Counting points on Elliptic Curve defined over GF($2^n$) and its software implementation

KAZUYA HIRADATE
ARAKI-LAB M2
Today’s Contents

• PKC and One way function
• DLP on Elliptic Curve and Number
• 2-adic Numbers and its Valuation Ring R
• Bit-Slice technique
• Addition, Multiplication and Inversion in Z2 and R
• Running times in Z2 and R
• Sato-Skjernna-Taguchi (SST) Algorithm
• Hasse’Theorem, Isogeny, Frobenius morphism
• Lifting the j-invariant and 2-torsion, compute trace
• Running Times over GF(2^7)
Public key cryptosystem

Transmitter → Plain text → Encryption → Cipher text → Decryption → Plain text → receiver

Transmitter’s secret key

Receiver’s Public key

Receiver’s secret key
One Way function

\[ x_{\text{public}} = f(x_{\text{secret}}) \]

\[ f^{-1}(x_{\text{public}}) = x_{\text{secret}} \]

- Assume the function “f” is public. The inverse function of “f” can’t be solved from public information. □ one way function

- Secret information can be gained by special way using each user’s secret key.
Elliptic Curve over Real Number

\[ E : y^2 + a_1 xy + a_3 y = x^3 + a_2 x^2 + a_4 x + a_6 \]

\[ \text{(x,y)} \]

\[ P + Q = -R \]
Discrete logarithm problem on Elliptic Curve over Finite Field

\[ k \times P = Q \mod \#E \]

- \( k \): Real Number (not zero)
- \( P, Q \): point on Elliptic curve over finite field
- \( \#E \): The number of points on Elliptic curve

- Let \( P \) be public. Then, heavy computation is needed to get \( k \) from given \( Q \). \( \Box \) DLP on EC
- The more the number of points are, the more difficult to solve DLP on EC is.
- Counting points on random Elliptic Curves is needed and must be fast applying CRYPTOGRAPHY.
Elliptic Curves over GF(2^n)

• Why GF(2^n)?
  – It’s easy to implement on computer because binary representation

• The form is following,
  \[ E : y^2 + xy = x^3 + a_6 \quad a_6 \in GF(2^n) \]

• j-invariant is defined by:
  \[ j(E) = 1/a_6 \quad j(E) \in GF(2^n) \]

This is the characteristic value for the each curve.
Numbers

p: prime

\( \mathbb{R} \)  
Real number

\( \mathbb{Q} \)  
rational number

\( \mathbb{Q}_p \)  
p-adic number

\( \mathbb{Z}_p \)  
p-adic integer

Field  \rightarrow  Ring
The Ring $\mathbb{Z}_2$ — 2-adic integer

• Definition (2-adic integer)
  – Let $\pi_n$ be the projection $\mathbb{Z}/2^{n+1}\mathbb{Z} \to \mathbb{Z}/2^n\mathbb{Z}$
  – A sequence $x=(x_1\ldots x_n\ldots)$, with $x_n \in \mathbb{Z}/2^n\mathbb{Z}$ and such that $\pi_n(x_{n+1}) = x_n$ for $n \geq 1$.
  – The ring of 2-adic integers is denoted by $\mathbb{Z}_2$.

• More precisely, $x \in \mathbb{Z}_2$,
  \[ x = x_0 2^0 + x_1 2^1 + x_2 2^2 + \cdots + x_n 2^n + \cdots \quad x_i = \{0,1\} \]

• The index "n" is larger, the absolute value is smaller.
  \[ 2^0 + 2^1 > 2^0 + 2^2 \]
  – For example, that is, $3 > 5$. 
Valuation Ring R

• Let f(t) be a monic polynomial \( Z_2[t] \) of degree N such that the polynomial \( \pi(f) \) obtained by projecting the coefficients is irreducible in \( GF(2^N) \).

• Valuation Ring R (Definition):

\[
Z_2[t] \mod f(t)
\]

• As following diagram, \( GF(2), Z_2, R \) and \( GF(2^N) \) are related.
Valuation Ring $R$ on computer

- The index “$n$” of $\mathbb{Z}_2$ (called “precision”) must be finite to implement on computer.
- Normally, Valuation Ring $R$ can be implemented as following,

$$\mathbb{Z}^R_{\frac{M}{2}} = n = 0, 1, \ldots, N - 1$$

Where $M$ is a required precision and $W$ is WORD SIZE of CPU.

- We must treat $\mathbb{Z}_2$ as Multi precision integer.
Bit Slice Technique

- $W$ data are implemented simultaneously and this can be reduced redundancy in last one word of precision.
- Memory requirement is $N \times M$ words.
- An algorithm has any condition branches cannot be applied.
Bit Slice Addition in $\mathbb{Z}_2$

- Let $x, y \in \mathbb{Z}_2$ be 2-adic integers, then 2-elements addition is represented as following.

\[
x = x_0 \cdot 2^0 + x_1 \cdot 2^1 + x_2 \cdot 2^2 + \cdots + x_{n-1} \cdot 2^{n-1} + \cdots \quad x_i = \{0, 1\}
\]

\[
y = y_0 \cdot 2^0 + y_1 \cdot 2^1 + y_2 \cdot 2^2 + \cdots + y_{n-1} \cdot 2^{n-1} + \cdots \quad y_i = \{0, 1\}
\]

\[
x + y = (x_0 \oplus y_0) \cdot 2^0 + ((x_0 \& y_0) \oplus x_1 \oplus y_1) \cdot 2^1 + \\
\cdots + [(x_{n-2} \& y_{n-2}) \oplus (x_{n-2} \& (x_{n-3} \& y_{n-3})) \\
\oplus (y_{n-2} \& (x_{n-3} \& y_{n-3}))] \oplus x_{n-1} \oplus y_{n-1}] \cdot 2^{n-1} + \cdots
\]

- $(4n-5)$ times XOR operation and $(5n-9)$ times AND operation are needed for 2-adic integers addition which precision is $n$ ($n > 2$).
Bit Slice Subtraction in $\mathbb{Z}_2$

- Let $x, y \in \mathbb{Z}_2$ be 2-adic integers, then 2-elements subtraction is represented as following.

$$x = x_0 2^0 + x_1 2^1 + x_2 2^2 + \cdots + x_{n-1} 2^{n-1} + \cdots \quad x_i = \{0, 1\}$$
$$y = y_0 2^0 + y_1 2^1 + y_2 2^2 + \cdots + y_{n-1} 2^{n-1} + \cdots \quad y_i = \{0, 1\}$$
$$x - y = x + 1 + \{(1 \oplus y_0) 2^0 + (1 \oplus y_1) 2^1 + \cdots + (1 \oplus y_{n-1}) 2^{n-1} + \cdots\}$$

where $-y$ is represented as two’s complement of $y$ and $x-y$ can be calculated as additions in $\mathbb{Z}_2$.

- $(9n-10)$ times XOR operation and $(10n-18)$ times AND operation are needed for 2-adic integers subtraction which precision is $n \ (n>2)$.
Bit Slice ADD and SUB in R

• Let X, Y be 2-adic Valuation Ring, then 2 elements addition and subtraction are represented as following.

\[
X = X_0 + X_1 \psi + X_2 \psi^2 + \cdots + X_{n-1} \psi^{n-1} \quad X_i \in \mathbb{Z}_2
\]

\[
Y = Y_0 + Y_1 \psi + Y_2 \psi^2 + \cdots + Y_{n-1} \psi^{n-1} \quad Y_i \in \mathbb{Z}_2
\]

\[
X + Y = (X_0 + Y_0) + (X_1 + Y_1) \psi + \cdots + (X_{n-1} + Y_{n-1}) \psi^{n-1}
\]

• N times addition (subtraction) in \( \mathbb{Z}_2 \) can realize ADD (SUB) in R.
Bit Slice Multiplication in $\mathbb{Z}_2$

- Let $x, y \in \mathbb{Z}_2$ be 2-adic integers, then 2-elements multiplication is represented as following.

$$
x = x_0 2^0 + x_1 2^1 + x_2 2^2 + \cdots + x_n 2^n + \cdots \quad x_i = \{0, 1\}
$$

$$
y = y_0 2^0 + y_1 2^1 + y_2 2^2 + \cdots + y_n 2^n + \cdots \quad y_i = \{0, 1\}
$$

$$
x \times y = x_0 \& (y_0 2^0 + \cdots + y_{n-1} 2^{n-1})
$$

$$
+ x_1 2^1 \& (y_0 2^0 + \cdots + y_{n-1} 2^{n-1})
$$

$$
\cdots + x_n 2^{n-1} \& y_0 2^0
$$

- $\square (4k-5)$ times XOR operation and
- $\square (5k-9) + n(n+1)/2$ times AND operation are needed for 2-adic integers multiplication which precision is n ($n > 2$).
Efficient Multiplication in R

• Karatsuba Method (1962)

Let \( X, Y \in R \) then,

\[
X \times Y = (X_A \psi^k + X_B)(Y_A \psi^k + Y_B)
\]

\[
= X_A Y_A \psi^{2k} + X_B Y_B + (X_A Y_B + X_B Y_A)\psi^k
\]

\[
= X_A Y_A \psi^{2k} + X_B Y_B
\]

\[
- \{X_A Y_A + X_B Y_B - (X_A + X_B)(Y_A + Y_B)\}\psi^k
\]

• If two elements have 2k-length, its multiplication can be realized 3-times k-length MUL and some ADD.

• The iteration control structure can be used.
Efficient Multiplication in $\mathbb{Z}_2$

- **Modified Karatsuba Method**

Let $x, y \in \mathbb{Z}_2$ and their precision be $M$. Let

$$Kara(x, y) = x_A y_A 2^{2k} + x_B y_B$$

$$-\{x_A y_A + x_B y_B - (x_A + x_B)(y_A + y_B)\}2^k$$

$$Mkara(x, y) = x_B y_B + (x_A y_B + x_B y_A)2^k \ (k > M/2)$$

Then,

$$x \times y = (x_1 2^k + x_2)(y_1 2^k + y_2)$$

$$= Kara(x_2, y_2)$$

$$+ (Mkara(x_1, y_2) + Mkara(x_2 + y_1))2^k$$
Newton iteration and inversion

• Quadratic convergence of Newton iteration

Let \( x \in R \) and \( f(x) \in R[t] \). Let \( k \) be such that \( 2^k \parallel f'(x) \) and assume \( 2^{n+k} \mid f(x) \) for some \( n > k \). Let

\[
\Delta = \frac{2^{-k} f(x)}{2^{-k} f'(x)}
\]

\[
y = x - \Delta
\]

Then \( y \equiv x \mod 2^n \), \( 2^{2n} \mid f(y) \) and \( 2^k \parallel f'(y) \)

• Inverse of an invertible \( a \in R \) can be obtained by Newton iteration whose \( f(x) = 1/x - a \). That is,

\[
x \leftarrow x - f(x) / f'(x) = x + x(1 + ax)
\]
# Running Times

<table>
<thead>
<tr>
<th>CPU</th>
<th>Pentium □ 1GHz (seagull)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second cache</td>
<td>256KB</td>
</tr>
<tr>
<td>Main memory</td>
<td>512MB</td>
</tr>
<tr>
<td>Programming lang.</td>
<td>C++</td>
</tr>
<tr>
<td>Compiler</td>
<td>gcc version 2.95.3</td>
</tr>
<tr>
<td>Compile option</td>
<td>-O3</td>
</tr>
</tbody>
</table>
Running Times in $\mathbb{Z}_2$ (pre=95)

<table>
<thead>
<tr>
<th>Operation</th>
<th>BITSLICE (*32)</th>
<th>BITSLICE (*1)</th>
<th>NORMAL (*1)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[ m s]</td>
<td>[ s]</td>
<td>[ s]</td>
</tr>
<tr>
<td>ADD</td>
<td>5.4</td>
<td>0.169</td>
<td>0.709</td>
</tr>
<tr>
<td>SUB</td>
<td>17.8</td>
<td>0.556</td>
<td>0.734</td>
</tr>
<tr>
<td>MUL</td>
<td>130.1</td>
<td>4.066</td>
<td>5.17</td>
</tr>
<tr>
<td>INV</td>
<td>7500</td>
<td>234.8</td>
<td>No data</td>
</tr>
</tbody>
</table>
## Running Times in R (deg=163)

<table>
<thead>
<tr>
<th>Operation</th>
<th>BITSLICE (*32) [ s]</th>
<th>BITSLICE (*1) [ s]</th>
<th>NORMAL (*1) [ s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>3099</td>
<td>96.84</td>
<td>231</td>
</tr>
<tr>
<td>SUB</td>
<td>3314</td>
<td>103.7</td>
<td>168</td>
</tr>
<tr>
<td>MUL</td>
<td>5224000</td>
<td>163250</td>
<td>144000</td>
</tr>
<tr>
<td>Karatsuba MUL</td>
<td>under construction</td>
<td>under construction</td>
<td>4100</td>
</tr>
</tbody>
</table>
How to count points - SST Algorithm

- Sato-Skjernna-Taguchi (SST) Algorithm
  - Time complexity: $O(N^{3.5})$
  - Memory complexity: $O(N^{2.5})$

- The process is following,
  - Lift $j$-invariant from $GF(2^n)$ to $R$
  - Determine the Kernel of 2-th Verschiebung $V$
  - Let $\mathfrak{m}$ The local parameter at the point at infinity. Then find the value $c^2$ from expansion $V(\mathfrak{m}) = c \mathfrak{m} + O(\mathfrak{m}^2)$
  - Find an integer $t$ satisfying $t^2 = \text{Norm}(c^2)$
  - Finally we get $\#E = 1 + 2^n - t$
Isogeny (Endomorphism)

- (definition) A rational map which is furthermore a group homomorphism

\[
\phi : P(x, y) \mapsto (x^{2^N}, y^{2^N})
\]

\[
[m] : P \mapsto mP
\]
Hasse’s Theorem

• Hasse’s Theorem (1934)

$$\# E = 2^N + 1 - t$$

$$-2\sqrt{2^N} < t < 2\sqrt{2^N}$$

• What is “t”??

$$\varphi \circ \varphi + [t] \circ \varphi + [2^N] = [0]$$

$$\varphi: Frobenius \text{ endomorphism}$$

$$[m]: m - \text{maps}$$
The little Frobenius

• 2-th power Frobenius (the little Frobenius) $\sigma$ is defined as follows.

$$\sigma : P(x, y) \mapsto (x^2, y^2)$$

$$\sigma = \sigma_0 \sigma_1 \cdots \sigma_{N-1}$$
The canonical lift of an ordinary elliptic curve $E$ is unique elliptic curve $E \hat{\square}$ defined over $\mathbb{R}$, which satisfies:

- The reduction of $E \hat{\square}$ is $E$.
- $\text{End}(E) = \text{End}(E \hat{\square})$
Modular polynomial and lifting j-inv

• 2-th modular polynomial is defined as:
\[ \Phi_2(X,Y) = X^3 + Y^3 - X^2 Y^2 + 2^4 \cdot 3 \cdot 3 \cdot (X^2 Y + XY^2) - 2^4 \cdot 3^4 \cdot 5^3 (X^2 + Y^2) + 3^4 \cdot 5^3 \cdot 4027XY + 2^8 \cdot 3^7 \cdot 5^6 (X + Y) - 2^{12} \cdot 3^9 \cdot 5^9 \]

• If two Elliptic curves E and E’ are related via a cyclic isogeny of degree N,
\[ \Phi_2(j(E), j(E')) = 0 \]

• (Lubin-Serre-Tate) If J is the j-invariant of the canonical lift of E, then there is a unique J in R such that
\[ \Phi_2(J, \Sigma(J)) = 0 \quad \text{and} \quad J \equiv j \mod 2 \]
Lift the j-invariant

- The process of lifting the j-invariant is iterative Newton’s method.

\[
\text{for (int } i = 0; i < \text{PRECISION} ; i++)\
\quad \text{\{\
\text{\quad } x = \Sigma(J) \mod 2^{i+1} ; \\
\text{\quad } J = J - \frac{\Phi_2(x, J)}{\partial_x \Phi_2(x, J)} \mod 2^{i+1} ; \\
\text{\}}\n\]
Lifting the Kernel of $\mathcal{E}$

- The Kernel of $\mathcal{E}$ is 2-torsion point of $E \mathcal{E}$ i.e. infinity and another non-trivial point.
- $x$ coordinates of non-trivial point can be computed by j-invariant of $E \mathcal{E}$ and $E'$ $\mathcal{E}$.

$$
\begin{align*}
\frac{x}{2} &= -\frac{(J'(J^2 + 195120J' + 4095J + 660960000)) / 2^{12}}{2(J'^2 + J'(563760 - 512J) + 372735J + 8981280000) / 2^9}
\end{align*}
$$
Computing the trace

- Let \( \Box \) be the local parameter of \( E \) \( \Box \) around infinity. Then, \( \Box(\Box) \) can be expanded as:

\[
\Sigma(\tau) = c\tau + O(\tau^2)
\]

- \( c \) is computed by:

\[
c^2 = \frac{J - (504 + 12096z)t}{J + 240t}
\]

where, \( z = x/2 \), \( t = (12z^2 + z)(J - 1728) - 36 \)

- Trace \( t \) is square root of

\[
t^2 \equiv \prod_{0 \leq i \leq N} c^2 \mod 2^{\text{PRECISION} - 1}
\]
Applicable??

• INPUT $j$-invariant $\in \text{GF}(2^N)$

OUTPUT $\text{trace} \in \mathbb{Z}_2$

It’s easy to translate bit-slice representation to normal one.

• Algorithm has NO Condition branch.

• We can apply bit-slice technique!
Running Times over GF(2^7)

<table>
<thead>
<tr>
<th>Irreducible polynomial</th>
<th>t^7+t+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>curve</td>
<td>Y^3+xy=x^3+1/j</td>
</tr>
<tr>
<td>j-invariant</td>
<td>t^5+t+1</td>
</tr>
<tr>
<td>trace precise (work precise)</td>
<td>2^6 (2^17)</td>
</tr>
<tr>
<td>trace</td>
<td>-3</td>
</tr>
<tr>
<td>#E</td>
<td>2^7+1-3=126</td>
</tr>
<tr>
<td>Time</td>
<td>1.454/32=0.0454[s]</td>
</tr>
</tbody>
</table>
Future works

• Counting points on EC over Large extension fields
• Implementing Karatsuba method
• Implementing faster algorithm of Lifting the j-invariant and Norm computing

…Thank you for time!
Addition in $\mathbb{Z}_2$

- Let $x, y \in \mathbb{Z}_2$ be 2-adic integers, then 2-elements addition is represented as following.

$$x = x_0 b^0 + x_1 b^1 + x_2 b^2 + \cdots + x_{n-1} b^{n-1} + \cdots \quad 0 \leq x_i \leq b$$

$$y = y_0 b^0 + y_1 b^1 + y_2 b^2 + \cdots + y_{n-1} b^{n-1} + \cdots \quad 0 \leq y_i \leq b$$

$$x + y =$$

- $(4n-5)$ times XOR operation and $(5n-9)$ times AND operation are needed for 2-adic integers addition which precision is $n$ ($n > 2$).