptosystem.

The hardware and software in this experiment are the computer with Pentium 2.0G CPU, 1G memory and Windows XP platforms. The underlying fields of Koblitz curves in Table 3 and seven underlying fields of pseudo-random curve in Table 4 are selected for testing the system efficiency. The data was analyzed in the individual underlying field shown as Table 3 and Table 5.

Table 5. The data analyzed in the individual $q$

<table>
<thead>
<tr>
<th>No.</th>
<th>$N_S$</th>
<th>$N_A$</th>
<th>$N_T$</th>
<th>$N_1$</th>
<th>P</th>
<th>$T_A$</th>
<th>$T_S$</th>
<th>$T_1$</th>
<th>$T_P$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>num.</td>
<td>num.</td>
<td>num.</td>
<td>num.</td>
<td>%</td>
<td>hr.</td>
<td>hr.</td>
<td>sec.</td>
<td>sec.</td>
</tr>
<tr>
<td>B-163</td>
<td>10</td>
<td>927</td>
<td>177</td>
<td>93</td>
<td>97</td>
<td>0.5</td>
<td>105</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>B-191</td>
<td>10</td>
<td>1298</td>
<td>228</td>
<td>130</td>
<td>18</td>
<td>17.1</td>
<td>270</td>
<td>243</td>
<td></td>
</tr>
<tr>
<td>B-233</td>
<td>10</td>
<td>1120</td>
<td>175</td>
<td>112</td>
<td>16</td>
<td>59.0</td>
<td>1214</td>
<td>1239</td>
<td></td>
</tr>
<tr>
<td>B-239</td>
<td>10</td>
<td>799</td>
<td>153</td>
<td>80</td>
<td>19</td>
<td>58.9</td>
<td>1386</td>
<td>1330</td>
<td></td>
</tr>
<tr>
<td>B-283</td>
<td>8</td>
<td>721</td>
<td>137</td>
<td>90</td>
<td>19</td>
<td>186.2</td>
<td>4894</td>
<td>4222</td>
<td></td>
</tr>
<tr>
<td>B-307</td>
<td>3</td>
<td>296</td>
<td>51</td>
<td>99</td>
<td>17</td>
<td>134.3</td>
<td>9478</td>
<td>9816</td>
<td></td>
</tr>
<tr>
<td>B-359</td>
<td>1</td>
<td>110</td>
<td>16</td>
<td>110</td>
<td>15</td>
<td>137.8</td>
<td>31006</td>
<td>33601</td>
<td></td>
</tr>
</tbody>
</table>

Because of adopting the early abort strategy, the system doesn’t count the number of points for every curve. Therefore, execution time $T_A$ includes the time spent in counting the points on the early abort curve. That is, $T_1$ isn’t
representative of the exact time of counting points on the curve. The time are measured again to count points on every prime order curve that had been searched out in order to get the exact counting time, and the average time is $T_p$.

**Symbol description:**
- $N_S$: searching out the number of secure elliptic curve
- $N_A$: searching out the number of all the elliptic curve
- $N_T$: the number of the curve which is indeed counted
  - $N_A - N_T$: the number of curve which is terminated early
- $N_1$: the average number of searched curve (when a curve of prime order is searched) = $N_A / N_S$
- $P$: the proportion of the curve which is indeed counted the number of points
  - $P = (N_T / N_A) \times 100\%$
- $T_A$: total execution time
- $T_S$: the average time of searching out the curve of prime order = $T_A / N_S$
- $T_1$: the average time of counting points on a curve = $T_A / N_T$
- $T_p$: the average time of counting points on a curve of prime order

### 3.4 Parameter of Elliptic Curve

Finally, the study provides two elliptic curve parameters that were produced by system of implementation of Schoof’s algorithm.

**The parameters of secure Koblitz curve**

$m = 409$
$q = t^{409} + t^{87} + 1$
$a = 0$
$b = 1$
order = 7FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
E5F83B2D4EA20400EC4557D5ED3E7CA5B45C83B8E01E5FCF
$x = 1E502DEAA20589942FD5521F08367E3F21BA52531CE0A59BD0ED26753$
D06F8E83F9A74EAB20CB53C9EC55B8D047EF58E8
$y = 305C3A4F659F3524FAACD9928EB921A95D0B044F98BCC4212887EC6604B$
F54D0DC4C97FA041C44A74E31D2DB82B4F137CF64

**The parameters of secure pseudo-random curve**

$m = 359$
$q = t^{359} + t^{68} + 1$
$a = 1$
$b = 5DFDB296E6296AADECEF7A801209CC31779915DA601D3C709F4671A4D7D6$
94414BC98BE9754F14706C5A6A8CD
order = 40000000000000000000000000000000000000000000007D0104F38263B29A31$
A1D3ABB026930D4D63A2C0A19$
$x = 2CF4E3A3051EB715D11C6C9F79F9905270285E9875F993D84CA707708565$
FB212733083A6F7C9D8C71D18371$
$y = 54F4E50435350C1C4F7AD7064873E083C90CB14FBF219B5C496E62E78E$
311356CBC3777573D037664187C
4 Conclusions

Elliptic curve cryptosystem has been a mature development. Although NIST recommends ten elliptic curve parameters over prime field in FIPS 186-2, the cryptosystem will be more secure if users can randomly produce the elliptic curve parameters over binary field. For this reason, to implement Schoof’s algorithm for searching the elliptic curve of prime order is very important to build up secure elliptic curve cryptosystem.

Conversion from affine coordinate into projective coordinate or Jacobian coordinate can optimize the implementation. And setting the parameter $a = 0$ and $z_1 = 1$ can decrease the computational cost of point addition. In addition, when the system can determine that the number of points is not a prime, it doesn’t count points, and skips from this into another by adopting the early abort strategy.

In this study, the system implements Schoof’s algorithm on Windows platforms. The hardware of experimental computer is Pentium 2.0G CPU with 1G memory. On average, it takes respectively 0.5, 1.7, 5.9, 5.9, 23.3, 44.8 and 137.8 hours to search out elliptic curve over binary field in 163, 191, 233, 239, 283, 307 and 359 degree of underlying field and takes 93, 243, 1239, 1330, 4222, 9816 and 33601 seconds to count points on elliptic curve.

References