Improved Scalar Multiplication on Elliptic Curves Defined over $F_{2^{mn}}$

Dong Hoon Lee, Seongtaek Chee, Sang Cheol Hwang, and Jae-Cheol Ryou

We propose two improved scalar multiplication methods on elliptic curves over F_{q^n} where $q = 2^m$ using Frobenius expansion. The scalar multiplication of elliptic curves defined over subfield F_q can be sped up by Frobenius expansion. Previous methods are restricted to the case of a small m. However, when m is small, it is hard to find curves having good cryptographic properties.

Our methods are suitable for curves defined over medium-sized fields, that is, $10 \leq m \leq 20$. These methods are variants of the conventional multiple-base binary (MBB) method combined with the window method. One of our methods is for a polynomial basis representation with software implementation, and the other is for a normal basis representation with hardware implementation. Our software experiment shows that it is about 10% faster than the MBB method, which also uses Frobenius expansion, and about 20% faster than the Montgomery method, which is the fastest general method in polynomial basis implementation.

Keywords: Elliptic curve, scalar multiplication, Frobenius expansion, composite field.

Manuscript received June 20, 2003; revised Dec. 17, 2003.

Dong Hoon Lee (phone: +82 42 860 1436, email: dlee@etri.re.kr) and Seongtaek Chee (email: chee@etri.re.kr) are with the National Security Research Institute, Daejeon, Korea.

Sang Cheol Hwang (email: schwang@cnu.ac.kr) and Jae-Cheol Ryou (email: jcryou@cnu.ac.kr) are with the Chungnam National University, Daejeon, Korea.

I. Introduction

In elliptic curve cryptosystems, main operations such as key agreement and signing/verifying involve scalar multiplications using a large integer, k. The speed of scalar multiplication plays an important role in the efficiency of the whole system. In particular, fast multiplication is more crucial in some environments such as central servers, where large numbers of key agreements or signature generations occur, and in handheld devices with low computational power.

There are several points that influence the speed of multiplication: the choice of base field, the choice of curve, the representation of a point, the representation of a scalar, and the multiplication algorithm. There are several scalar multiplication methods used on general elliptic curves: the binary method, signed binary method, sliding window method, and the Montgomery method. According to [1], the Montgomery method [2] is the fastest among such general methods.

If an elliptic curve admits an efficient endomorphism, its use can speed up scalar multiplication. In 1991, Koblitz proposed anomalous binary curves and introduced the Frobenius expansion method to compute scalar multiplications [3]. In 1997, Solinas improved Koblitz's ideas by combining them with a non-adjacent form representation of scalars [4]. Muller [5], Cheon and others [6] extended the Frobenius expansion method to elliptic curves defined over $F_{2^{mn}}$, where m is small. Elliptic curves defined over composite field F_{q^n}, where $q = 2^m$, are attractive in that they admit a special endomorphism called the Frobenius map.

If m is small, however, it is hard to find curves having good cryptographical properties since there are only $2(2^m\text{-}1)$ such curves at most. Hence this small set of curves can be targets for attack. Therefore, it is reasonable to consider composite fields where m is of medium size such as $10 \le m \le 20$. Moreover, these composite fields can be used to develop efficient multiplication algorithms [7]-[9]. However previous methods become very inefficient as m grows larger, forcing us to use more general methods. Of course, there are security flaws in curves defined over composite fields such as Weil descent attacks, but we can control the genus of the Jacobian obtained by a Weil descent attack via the magic number m [10].

Conventionally, for a large m, we can modify the binary method for application to multiple base points, which are obtained by applying the Frobenius map. This method, called multiple-base binary (MBB), was used to speed up scalar multiplication on elliptic curves defined over an Optimal Extension Field of odd characteristic [11]. It is faster than the Montgomery method.

In this paper, we propose two variants of the MBB method, combining it with the window method when the defining field is made up of composite degree mn and when m is of medium size. One of our proposals is suitable for the normal basis representation where we assume the squaring field elements are free, thus the same is true for a Frobenius map. The other proposal is apt for the ordinary polynomial basis representation. Our methods are 20% faster than the general Montgomery method and about 10% faster than the conventional MBB method in the polynomial basis implementation.

This paper is organized as follows. In section II, we recall the preliminaries on the representation of field and elliptic curve elements. We also recall the basic Frobenius expansion method. In section III, we describe the proposed scalar multiplication methods together with previous methods. We consider some security issues about curves over subfields in section IV and conclude in section V.

II. Preliminaries

1. Representation of Finite Fields

Let F_{q^n} be a finite field of q^n elements where $q = 2^m$. The elements of F_{q^n} can be represented in different ways including the polynomial basis and normal basis. In the polynomial basis, any element a of F_{q^n} is represented as a polynomial of the form

$$a = a_{mn-1}x^{mn-1} + a_{mn-2}x^{mn-2} + \cdots + a_1 x + a_0,$$

where $a_1 \in \{0,1\}$.

In the normal basis, $a \in F_{q^n}$ is represented as

$$a = a_0\theta + a_1\theta^2 + \cdots + a_{mn-2}\theta^{2^{mn-2}} + a_{mn-1}\theta^{2^{mn-1}},$$

where $a_i \in \{0,1\}$ and θ is a generator of the normal basis. In this basis, the cost of the squaring operation is almost free, as is that of the Frobenius map. But multiplication is not as fast as in the polynomial basis.

Generally, polynomial basis representation is proper for software implementation and normal basis representation is proper for hardware implementation. We'll propose two scalar multiplication methods suitable for each basis.

2. Representation of Points

A non-supersingular elliptic curve $E(F_{q^n})$ over q^n is given by the Weierstrass equation in the form

$$E(F_{q^n}) : y^2 + xy = x^3 + ax^2 + b,$$

together with the point at infinity denoted by O, where $a, b \in F_q$ and $b \neq 0$. We will write E to denote $E(F_{q^n})$ unless otherwise specified.

The basic operations on an elliptic curve such as addition and doubling require field inversions. Since a field inversion is very expensive, it is more efficient to use a projective coordinate system to avoid inversions. Among the several projective coordinate systems available, we will use the Lopez-Dahab's system, which provides the fastest operations. Table 1 shows the number of field operations needed to execute basic operations on an elliptic curve: addition (ADD, ADD1, ADD2), doubling (DBL), and Frobenius map (ϕ). Both *ADD1* and *ADD2* are the sum of two points represented with a projective coordinate system, but in the case of *ADD1*, the z component of one of the points is restricted to 1, while *ADD2* allows both points to have a general projective coordinate. See [12] for more details.

Table 1. The number of field arithmetics for basic operations on an elliptic curve (a=0 or 1).

Operation		Inv.	Mul.	Sqr.
Affine	*ADD*	1	2	1
	DBL	1	2	1
	ϕ	0	0	$2m$
Projective	*ADD1*	0	9	4
	ADD2	0	13	6
	DBL	0	4	5
	ϕ	0	0	$3m$
Projective to affine		1	2	1

3. Frobenius Expansion

The Frobenius map ϕ of E is defined by

$$\phi : E \to E, \quad (x, y) \mapsto (x^q, y^q).$$

Let t be the trace of the Frobenius map. The number of F_q-rational points of E is then given by the equation

$$\# E(F_q) = q + 1 - t,$$

and the following equation holds in the endomorphism ring:

$$\phi^2 - t\phi + q = 0. \tag{1}$$

Since there is a natural homomorphism from the ring Z[α] to the endomorphism ring End(E) which maps $\alpha = (t + \sqrt{t^2 - 4q})/2$ to ϕ, if an integer k is expressed as $k = \sum c_i \alpha^i$, then we can get a corresponding representation of $k = \sum c_i \phi^i$ in the endomorphism ring. This implies that kP can be computed as $kP = \sum c_i \phi^i (P)$.

Lemma 1. Let $s \in \mathrm{Z}[\phi]$. An integer, $y \in \{-q/2, \cdots, q/2\}$, and an element, $x \in \mathrm{Z}[\phi]$, exist such that

$$s = x\phi + y.$$

Proof. See [5]. □

Lemma 2. Let $q \geq 64$ and denote by N the norm function from Z[ϕ] to Z given by $N(a+b\phi)=a^2+tab+qb^2$ for $a, b \in \mathrm{Z}$. Given $s \in \mathrm{Z}[\phi]$, set

$$l_1 = \begin{cases} \lceil \log_q (N(s)) \rceil + 1 & \text{if } |t| \leq \sqrt{q}, \\ \lceil \log_q (N(s)) \rceil + 3 & \text{otherwise}. \end{cases}$$

Integers $c_j \in \{-q/2, \cdots, q/2\}$ for $0 \leq j \leq l_1 - 1$ then exist such that

$$s = \sum_{j=0}^{l_1 - 1} c_j \phi^j.$$

Proof. We set $s_0 = s$ and define for $j \geq 1$ the elements $s_j \in \mathrm{Z}[\phi]$ by

$$s_{j-1} = s_j \phi + c_{j-1},$$

where $c_{j-1} \in \{-q/2, \cdots, q/2\}$. By Lemma 1, such c_{j-1} always exist. If we define $\| \cdot \| = \sqrt{N(\cdot)}$, then we obtain by the triangle inequality

$$\begin{aligned} \| s_j \| &\leq \frac{\| s_{j-1} \| + \| c_{j-1} \|}{\sqrt{q}} \leq \frac{\| s_{j-1} \| + q/2}{\sqrt{q}} \\ &\leq \frac{\| s_0 \|}{q^{j/2}} + \frac{q}{2} \sum_{i=1}^{j} q^{-i/2} = \frac{\| s_0 \|}{q^{j/2}} + \frac{q(1 - q^{-j/2})}{2(\sqrt{q} - 1)} \\ &\leq \frac{\| s_0 \|}{q^{j/2}} + \sqrt{q}. \end{aligned} \tag{2}$$

Hence, if $j \geq \lceil 2 \log_q \| s_0 \| \rceil$, we have $\| s_j \| \leq (\sqrt{q} + 1)$. If we write $s_j = a + b\phi$ with $a, b \in Z$, then we have

$$\begin{aligned} N(s_j) &= a^2 + tab + qb^2 \\ &= \left(a + \frac{tb}{2} \right)^2 + \left(\frac{4q - t^2}{4} \right) b^2 \\ &= q \left(b + \frac{ta}{4q} \right)^2 + \left(\frac{4q - t^2}{4q} \right) a^2. \end{aligned}$$

Assuming that E is not supersingular, we obtain $4q - t^2 \geq 3$. Since $\| s_j \| \leq \sqrt{q} + 1$ and $q \geq 64$, we have

$$\begin{aligned} |b| &\leq \frac{2}{\sqrt{3}} (\sqrt{q} + 1) \leq \frac{q}{2}, \\ |a| &\leq \frac{2}{\sqrt{3}} (q + \sqrt{q}) \leq \frac{3q}{2}, \\ |b \pm t| &\leq |b| + |t| \leq \left(2 + \frac{2}{\sqrt{3}} \right) \sqrt{q} + \frac{4}{\sqrt{3}} < \frac{q}{2}. \end{aligned}$$

If $|a| \leq q/2$, then $a+b\phi$ is itself a valid expansion of length two. Otherwise, we can obtain a valid expansion of length three for $a > q/2$, which can be written as

$$a + b\phi = (a - q) + (b + t)\phi - \phi^2,$$

and the case $a < -q/2$ can be treated symmetrically.

Now, assume that $|t| \leq \sqrt{q}$. If $j \geq \lceil 2 \log_q \| s_0 \| \rceil - 1$ in (2), then $\| s_j \| \leq 2\sqrt{q}$, and we have

$$\begin{aligned} |b| &\leq \frac{2}{\sqrt{3q}} (2\sqrt{q}) < 3, \\ |a| &\leq \frac{2}{\sqrt{3q}} (2q) \leq \frac{q}{2}. \end{aligned}$$

Therefore, $a + b\phi$ is itself a valid expansion of length two. □

According to Lemma 2, the length of the Frobenius expansion of an integer k is about $2 | \log_q k |$. As in [4], we can reduce the expansion length by replacing the multiple kP

with $k'P$, where $k'=k \bmod (\alpha^n - 1)$, since $\phi^n(P)=P$. Moreover, assuming that $P \in E(\mathrm{F}_{q^n}) \setminus E(\mathrm{F}_q)$, kP can be computed as $k''P$, where $k''= k \bmod (\alpha^{n-1} + \alpha^{n-2} + \cdots + 1)$.

Actually, the norms of $\phi^n - 1$ and $(\phi^{n-1} + \phi^{n-2} + \cdots + 1)$ are $\#E(\mathrm{F}_{q^n})$ and $\#E(\mathrm{F}_{q^n})/\#E(\mathrm{F}_q)$. Therefore, by the reduction step, the expansion length is reduced by about half. $(\alpha^{n-1} + \alpha^{n-2} + \cdots + 1)$ can be represented as $r + s\alpha$ using the following Lucas sequence:

$$\begin{aligned} &U_0 = 0, \qquad U_1 = 1, \\ &U_i = tU_{i-1} - qU_{i-2} \text{ for } i \ge 2, \\ &\alpha^i = U_i \alpha - qU_{i-1}. \end{aligned}$$

Note that r and s can be pre-computed and stored.

Lemma 3. For $k \in \mathrm{Z}$, $w + z\alpha$ and $x + y\alpha \in \mathrm{Z}[\alpha]$ exist such that

$$\begin{aligned} &k = (w + z\alpha)(r + s\alpha) + (x + y\alpha), \\ &\| x + y\alpha \| \le \left(\frac{1+\sqrt{q}}{2} \right) \| r + s\alpha \|, \end{aligned}$$

where $\| a + b\alpha \| = \sqrt{a^2 + tab + qb^2}$.

Proof. Let $k' = k/(r + s\alpha) = w' + z'\alpha \in \mathrm{R}[\alpha]$. Then we can write

$$k = (rw' - sqz') + (sw' + (r + st)z')\alpha.$$

Therefore,

$$\begin{aligned} &w' = (r + st)k/(r^2 + trs + qs^2), \\ &z' = -sk/(r^2 + trs + qs^2). \end{aligned}$$

Let w, z be the integers nearest to w', z', respectively. Then we have

$$k = (w + z\alpha)(r + s\alpha) + [(w' - w) + (z' - z)\alpha](r + s\alpha).$$

Let $(x + y\alpha) = k - (w + z\alpha)(r + s\alpha)$. Then we have

$$\begin{aligned} &x = k - rw + qsz, \\ &y = -sw - (r + ts)z, \end{aligned}$$

and

$$\begin{aligned} \| x + y\alpha \| &= \| (w' - w) + (z' - z)\alpha \| \, \| r + s\alpha \| \\ &\le \frac{1}{2}(1 + \sqrt{q}) \| r + s\alpha \|. \end{aligned}$$

This completes the proof. □

Algorithm 1 gives the procedure of the Frobenius expansion for an integer k.

Algorithm 1. Frobenius expansion of k

```
Input: Integer $k$ and integers $r$ and $s$ such that
       $r + s\alpha = \alpha^{n-1} + \alpha^{n-2} + \cdots + \alpha + 1$
Output: Frobenius expansion $\sum c_j \phi^j$ of $k$
 1: $g = (r + ts)k,\ h = -sk$
 2: $N = r^2 + trs + qs^2$
    {$N$ is the norm of $r + s\alpha$, that is, $\#E(\mathrm{F}_{q^n})/\#E(\mathrm{F}_q)$}
 3: $w = \mathrm{Round}(g/N),\ z = \mathrm{Round}(h/N)$
 4: $x = k - rw + qsz,\ y = -sw - (r + ts)z$
    {$k = x + y\alpha \bmod (r + s\alpha)$}
 5: $C = \langle\ \rangle$
 6: while $x \ne 0 \vee y \ne 0$ do
 7:     $u = x$ MOD $q$ such that $u \equiv x \pmod{q}$ and
        $-q/2 < u \le q/2$
 8:     $v = (x - u)/q$
 9:     $x = tv + y$
10:     $y = -v$
11:     Prepend $u$ to $C$
12: end while
13: Return $C = \langle c_{l_1-1}, c_{l_1-2}, \cdots, c_1, c_0 \rangle$
```

Proposition 1. The expansion length l_1 of integer k by Algorithm 1 satisfies

$$l_1 \le \begin{cases} n+1 & \text{if } t \le \sqrt{q}, \\ n+3 & \text{otherwise}. \end{cases}$$

Proof. For $q \ge 64$, we have

$$\#E(\mathrm{F}_{q^n}) < \frac{4q^n}{3} \ and \ \frac{(1+\sqrt{q})^2}{4\#E(\mathrm{F}_q)} < \frac{3}{4}.$$

By Lemma 3,

$$\begin{aligned} \left\lceil \log_q (N(x + y\phi)) \right\rceil &\le \left\lceil \log_q \left(\frac{(1+\sqrt{q})^2}{4} N(r + s\phi) \right) \right\rceil \\ &= \left\lceil \log_q \left(\frac{(1+\sqrt{q})^2}{4} \frac{\#E(\mathrm{F}_{q^n})}{\#E(\mathrm{F}_q)} \right) \right\rceil \\ &\le n. \end{aligned}$$

Therefore, the proof is completed by Lemma 2. □

III. Scalar Multiplication Using Frobenius Expansions

In this section, we assume that the previous Frobenius expansion created by Algorithm 1 will be used. Though Muller's original method does not utilize the reduction step which divides k by $r+s\alpha$, every scalar multiplication method using the Frobenius expansion can be sped up by the reduction.

1. Previous Methods

A. Muller's Method

Muller [5] proposed an algorithm that uses a reference table. The algorithm computes and stores jP for the range of $1 \le j \le q/2$ found in the table, which is used to proceed with the following scalar multiplication:

$$\begin{aligned} kP &= \left(\sum_{j=0}^{l_1-1} c_j \phi^j\right)(P) \\ &= \phi(\cdots\phi(\phi(c_{l_1-1}P) + c_{l_1-2}P)\cdots + c_1P) + c_0P. \end{aligned}$$

As m becomes larger, the cost of table computation and the size of the table grow exponentially. Consequently, this method is applicable only to a small-sized m.

B. Cheon and Others Method

Cheon and others [6] improved on Muller's method by applying BGMW's idea [13]. This method computes and stores $\phi^j(P)$ for $0 \le j \le l_1 - 1$ and proceeds with the following scalar multiplication:

$$\begin{aligned} kP &= \left(\sum_{j=0}^{l_1-1} c_j \phi^j\right)(P) \\ &= \sum_{d=1}^{q/2} dT_d \end{aligned}$$

where $T_d = \sum_{|c_j|=d} \varepsilon_j \phi^j(P),\ \varepsilon_j = c_j / |c_j|.$

This method is also applicable only when m is small because the loop (steps 3 thru 7 in Algorithm 2) needs at least $q/2$ elliptic curve additions.

Algorithm 2. Scalar multiplication (Cheon and others)

Input: Frobenius expansion $\sum c_j \phi^j$ of k and an element $P \in E$

Output: kP

```
1:  Compute and store $\phi^j(P)$ for $0 \le j \le l_1 - 1$
2:  $T = O,\ Q = O$
3:  for $i = q/2$ to 1 by $-1$ do
4:      for each $j$ such that $c_j = i$, set $T = T + \phi^j(P)$
5:      for each $j$ such that $c_j = -i$, set $T = T - \phi^j(P)$
6:      $Q = Q + T$
7:  end for
8:  Return $Q$
```

C. Multiple-Base Binary Method (MBB)

The binary method was originally applied to a single base point. However, it can be modified to support multiple base points and has been applied to the scalar multiplication of elliptic curves defined over an Optimal Extension Field of odd characteristic [11].

Let $P_j = \phi^j(P)$ if $c_j \ge 0$; otherwise $P_j = -\phi^j(P)$. Let $(c_{j,m-1}, c_{j,m-2}, \cdots, c_{j,0})_2$ be the binary representation of $|c_j|$ and

$$T_i = \sum_{j=0}^{l_1-1} c_{j,i} P_j.$$

kP can then be computed using $kP = \sum_{i=0}^{m-1} 2^i T_i$.

Algorithm 3. Scalar multiplication (MBB method)

Input: Frobenius expansion $\sum c_j \phi^j$ of k and an element $P \in E$

Output: kP

```
1: Compute and store $P_j$ for $0 \le j \le l_1 - 1$
     $P_j = \phi^j(P)$ if $c_j \ge 0$,
     $P_j = -\phi^j(P)$ otherwise
2:  $T = O$
3:  for $i = m-1$ to 0 by $-1$ do
4:      $T = 2T$
5:      for each $i$ such that $c_{j,i} = 1$, set $T = T + P_j$
6:  end for
7:  Return $T$
```

2. Proposed Methods

We propose two variants of the MBB method by combining it with the window method. The first one is suitable for the normal basis representation. It reduces elliptic curve addition by increasing the number of Frobenius map operations, which are almost free in the normal basis representation. The second one is suitable for the ordinary polynomial basis representation.

A. Proposed Method 1

In this method, we will find a Frobenius expansion of an integer with the coefficient c_j contained in the interval $[0, q-1]$. If we expand the scalar in this way using Algorithm 1, the process may not end in the case of $t > 0$. Therefore, we assume $t < 0$. In this way, we have the following upper bound on the expansion length.

Lemma 4. Assume that $t < 0$ and $q \geq 32$. Denote by N the norm function from $Z[\phi]$ to Z given by $N(a+b\phi) = a^2 + tab + qb^2$ for $a, b \in Z$. Given $s \in Z[\phi]$, set

$$l_2 = \begin{cases} \lceil \log_q(N(s)) \rceil + 3 & \text{if } -\sqrt{q} \leq t < 0, \\ \lceil \log_q(N(s)) \rceil + 5 & \text{otherwise.} \end{cases}$$

An integer $c_j \in \{0, \cdots, q-1\}$, $0 \leq j \leq l_2 - 1$, then exists such that

$$s = \sum_{j=0}^{l_2-1} c_j \phi^j.$$

Proof. Similar to the proof of Lemma 2, we set $s_0 = s$ and define for $j \geq 0$ the elements $s_j \in Z[\phi]$ by

$$s_{j-1} = s_j \phi + c_{j-1}.$$

If we define $\|\cdot\| = \sqrt{N(\cdot)}$, then by the triangle inequality we obtain

$$\begin{aligned} \| s_j \| &\leq \frac{\| s_{j-1} \| + \| c_{j-1} \|}{\sqrt{q}} \leq \frac{\| s_{j-1} \| + (q-1)}{\sqrt{q}} \\ &\leq \frac{\| s_0 \|}{q^{j/2}} + (q-1) \sum_{i=1}^{j} q^{-i/2} \\ &\leq \frac{\| s_0 \|}{q^{j/2}} + \sqrt{q} + 1. \end{aligned} \tag{3}$$

If $j \geq \lceil 2 \log_q \| s_0 \| \rceil$ in (3), then $\| s_j \| \leq (\sqrt{q} + 2)$. If we write $s_j = a + b\phi$ with $a, b \in Z$, then we have

$$\begin{aligned} N(s_j) &= a^2 + tab + qb^2 \\ &= \left(a + \frac{tb}{2} \right)^2 + \left(\frac{4q - t^2}{4} \right) b^2 \\ &= q\left(b + \frac{ta}{4q} \right)^2 + \left(\frac{4q - t^2}{4q} \right) a^2. \end{aligned}$$

Assuming that E is not supersingular, we obtain $4q - t^2 \geq 3$. Since $\| s_j \| \leq \sqrt{q} + 2$ and $q \geq 32$, we have

$$\begin{aligned} | b | &\leq \frac{2}{\sqrt{3}} (\sqrt{q} + 2) < q, \\ | a | &\leq \frac{2}{\sqrt{3}} (q + 2\sqrt{q}) < 2q, \\ | b \pm 2t | &\leq | b | + 2| t | \leq \left(4 + \frac{2}{\sqrt{3}} \right) \sqrt{q} + \frac{4}{\sqrt{3}} \leq q. \end{aligned}$$

Case 1 $(a \geq q)$. We can write $(a + b\phi) = (a - q) + (b + t)\phi - \phi^2$. If $(b + t) < 0$,

$$a + b\phi = (a - q) + (q - (b + t))\phi + (-t - 1)\phi^2 + \phi^3.$$

Otherwise,

$$a + b\phi = (a - q) + (b + t)\phi + (q - 1)\phi^2 + (-t)\phi^3 + \phi^4.$$

Case 2 $(0 \leq a < q)$. If $b \geq 0$, then $a + b\phi$ is already the desired form; otherwise, we can write

$$a + b\phi = a + (q - b)\phi - t\phi^2 + \phi^3.$$

Case 3 $(-q \leq a < 0)$. We can write $a + b\phi = (a + q) + (b - t)\phi + \phi^2$. If $b - t > 0$, then the above formula is the desired form; otherwise,

$$a + b\phi = (a + q) + (q + (b - t))\phi + (1 - t)\phi^2 + \phi^3.$$

Case 4 $(-2q < a < -q)$. We can write $a + b\phi = (a + 2q) + (b - 2t)\phi + 2\phi^2$. If $b - 2t > 0$, then the above formula is the desired form; otherwise,

$$a + b\phi = (a + q) + (q + (b - 2t))\phi + (2 - t)\phi^2 + \phi^3.$$

Therefore, $s_j = a + b\phi$ has a Frobenius expansion length of 5 at most.

Assume that $-q \leq t < 0$. If $j \geq \lceil 2 \log_q \| s_0 \| \rceil - 1$ in (3), then $\| s_j \| \leq 2\sqrt{q} + 1$, and we have

$$\begin{aligned} | b | &\leq \frac{2}{\sqrt{3}} \left(2 + \frac{1}{\sqrt{q}} \right) < 3, \\ | a | &\leq \frac{2}{\sqrt{3}} \left(2\sqrt{q} + 1 \right) < q. \end{aligned}$$

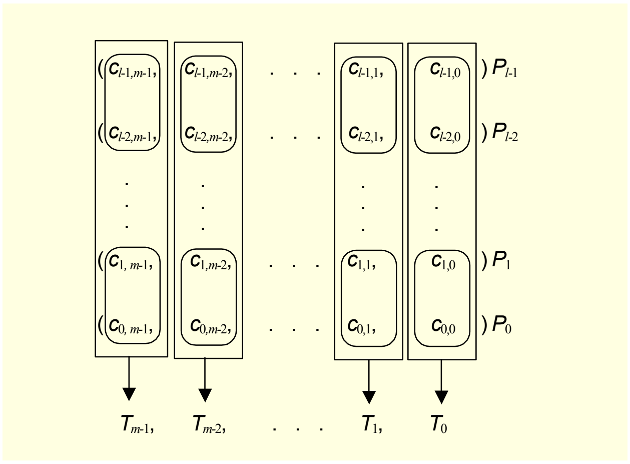
Hence, $s_j = a + b\phi$ has a Frobenius expansion length of 4 at most in a manner similar to the above cases. Therefore, the lemma is proved. □

Let $S_a = a_{w-1}\phi^{w-1}(P) + \cdots + a_0 P$ for $a = (a_{w-1}, \cdots, a_0) \in \{0, 1\}^w$ and

$$T_j = \sum_{i=0}^{\lceil l_2/w \rceil - 1} \phi^{wi} S_{(c_{wi+w-1,j},\cdots,c_{wi,j})}$$

for $0 \le j \le m-1$. Then, as shown in Fig. 1, kP can be computed as follows:

$$kP = \sum_{j=0}^{m-1} 2^j T_j.$$



$(c_{l-1,m-1},$ $c_{l-1,m-2},$ . . . $c_{l-1,1},$ $c_{l-1,0}$ $)\,P_{l-1}$
$(c_{l-2,m-1},$ $c_{l-2,m-2},$ . . . $c_{l-2,1},$ $c_{l-2,0}$ $)\,P_{l-2}$
$(c_{1,m-1},$ $c_{1,m-2},$ . . . $c_{1,1},$ $c_{1,0}$ $)\,P_1$
$(c_{0,m-1},$ $c_{0,m-2},$ . . . $c_{0,1},$ $c_{0,0}$ $)\,P_0$
$T_{m-1},$ $T_{m-2},$ . . . $T_1,$ T_0

Fig. 1. A variant of the multiple-base binary method (proposed method 1).

Algorithm 4. Scalar multiplication (proposed method 1)

Input: $k = \sum c_j \phi^j$ for $(0 \le c_j < q)$, P

Output: kP

```
1:   Compute and store $S_a = a_{w-1}\phi^{w-1}(P) + \cdots + a_0 P$
     for all $(a_{w-1},\cdots,a_0) \in \{0,1\}^w$
2:   $Q = O$
3:   for $j = m-1$ to 0 by $-1$ do
4:       $Q = 2Q,\ T = O$
5:       for $i = \lceil l_2/w \rceil - 1$ to 0 by $-1$ do
6:           $T = \phi^w(T)$
7:           $a = (c_{wi+w-1,j},\cdots,c_{wi,j})$
8:           $T = T + S_a$
9:       end for
10:      $Q = Q + T$
11:  end for
12:  Return $Q$
```

B. Proposed Method 2

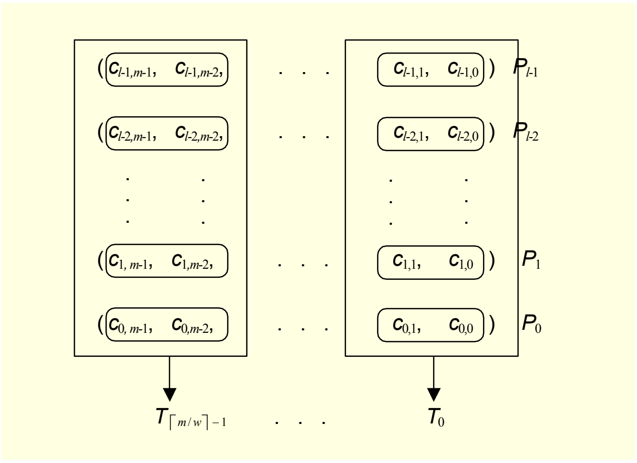
In this method, we use the usual Frobenius expansion method (Algorithm 1). Let $(c_{j,m-1}, c_{j,m-2}, \cdots, c_{j,1}, c_{j,0})_2$ be a binary representation of $|c_j|$. Since $|c_j| \le q/2$, $c_{j,m-1} = 1$ holds if and only if $|c_j| = q/2$.

Let $\varepsilon_j = c_j / |c_j|$ if $c_j \ne 0$; otherwise, $\varepsilon_j = 0$. Let $S_a = a_{w-1}2^{w-1}(P) + \cdots + a_0 P$ for $a = (a_{w-1},\cdots,a_0) \in \{0,1\}^w$ and

$$T_i = \sum_{j=0}^{l_1 - 1} \varepsilon_j \phi^j S_{(c_{j,wi+w-1},\cdots,c_{j,wi})}$$

for $0 \le i \le \lceil m/w \rceil - 1$. As shown in Fig. 2, kP can then be computed as follows:

$$kP = \sum_{i=0}^{\lceil m/w \rceil - 1} 2^{wi} T_i.$$



$(c_{l-1,m-1},$ $c_{l-1,m-2},$ . . . $c_{l-1,1},$ $c_{l-1,0})$ P_{l-1}
$(c_{l-2,m-1},$ $c_{l-2,m-2},$ . . . $c_{l-2,1},$ $c_{l-2,0})$ P_{l-2}
$(c_{1,m-1},$ $c_{1,m-2},$ . . . $c_{1,1},$ $c_{1,0})$ P_1
$(c_{0,m-1},$ $c_{0,m-2},$ . . . $c_{0,1},$ $c_{0,0})$ P_0
$T_{\lceil m/w \rceil - 1}$ . . . T_0

Fig. 2. A variant of the multiple-base binary method (proposed method 2).

Algorithm 5. Scalar multiplication (proposed method 2)

Input: $k = \sum c_j \phi^j$, P

Output: kP

```
1:   Compute and store $S_a = a_{w-1}2^{w-1}(P) + \cdots + a_0 P$
     for all $(a_{w-1},\cdots,a_0) \in \{0,1\}^w$
2:   $Q = O$
3:   for $i = \lceil m/w \rceil - 1$ to 0 by $-1$ do
4:       $Q = 2^w Q,\ T = O$
5:       for $j = l_1 - 1$ to 0 by $-1$ do
6:           $T = \phi(T)$
7:           $a = (c_{j,wi+w-1},\cdots,c_{j,wi})$
8:           If $c_j > 0$ then $T = T + S_a$ else $T = T - S_a$
9:       end for
10:      $Q = Q + T$
11:  end for
12:  Return $Q$
```

We note that the probability of $c_{j,m-1} = 1$ is 2/q. Therefore, the average number of iterations for the loop can be reduced to $\lceil (m-1)/w \rceil$.

C. Comparison

We used Lopez and Dahab's projective coordinate systems, shown in Table 1, assuming the bit length of k is $mn-m$. In Tables 2 and 3, we compare the number of basic elliptic curve operations and field operations needed for each scalar multiplication method, assuming $w=4$.

In proposal 1, we use an affine coordinate for computing S_a in step 8 of Algorithm 4 so that we can use *ADD1* operations.

We give three concrete examples. Their security levels are almost equivalent to the ordinary curves defined over 160-, 192-, and 256-bit fields, respectively. We have selected example curves to create the minimal cofactors as follows:

$$E_1(\mathrm{F}_{2^{11\cdot 17}}): y^2+xy=x^3+b_1, \quad (t=-47)$$
$$E_2(\mathrm{F}_{2^{17\cdot 13}}): y^2+xy=x^3+b_2, \quad (t=-107)$$
$$E_3(\mathrm{F}_{2^{16\cdot 17}}): y^2+xy=x^3+b_3, \quad (t=-95),$$

where

b_1 = 0684488E 0146CB9F 1517781B 9F8D3381 0C2AB22E F99F897E
b_2 = 16E6CBDE 394F95BC 234A53FF 0672DAFA A04BCF1E 3398FBAF 3F122289
b_3 = 00009B68 7AC2E69E 134B9992 AAA3A99F 675CC5A8 CF7BAA9B 7F2B7B98 AD243268 E3B4D9B9,

each with the polynomial basis such that

$$\mathrm{F}_{2^{11\cdot 17}}=\mathrm{F}_2[x]/(x^{187}+x^7+x^6+x^5+1),$$
$$\mathrm{F}_{2^{17\cdot 13}}=\mathrm{F}_2[x]/(x^{221}+x^8+x^6+x^2+1),$$
$$\mathrm{F}_{2^{16\cdot 17}}=\mathrm{F}_2[x]/(x^{272}+x^9+x^3+x^2+1).$$

We implemented each method on a Pentium IV/2.4 GHz computer running Windows 2000 and using MSVC 6.0 as a compiler. Table 5 shows the speed of the field operations and scalar multiplications described in the previous section. We used the left-to-right comb method with a window size of 4

Table 2. The number of basic elliptic curve operations for each scalar multiplication method (A: affine, P: projective).

Operation	Cheon	MBB	Proposal 1	Proposal 2
ADD (A)	-	-	$2^{w-1}-1$	$2^{w-1}-1$
DBL (A)	-	-	-	$2^{w-1}-1$
ϕ(A)	l_1-1	l_1-1	$2^{w-1}-1$	-
ADD1(P)	l_1	$ml_1/2$	$m(\lceil l_2/w\rceil-1)$	$(l_1-1)\lceil(m-1)/w\rceil$
ADD2(P)	2^{m-1}	-	$m-1$	$\lceil(m-1)/w\rceil-1$
DBL (P)	-	$m-1$	$m-1$	$w(\lceil(m-1)/w\rceil-1)$
ϕ(P)	-	-	$m(\lceil l_2/w\rceil-1)w$	$(l_1-1)\lceil(m-1)/w\rceil$
P→A	1	1	1	1

Table 3. The number of field operations for each scalar multiplication method.

Operation	Inv.	Mul.	Sqr.
Montgomery	1	$6nm-6m+4$	$5nm-5m$
Cheon	1	$9n+13\cdot 2^{m-1}+29$	$4n+6\cdot 2^{m-1}+2m(n+2)+13$
MBB	1	$m(9n+35)/2-2$	$m(4n+15)-4$
Proposal 1	8	$9m\lceil(n+1)/4\rceil+17m-1$	$(4m+12m^2)\lceil(n+1)/4\rceil+25m-3$
Proposal 2	15	$(9n+47)\lceil(m-1)/4\rceil+1$	$(3mn+6m+4n+34)\lceil(m-1)/4\rceil-11$
MBB ($\lvert t\rvert\le\sqrt{q}$)	1	$m(9n+17)/2-2$	$m(4n+3)-4$
Proposal 1 ($\lvert t\rvert\le\sqrt{q}$)	8	$9m\lceil(n-1)/4\rceil+17m-1$	$(4m+12m^2)\lceil(n-1)/4\rceil+25m-3$
Proposal 2 ($\lvert t\rvert\le\sqrt{q}$)	15	$(9n+29)\lceil(m-1)/4\rceil+1$	$(3mn+4n+26)\lceil(m-1)/4\rceil-11$

Table 4. The number of field operations for scalar multiplication.

Operation	$m = 11, n = 17$			$m = 17, n = 13$			$m = 16, n = 17$		
	Inv.	Mul.	Sqr.	Inv.	Mul.	Sqr.	Inv.	Mul.	Sqr.
Montgomery	1	1,060	880	1	1,228	1,020	1	1,540	1,280
Cheon	1	13,494	6,643	1	852,114	393,791	1	426,166	197,297
MBB	1	1,032	909	1	1,290	1,135	1	1,502	1,324
Proposal 1	8	681	7,752	8	900	14,566	8	991	16,077
Proposal 2	15	601	2,176	15	657	3,393	15	801	4,045
MBB ($\lvert t \rvert \le \sqrt{q}$)	1	933	777	1	1,137	931	1	1,358	1,132
Proposal 1 ($\lvert t \rvert \le \sqrt{q}$)	8	582	6,256	8	747	11,030	8	847	12,941
Proposal 2 ($\lvert t \rvert \le \sqrt{q}$)	15	547	1,954	15	585	2,953	15	729	3,629

Table 5. Speed of field operations (in μs) and scalar multiplications (in ms) run on a Windows 2000 P-IV/2.4GHz computer.

	Inv.	Mul.	Qdr.	Sqr.	Mont.	MBB	Prop.1	Prop.2
E_1	18.78	1.515	0.375	0.203	1.781	1.669	2.140	**1.540**
E_2	21.22	1.912	0.500	0.266	2.714	2.488	3.525	**2.169**
E_3	35.09	2.690	0.513	0.312	4.569	4.143	5.350	**3.509**

and the extended Euclidean algorithm for field multiplication and inversion.1) We can see that Proposal 2 is about 10% faster than the conventional multiple-base binary method and about 20% faster than the Montgomery method with a polynomial basis.

Proposal 1 contains a large number of squaring so it is not suitable for the polynomial basis. But if we assume that squaring is free (e.g., hardware-based normal basis representation), we can consider implementing it.

We remark that quadrupling is more efficient than double squaring in some fields. In such cases we use quadrupling for computing ϕ.

IV. Security Consideration

The best attack known on general elliptic curve cryptosystems is the parallel collision search based on Pollard's ρ-method, where its complexity is the square root of the prime order of a base point.

However, faster attacks exist for a special family of curves such as subfield curves, supersingular curves, and anomalous curves. We can easily check whether the selected curves are supersingular or anomalous and can therefore avoid the attacks of [14]-[16]. According to [17] and [18], the attack time of the Pollard ρ-method can be reduced by a factor of $\sqrt{2n}$ for subfield curves. However, this factor is too small to influence the security of the curves, because n is less than 500. In this case, the size of $\sqrt{2n}$ is only 5 bits.

A more serious attack which utilizes the Weil descent method was proposed by Frey [19]. Gaudry, Hess and Smart [20] showed how the Weil descent can be used to reduce the elliptic curve discrete logarithm problem in E to the discrete logarithm problem in the Jacobian subgroup of a hyperelliptic curve. According to [10], the genus of the Jacobian obtained by the GHS attack is $2^{m(b)-1}$ or $2^{m(b)-1}-1$ where $m(b)$ is *the magic number* of b when $a \in \{0,1\}$.

The magic number of $b \in \mathrm{F}_{2^m}$ is at most m when we take q=2 in the GHS attack. If m is prime and 2 is primitive in F_m, then $m(b)$=m. However E_2 and E_3 do not belong to this case. A direct computation shows that the minimum magic numbers (larger than 1) of b are 11, 17, and 15. The genus of the Jacobian for these values is too large to apply the GHS attack.

1) Since our implementation is not optimal, our program's speed may be slower in general than those found in other publications, e.g. [1]. However since the speed ratio of inversion versus multiplication is similar to those of other publications, comparing the algorithms of the scalar multiplication is still meaningful.

V. Conclusion

We proposed scalar multiplication methods on elliptic curves defined over subfields. Such subfield curves have a special endomorphism called the Frobenius endomorphism, which can be utilized to speed up scalar multiplication. If the size of a subfield is too small, it is hard to find curves having good cryptographic properties such as a minimal cofactor. Therefore, it is reasonable to consider medium-sized subfields such as F_{2^m} where $10 \le m \le 20$. Though subfield curves have some minor security flaws, they are still considerable due to their efficiency.

Our method becomes more efficient in cases where the cost of squaring is very small. For example, if we can implement squaring with 1/8 or less of the cost for multiplication, our method is more efficient than the MBB method. Our implementation shows that it is about 20% faster than the Montgomery method and about 10% faster than the conventional multiple-base binary method in the polynomial basis.

References

[1] D. Hankerson, J.L. Hernandez, and A. Menezes, "Software Implementation of Elliptic Curve Cryptography over Binary Fields," *Cryptographic Hardware Embedded System* (CHES 2000), LNCS 1965, Springer-Verlag, 2000, pp. 1-24.

[2] J. Lopez and R. Dahab, "Fast Multiplication on Elliptic Curves over *GF*(2^m) without Precomputation," *Cryptographic Hardware Embedded System* (*CHES*'99), LNCS 1717, Springer-Verlag, 1999, pp. 316-327.

[3] N. Koblitz, "CM-Curves with Good Cryptographic Properties," *Advances in Cryptology – Crypto*'91, LNCS, Springer-Verlag, 1992, pp. 279-287.

[4] J. Solinas, "Improved Algorithms for Arithmetic on Anomalous Binary Curves," CACR Technical Report, 1999. This is a updated version of the paper in the proceeding of CRYPTO'97.

[5] V. Muller, "Fast Multiplication on Elliptic Curves over Small Fields of Characteristic Two," *J. of Cryptology*, vol. 11, 1998, pp. 219-234.

[6] J. Cheon, S. Park, C. Park, and S. Hahn, "Scalar Multiplication on Elliptic Curves by Frobenius Expansions," *ETRI J.*, vol. 21, no. 1, 1999, pp. 27-38.

[7] C. Paar, P. Fleishmann, and P. Soria-Rodriguez, "Fast Arithmetic for Public-Key Algorithms in Galois Fields with Composite Exponents," *IEEE Trans. on Computers*, vol. 48, no. 10, 1999, pp. 1025-1034.

[8] S. Oh, C.H. Kim, J. Lim, and D.H. Cheon, "Efficient Normal Basis Multiplier in Composite Fields" *IEEE Trans. on Computers*, vol. 49, no. 10, 2000, pp. 1133-1138.

[9] J. Guajardo and C. Paar, "Efficient Algorithms for Elliptic Curve Cryptosystems," *Advances in Cryptology – Crypto*'97, LNCS 1294, Springer-Verlag, 1997, pp. 342-356.

[10] A. Menezes and M. Qu, "Analysis of the Weil Descent Attack of Gaudry, Hess and Smart," *Topics in Cryptology – CT-RSA*, LNCS 2020, Springer-Verlag, 2001, pp. 308-318.

[11] T. Kobayashi, H. Morita, K. Kobayashi, and F. Hoshino, "Fast Elliptic Curve Algorithm Combining Frobenius and Table Reference to Adapt to Higher Characteristic," *Advances in Cryptology – Eurocrypt*'99, LNCS 1592, Springer-Verlag, 1999, pp. 176-189.

[12] J. Lopez and R. Dahab, "Improved Algorithms for Elliptic Curve Arithmetic in *GF*(2^n)," *Selected Areas on Cryptography* (*SAC*'98), LNCS 1556, Springer-Verlag, 1999, pp. 201-212.

[13] E. Brickell, D. Gordon, K. McCurley, and D. Wilson, "Fast Exponentiation with Precomputation," *Advances in Cryptology – Eurocrypt*'92, LNCS 658, Springer-Verlag, 1993, pp. 200-207.

[14] A. Menezes, T. Okamoto, and S. Vanstone, "Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field," *IEEE Trans. on Information Theory*, vol. 39, 1993, pp. 1639-1646.

[15] G. Frey and H. Ruck, "A Remark Concerning m-Divisibility and the Discrete Logarithm in the Divisor Class Group of Curves," *Math. Comp.*, vol. 62, 1994, pp. 865-874.

[16] N. Smart, "The Discrete Logarithm Problem on Elliptic Curves of Trace One," *J. of Cryptology*, vol. 12, no. 4, 1999, pp. 193-196.

[17] I. Duursma, P. Gaudry and F. Morain, "Speeding Up the Discrete Log Computation on Curves with Automorphisms," *Advances in Cryptology – Asiacrypt*'99, LNCS 1716, Springer-Verlag, 1999, pp. 103-121.

[18] M.J. Wiener and R.J. Zuccherato, "Faster Attacks on Elliptic Curve Cryptosystems," *Selected Areas on Cryptography* (*SAC*'98), LNCS 1556, Springer-Verlag, 1999, pp. 190-200.

[19] G. Frey, "How to Disguise an Elliptic Curve (Weil Descent)," Talk at *ECC*'98, Waterloo, 1998.

[20] P. Gaudry, F. Hess, and N. Smart, "Constructive and Destructive Facets of Weil Descent on Elliptic Curves," *J. of Cryptology*, vol. 15, no. 1, 2002, pp. 19-46.

Dong Hoon Lee received the BS degree in mathematical education from Seoul National University in 1994. He also received the MS and PhD degrees in mathematics from Korea Advanced Institute of Science and Technology (KAIST) in 1996 and 2000. From 2000 to 2002, he worked at Cryptography & Network Security Center of Future Systems Inc. Since 2002 he has been with National Security Research Institute (NSRI) as a Senior Research Member. His interests include number theory, cryptography, and cryptanalysis.

Seongtaek Chee received the PhD degree in mathematics from Korea University, Seoul, Korea in 1998. Since 1989 he has been on the Research Staff at ETRI, where he is currently a Principal Member of NSRI. His research interests are in cryptographic functions.

Sang Cheol Hwang received the BS and the MS degrees in electronics engineering from Soongsil University in 1985 and 1988. He is currently a PhD student at Chungnam National University. His research interests include internet security, electronic commerce, and protocols.

Jae-Cheol Ryou received the BS degree in industrial engineering from Hanyang University in 1985, the MS degree in computer science from Iowa State University in 1988, and the PhD degree in electrical engineering and computer science from Northwestern University in 1990. He joined the faculty of the Department of Computer Science at Chungnam National University, Korea, in 1991. His research interests are internet security and electronic payment systems. He is currently with the Internet Intrusion Response Technology Research Center (IIRTRC), Chungnam National University, Korea.