PAPER *Special Section on Cryptography and Information Security*

Some Efficient Algorithms for the Final Exponentiation of η_T Pairing

Masaaki SHIRASE[†a)], **Tsuyoshi TAKAGI**[†b)], *and* **Eiji OKAMOTO**[††c)], *Members*

SUMMARY Recently Tate pairing and its variations are attracted in cryptography. Their operations consist of a main iteration loop and a final exponentiation. The final exponentiation is necessary for generating a unique value of the bilinear pairing in the extension fields. The speed of the main loop has become fast by the recent improvements, e.g., the Duursma-Lee algorithm and η_T pairing. In this paper we discuss how to enhance the speed of the final exponentiation of the η_T pairing in the extension field $\mathbb{F}_{3^{6n}}$. Indeed, we propose some efficient algorithms using the torus $T_2(\mathbb{F}_{3^{3n}})$ that can efficiently compute an inversion and a powering by $3^n + 1$. Consequently, the total processing cost of computing the η_T pairing can be reduced by 16% for $n = 97$.

key words: *Tate pairing, η_T pairing, final exponentiation, torus*

1. Introduction

Bilinear pairings deliver us new cryptographic applications such as identity-based encryptions [4], short signatures [6], and efficient broadcast encryptions [5]. Recently Duursma and Lee [7] proposed an efficient algorithm for computing Tate pairing. The Duursma-Lee algorithm uses the supersingular curves,

$$E^b(\mathbb{F}_{3^n}) : y^2 = x^3 - x + b \text{ with } b \in \{-1, 1\}. \qquad (1)$$

Kwon proposed an efficient variation of the Duursma-Lee algorithm that requires no cube root operation [11]. Barreto et. al. proposed the η_T pairing [2], which reduces the number of the main loop in the Duursma-Lee algorithm to half. Beuchat et. al. presented a faster variation of η_T pairing without a cube root operation [3]. Currently the η_T pairing is one of the fastest algorithms for computing the bilinear pairing.

Both the Duursma-Lee algorithm and the η_T pairing require the "final exponentiation," i.e., A^s for $A \in \mathbb{F}_{3^{6n}}$ and some integer s, since the resulting element by the pairing algorithms is contained in the quotient group $\mathbb{F}^*_{3^{6n}}/\mathbb{F}^*_{3^{3n}}$. The final exponentiations for the Duursma-Lee algorithm and the η_T pairing are $A^{3^{3n}-1}$ and A^W with $W = (3^{3n} - 1)(3^n + 1)(3^n + 1 - b3^{(n+1)/2})$, respectively. The η_T pairing without the final exponentiation is about twice faster than the Duursma-Lee algorithm, but the final exponentiation in the η_T pairing causes a relatively large overhead. For example, Shu et. al. [13] estimated that the η_T pairing with the final exponentiation is as fast as the Duursma-Lee algorithm in hardware. Ronan et. al. reported that the straightforward implementation of the final exponentiation is more than 35% of the whole algorithm [12]. In Sect. 4, we estimated that the currently fastest final exponentiation [2] is about 25% of the whole algorithm.

In this paper we try to reduce the cost of the final exponentiation of the η_T pairing. Barreto et. al. proposed an efficient calculation for the final exponentiation using Frobenius mapping [1]. We propose that we use not Frobenius but also Torus T_2 for it. Note that $A^{3^{3n}-1}$ is an element in the torus $T_2(\mathbb{F}_{3^{3n}})$, which is a subgroup of $\mathbb{F}^*_{3^{6n}}$. We show that an inversion and a powering by $(3^n + 1)$-th in $T_2(\mathbb{F}_{3^{3n}})$ are efficiently computed for the basis $\{1, \sigma\}$ of $\mathbb{F}_{3^{6n}}$ over $\mathbb{F}_{3^{3n}}$ with $\sigma^2 + 1 = 0$. We then present an efficient algorithm for the final exponentiation $A^W = B^{(3^n+1)(3^n+1-b3^{(n+1)/2})}$ with $B = A^{3^{3n}-1}$ of the η_T pairing in the torus $T_2(\mathbb{F}_{3^{3n}})$, which can be computed with 36 multiplications in $\mathbb{F}_{3^n}$ plus other negligible operations. Consequently, the final exponentiation of our proposed scheme requires only about 13% of the whole η_T pairing, which achieves about 16% faster η_T pairing than the previous known algorithms.

On the other hand, Granger et. al. presented an encoding method of $\mathbb{F}^*_{3^{6n}}/\mathbb{F}^*_{3^{3n}}$ [9], which eliminates the final exponentiation from the Duursma-Lee algorithm. We call it the GPS encoding according to the authors' name. In this paper, we discuss how to apply the GPS encoding to the η_T pairing. The η_T pairing with the GPS encoding can be faster depending on the information of b.

The remainder of this paper is organized as follows: In Sect. 2 we explain Tate pairing and the η_T pairing. In Sect. 3 we describe several representations (including the Torus $T_2(\mathbb{F}_{3^{3n}})$ and the GPS encoding) of group $\mathbb{F}^*_{3^{6n}}$ and apply them to the efficient computation of the final exponentiation for the Duursma-Lee algorithm. In Sect. 4 we propose new efficient algorithms of computing the final exponentiation for the η_T pairing and how to apply the GPS encoding to the η_T pairing. In Sect. 5 we conclude this paper.

Manuscript received March 23, 2007.
Manuscript revised July 13, 2007.

[†] The authors are with Future University-Hakodate, Hakodate-shi, 041-8655 Japan.

[††] The author is with University of Tsukuba, Tsukuba-shi, 305-8571 Japan.

a) E-mail: shirase@fun.ac.jp
b) E-mail: takagi@fun.ac.jp
c) E-mail: okamoto@risk.tsukuba.ac.jp

DOI: 10.1093/ietfec/e91–a.1.221

2. Tate Pairing and η_T Pairing

In this section we explain about Tate pairing and its efficient variations, namely the Duursma-Lee algorithm and the η_T pairing.

2.1 Tate Pairing

Let $\mathbb{F}_q$ be a finite field with q elements, where q be a power of the characteristic p. Let E be elliptic curves defined over $\mathbb{F}_q$, and let O_E be the point at infinity. Let l be a positive integer relatively prime to q with $l \,|\, \#E(\mathbb{F}_q)$, and let k be the minimal positive integer with $l \,|\, (q^k - 1)$. This k is called the embedded degree. Then Tate pairing is a map

$$\langle \cdot, \cdot \rangle_l : E(\mathbb{F}_q)[l] \times E(\mathbb{F}_{q^k})/lE(\mathbb{F}_{q^k}) \to \mathbb{F}^*_{q^k}/(\mathbb{F}^*_{q^k})^l,$$

which satisfies the bilinearity $\langle P, aQ \rangle_l = \langle aP, Q \rangle_l = \langle P, Q \rangle_l^a$ for any integer $a \neq 0$, and is non-degenerate, i.e., there exists a $Q \in E(\mathbb{F}_{q^k})$ such that $\langle P, Q \rangle_l \notin (\mathbb{F}^*_{q^k})^l$ for $P \in E(\mathbb{F}_{q^k})[l] \setminus \{O_E\}$.

It is typically selected that l and q^k are about 160 bits and 1024 bits, respectively. One of the most efficient classes for computing the bilinear map is constructed over supersingular elliptic curves. The embedded degree k of supersingular elliptic curves is one of 4, 6, or 2 for characteristic 2, 3 or $p > 3$, respectively. This paper deals with the case of characteristic 3 and uses elliptic curves formed by (1). It is known that $\#E^b(\mathbb{F}_{3^n}) = 3^n + 1 + b' 3^{(n+1)/2}$, where b' is defined as

$$b' = \begin{cases} b & \text{if } n \equiv 1, 11 \pmod{12}, \\ -b & \text{if } n \equiv 5, 7 \pmod{12}. \end{cases}$$

Note that we have $n \equiv 1, 5, 7, 11 \pmod{12}$ since n has to be coprime to 6 [7].

We require an injection ψ from $E(\mathbb{F}_{3^n})[l]$ to $E(\mathbb{F}_{3^{6n}})/lE(\mathbb{F}_{3^{6n}})$ since $\langle P, Q \rangle_l$ is defined for points $P \in E(\mathbb{F}_{3^n})[l]$ and $Q \in E(\mathbb{F}_{3^{6n}})/lE(\mathbb{F}_{3^{6n}})$. This ψ is sometimes called as the distortion map. In the case of characteristic three, the distortion map is defined as $\psi(x, y) = (-x + \rho, y\,\sigma)$ for $(x, y) \in E(\mathbb{F}_{3^n})$, where σ and ρ satisfy

$$\sigma^2 = -1 \text{ and } \rho^3 = \rho + b.$$

We usually select the basis $\{1, \sigma, \rho, \sigma\rho, \rho^2, \sigma\rho^2\}$ of $\mathbb{F}_{3^{6n}}$ over $\mathbb{F}_{3^n}$, where ρ and σ are utilized in the distortion map. Every element A in $\mathbb{F}_{3^{6n}}$ is then represented as $A = a_0 + a_1\sigma + a_2\rho + a_3\sigma\rho + a_4\rho^2 + a_5\sigma\rho^2$ for some $a_i \in \mathbb{F}_{3^n}$. Moreover an element A_0 in $\mathbb{F}_{3^{3n}}$ is represented as $A_0 = a_0 + a_2\rho + a_4\rho^2$. We denote by M_k, C_k and I_k the computational cost of multiplication, cubing, and inversion in $\mathbb{F}_{3^{kn}}$, respectively. Then the following relationships

$$\begin{gathered} M_6 = 3M_3,\ M_3 = 6M_1,\ C_6 = 2C_3,\ C_3 = 3C_1, \\ I_6 = 5M_3 + I_3,\ I_3 = 8M_1 + I_1 \end{gathered} \tag{2}$$

are held [10]. The computational costs appeared in this paper are estimated using Eq. (2). The computational cost is estimated without considering the costs of addition and subtraction which are usually negligible. Beuchat et. al. pointed out A^{3^n} in $\mathbb{F}_{3^{6n}}$ can be computed virtually for free [3] (see Appendix B). Therefore we have

$$\text{the cost of } A^{3^n+1} (= A^{3^n} \cdot A) \text{ is } M_6 = 18M_1. \tag{3}$$

The resulting value $\langle P, \psi(Q) \rangle_l$ of Tate pairing is contained in the quotient group $\mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^l$. Then there are many choices for representing elements in a coset of the quotient group. Indeed $A, B \in \mathbb{F}^*_{3^{6n}}$ are contained in the same coset, if they satisfies $B = A \cdot C^l$ for some $C \in \mathbb{F}^*_{3^{6n}}$. We are able to eliminate this *ambiguity* by using the final exponentiation. The final exponentiation tries to compute the $((3^{6n} - 1)/l)$-th powering to the output from the Tate pairing. Therefore we also deploy the modified Tate pairing $\hat{e}(P, Q)$ defined by $\hat{e} : E(\mathbb{F}_{3^n})[l] \times E(\mathbb{F}_{3^n})[l] \to \mathbb{F}^*_{3^{6n}}$, $(P, Q) \mapsto \hat{e}(P, Q) = \langle P, \psi(Q) \rangle_l^{(3^{6n}-1)/l}$, whose value in $\mathbb{F}^*_{3^{6n}}$ can be uniquely determined.

Granger et. al. proposed another technique to remove the ambiguity [9]. In this paper we denote by *GPS encoding* the technique proposed by Granger et. al. according to the authors' name (refer Sects. 3.2).

2.2 Efficient Pairings on Supersingular Curves over $\mathbb{F}_{3^n}$

We explain about some efficient algorithms for computing the bilinear pairing over supersingular curves with characteristic three. Algorithm 1 is the Duursma-Lee algorithm which outputs $\langle P, \psi(Q) \rangle_{3^{3n}+1}$ for $P, Q \in E^b(\mathbb{F}_{3^n})$ [11]. The Duursma-Lee algorithm has n interactions in the main loop and the whole computational cost is $15nM_1 + (10n + 2)C_1$. Note that the final exponentiation of the Duursma-Lee algorithm uses the powering to $(3^{6n} - 1)/(3^{3n} + 1) = (3^{3n} - 1)$.

Algorithm 1: Duursma-Lee Algorithm [11]

```
input: $P = (x_p, y_p),\ Q = (x_q, y_q) \in E^b(\mathbb{F}_{3^n})[l]$
output: $\langle P, \psi(Q) \rangle_{3^{3n}+1} \in \mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^{3^{3n}+1}$
$R_0 \leftarrow 1$ (in $\mathbb{F}_{3^{6n}}$)
$x_q \leftarrow x_q^3,\ y_q \leftarrow y_q^3$ (in $\mathbb{F}_{3^n}$)
$d \leftarrow (bn \bmod 3)$
for $i \leftarrow 0$ to $n - 1$ do
    $x_p \leftarrow x_p^9,\ y_p \leftarrow y_p^9$ (in $\mathbb{F}_{3^n}$)
    $r_0 \leftarrow x_p + x_q + d$ (in $\mathbb{F}_{3^n}$)
    $R_1 \leftarrow -r_0^2 - y_p y_q \sigma - r_0\rho - \rho^2$ (in $\mathbb{F}_{3^{6n}}$)
    $R_0 \leftarrow R_0^3$ (in $\mathbb{F}_{3^{6n}}$)
    $R_0 \leftarrow R_0 R_1$ (in $\mathbb{F}_{3^{6n}}$)
    $y_q \leftarrow -y_q$ (in $\mathbb{F}_{3^n}$)
    $d \leftarrow ((d - b) \bmod 3)$
end for
return $R_0$ (Cost: $15nM_1 + (10n + 2)C_1$)
```

Next Barreto et. al. introduced the η_T pairing [2]. The η_T pairing is also defined on supersingular elliptic curves formed by (1) for $n \equiv 1, 5 \pmod 6$ in the case of characteristic three. Beuchat et. al. proposed a variation of the η_T pairing (Algorithm 2), which requires no cube root calculation and outputs $\eta_T(P, Q)^{3^{(n+1)/2}}$ in the case of $n \equiv$

1 (mod 12) [3]. The number of iterations in the main loop of the η_T pairing becomes $(n+1)/2$, which is half for the Duursma-Lee algorithm. The computational cost of Algorithm 2 is $(7.5n + 8.5)M_1 + (5n+5)C_1$.

```
Algorithm 2: Computation of $\eta_T(P,Q)^{3^{(n+1)/2}}$
             for $n \equiv 1 \pmod{12}$ [3]
input: $P = (x_p, y_p), Q = (x_q, y_q) \in E^b(\mathbb{F}_{3^n})[l]$
output: $\eta_T(P,Q)^{3^{(n+1)/2}} \in \mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^{3^n+1+b'3^{(n+1)/2}}$
if $b = 1$ then $y_p \leftarrow -y_p$
$d \leftarrow b$  (in $\mathbb{F}_3$)
$R_0 \leftarrow -y_p(x_p + x_q + b) + y_q\sigma + y_p\rho$  (in $\mathbb{F}_{3^{6n}}$)
for $i \leftarrow 0$ to $(n-1)/2$ do
    $r_0 \leftarrow x_p + x_q + d$  (in $\mathbb{F}_{3^n}$)
    $R_1 \leftarrow -r_0^2 + y_p y_q \sigma - r_0\rho - \rho^2$  (in $\mathbb{F}_{3^{6n}}$)
    $R_0 \leftarrow R_0 R_1$  (in $\mathbb{F}_{3^{6n}}$)
    $y_p \leftarrow -y_p$  (in $\mathbb{F}_{3^n}$)
    $x_q \leftarrow x_q^9$, $y_q \leftarrow y_q^9$  (in $\mathbb{F}_{3^n}$)
    $R_0 \leftarrow R_0^3$  (in $\mathbb{F}_{3^{6n}}$)
    $d \leftarrow ((d - b) \bmod 3)$
end for
return $R_0$  (Cost: $(7.5n + 8.5)M_1 + (5n+5)C_1$)
```

Note that the η_T pairing itself does not satisfy the bilinearity. Therefore we have to compute the final exponentiation with W-th powering with

$$\begin{aligned} W &= (3^{3n}-1)(3^n+1)(3^n+1-b'3^{(n+1)/2}) \\ &= (3^{6n}-1)/\#E^b(\mathbb{F}_{3^n}). \end{aligned} \tag{4}$$

This powering function by W is the final exponentiation in the η_T pairing.

We note that $(\eta_T(P,Q)^{3^{(n+1)/2}})^W$ is a bilinear and non-degenerate pairing as well as the modified Tate pairing or the η_T pairing with final exponentiation, where W is given by Eq. (4). Then we can use $(\eta_T(P,Q)^{3^{(n+1)/2}})^W$ in almost cryptographic protocols which require a pairing without conversion to Tate pairing or $\eta_T(P,Q)^W$.

If necessary, we can calculate the modified Tate pairing from $(\eta_T(P,Q)^{3^{(n+1)/2}})^W$. First $\eta_T(P,Q)^W$ is obtained due to powering by $-3^{(n+1)/2}$. Next we use the following relationship between the modified Tate pairing and the η_T pairing,

$$(\eta_T(P,Q)^W)^{3T^2} = \hat{e}(P,Q)^Z,$$

where $T = -b'3^{(n+1)/2} - 1, Z = -b'3^{(n+3)/2}$ [2].

3. Efficient Final Exponentiation and GPS Encoding

In this section we present the final exponentiation of Tate pairing and the GPS encoding that requires no final exponentiation.

3.1 Efficient Final Exponentiation for Duursma-Lee Algorithm

We recall how to efficiently compute the final exponentiation of the Duursma-Lee algorithm, namely $A^{3^{3n}-1}$ for $A \in \mathbb{F}_{3^{6n}}$ [10].

The base of $\mathbb{F}_{3^{6n}}$ over $\mathbb{F}_{3^n}$ is fixed with $\{1, \sigma, \rho, \sigma\rho, \rho^2, \sigma\rho^2\}$ as we discussed in Sect. 2. Let A_0 and A_1 be elements in $\mathbb{F}_{3^{3n}}$ with $A_0 = a_0 + a_2\rho + a_4\rho^2$ and $A_1 = a_1 + a_3\rho + a_5\rho^2$. Then every element $A \in \mathbb{F}_{3^{6n}}$ is represented as

$$\begin{aligned} A &= A_0 + A_1\sigma \\ &= a_0 + a_1\sigma + a_2\rho + a_3\sigma\rho + a_4\rho^2 + a_5\sigma\rho^2. \end{aligned}$$

This means that $\mathbb{F}_{3^{6n}}$ is a quadratic extension from $\mathbb{F}_{3^{3n}}$ with the basis $\{1, \sigma\}$. It is easily to know that $\sigma^{3^{3n}} = -\sigma$ for $n \equiv 1, 5 \pmod 6$ which is a necessary condition for the Duursma-Lee algorithm and the η_T pairing algorithms. We then have the relationship

$$\begin{aligned} A^{3^{3n}} &= (A_0 + A_1\sigma)^{3^{3n}} = A_0^{3^{3n}} + A_1^{3^{3n}}\sigma^{3^{3n}} \\ &= A_0 - A_1\sigma \end{aligned}$$

for $A = A_0 + A_1\sigma \in \mathbb{F}^*_{3^{6n}}$. Therefore, the final exponentiation for the Duursma-Lee algorithm is performed as follows:

$$A^{3^{3n}-1} = \frac{A^{3^{3n}}}{A} = \frac{A_0 - A_1\sigma}{A_0 + A_1\sigma}.$$

Moreover $(A_0 + A_1\sigma)\cdot(A_0 - A_1\sigma) = A_0^2 + A_1^2 \in \mathbb{F}^*_{3^{3n}}$ yields the equation

$$A^{3^{3n}-1} = \frac{(A_0 - A_1\sigma)^2}{A_0^2 + A_1^2} = \frac{(A_0^2 - A_1^2) - 2A_0A_1\sigma}{A_0^2 + A_1^2}. \tag{5}$$

Then the computational cost of the final exponentiation for the Duursma-Lee algorithm is

$$5M_3 + I_3 = 30M_1 + I_3. \tag{6}$$

3.2 GPS Encoding in $\mathbb{F}^*_{3^{6n}}/\mathbb{F}^*_{3^{3n}}$

The GPS encoding is another technique of removing the ambiguity of representation from the cosets in a quotient group $\mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^l$ [9].

Denote by $\mathcal{G}$ be a quotient group resulting from the Duursma-Lee algorithm, namely $\mathcal{G} = \mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^{3^{3n}+1}$. This group $\mathcal{G}$ has a group law which is isomorphic to a subgroup of $\mathbb{F}^*_{3^{6n}}$. We then have the relationship $\mathcal{G} = \mathbb{F}^*_{3^{6n}}/\mathbb{F}^*_{3^{3n}}$ due to $\mathbb{F}^*_{3^{3n}} = (\mathbb{F}^*_{3^{6n}})^{3^{3n}+1}$. In other words, both $A_0 + A_1\sigma$ and $(\lambda A_0) + (\lambda A_1)\sigma$ are contained in the same coset for any $\lambda \in \mathbb{F}^*_{3^{3n}}$. Especially $A_0 + A_1\sigma$ is equivalent to $A_0/A_1 + \sigma$ in $\mathcal{G}$ in the case of $A_1 \neq 0$. Therefore the map

$$\begin{aligned} \tau \ : \quad \mathcal{G} &\to \mathbb{F}_{3^{3n}} \cup \{\mathcal{O}\}, \\ A_0 + A_1\sigma &\mapsto \begin{cases} A_0/A_1 & \text{if } A_1 \neq 0 \\ \mathcal{O} & \text{if } A_1 = 0 \end{cases} \end{aligned}$$

is a bijection and gives a representation for $\mathcal{G}$ without ambiguity, where $\mathcal{O}$ is the point at infinity. This representation for $\mathcal{G}$ is called the GPS encoding in this paper. The computational cost for computing the GPS encoding for a given $A \in \mathbb{F}^*_{3^{6n}}$ is

$$M_3 + I_3 = 6M_1 + I_3,$$

Table 1 Final exponentiation and GPS encoding for the Duursma-Lee algorithm.

	Output of Duursma-Lee algorithm	GPS encoding
Group	$\mathcal{G} = \mathbb{F}^*_{3^{6n}}/\mathbb{F}^*_{3^{3n}}$	$\mathbb{F}^*_{3^n} \cup \{\mathcal{O}\}$
Element	$A_0 + A_1\sigma,\ A_0, A_1 \in \mathbb{F}_{3^{3n}}$	A_0/A_1 (Cost: $6M_1 + I_3$)
Final exponentiation	$\dfrac{A_0 - A_1\sigma}{A_0 + A_1\sigma}$ (Cost: $30M_1 + I_3$)	–

GPS encoding requires no final exponentiation.

because the map τ is performed by one division in $\mathbb{F}_{3^{3n}}$ (= one inversion and one multiplication).

Table 1 gives a comparison of the final exponentiation with the GPS encoding for the Duursma-Lee algorithm.

4. The Proposed Algorithm

In this section we present a new efficient final exponentiation and the GPS encoding for the η_T pairing.

4.1 Torus $T_2(\mathbb{F}_{3^{3n}})$

Granger et. al. introduced the torus $T_2(\mathbb{F}_{3^{3n}})$ for compressing the value of $\mathbb{F}_{3^{6n}}$ [9]. At first we describe the arithmetic of the torus.

Let L be an m-th extension field of a field k. Let $N_{L/F}$ be a norm map to field F with $k \subset F \subsetneq L$. The torus $T_m(k)$ is a subgroup of L^* defined by $T_m(k) = \cap_{k \subset F \subsetneq L} Ker[N_{L/F}]$. In the paper we especially deal with the $T_2(k) = Ker[N_{L/k}]$ in the case of $m = 2$, $k = \mathbb{F}_{3^{3n}}$, and $L = \mathbb{F}_{3^{6n}}$. Every element in $\mathbb{F}^*_{3^{6n}}$ is represented as $A = A_0 + A_1\sigma$ with $A_0, A_1 \in \mathbb{F}_{3^{3n}}$. The conjugate element of $A = A_0 + A_1\sigma$ in $\mathbb{F}^*_{3^{6n}}$ is $\bar{A} = A_0 - A_1\sigma$, and thus $N_{\mathbb{F}_{3^{6n}}/\mathbb{F}_{3^{3n}}}(A) = A\bar{A} = A_0^2 + A_1^2$. Therefore $T_2(\mathbb{F}_{3^{3n}})$ can be represented by

$$T_2(\mathbb{F}_{3^{3n}}) = \{A_0 + A_1\sigma \in \mathbb{F}^*_{3^{6n}} \ :\ A_0^2 + A_1^2 = 1\}.$$

The element $A_0 + A_1\sigma \in \mathbb{F}_{3^{6n}}$ can be compressed to the half using the relationship $A_0^2 + A_1^2 = 1$ (Refer [9] for the further results about the compression of the pairing value).

4.2 The Proposed Final Exponentiation

We point out that some operations in the torus $T_2(\mathbb{F}_{3^{3n}})$ can be computed efficiently. We then present a new efficient final exponentiation algorithm for the η_T pairing.

At first we can easily prove the following lemma.

Lemma 1: The torus $T_2(\mathbb{F}_{3^{3n}})$ has following properties.
(i) $A_0 - A_1\sigma = (A_0 + A_1\sigma)^{-1}$ for $A_0 + A_1\sigma \in T_2(\mathbb{F}_{3^{3n}})$.
(ii) $(A_0 + A_1\sigma)^{3^{3n}-1} \in T_2(\mathbb{F}_{3^{3n}})$ for $A_0 + A_1\sigma \in \mathbb{F}^*_{3^{6n}}$.

proof (i) $A_0 - A_1\sigma$ is the inverse of $A_0 + A_1\sigma$ due to $(A_0+A_1\sigma)(A_0-A_1\sigma) = A_0^2+A_1^2 = 1$ for $A_0+A_1\sigma \in T_2(\mathbb{F}_{3^{3n}})$. (ii) The summation of a squaring of the constant term and that of the coefficient of Eq. (5) is equal to $\dfrac{(A_0^2 - A_1^2)^2 + (2A_0A_1)^2}{(A_0^2 + A_1^2)^2} = 1$, and thus we obtain $(A_0 + A_1\sigma)^{3^{3n}-1} \in T_2(\mathbb{F}_{3^{3n}})$. □

Therefore, the computational cost of the inversion in the torus $T_2(\mathbb{F}_{3^{3n}})$ is virtually for free.

Next let $A \in \mathbb{F}_{3^{6n}}$ be an output value from the η_T pairing. Note that $B = A^{3^{3n}-1}$ is contained in the torus $T_2(\mathbb{F}_{3^{3n}})$ due to Lemma 1. Then the final exponentiation A^W with $W = (3^{3n} - 1)(3^n + 1)(3^n + 1 - b'3^{(n+1)/2})$ can be computed as follows:

$$A^W = \begin{cases} D \cdot E^{-1} & \text{if } b' = 1 \\ D \cdot E & \text{if } b' = -1, \end{cases}$$

where $D = C^{3^n+1}$ and $E = C^{3^{(n+1)/2}}$ with $C = B^{3^n+1}$. It is easily to see that C, D and $E \in T_2(\mathbb{F}_{3^{3n}})$ since $T_2(\mathbb{F}_{3^{3n}})$ is a subgroup of $\mathbb{F}^*_{3^{6n}}$. The computation of $C^{3^{(n+1)/2}}$ can be efficiently performed by repeatedly calling the cubing algorithm in $\mathbb{F}_{3^n}$. On other hand we have the following lemma for the computation of X^{3^n+1} with $X \in T_2(\mathbb{F}_{3^{3n}})$ that requires no cubing.

Lemma 2: Let $n \equiv 1, 5 \pmod 6$. For $X = X_0 + X_1\sigma \in T_2(\mathbb{F}_{3^{3n}})$ we can compute $Y = \Lambda(X) = X^{3^n+1} = Y_0 + Y_1\sigma$ with 9 multiplications in $\mathbb{F}_{3^n}$ as follows:
Let $z_0 \sim z_8$ be defined as

$$\begin{aligned} &z_0 = x_0x_4, \quad z_1 = x_1x_5, \quad z_2 = x_2x_4, \\ &z_3 = x_3x_5, \quad z_4 = (x_0 + x_1)(x_4 - x_5), \\ &z_5 = x_1x_2, \quad z_6 = x_0x_3, \\ &z_7 = (x_0 + x_1)(x_2 + x_3), \\ &z_8 = (x_2 + x_3)(x_4 - x_5), \end{aligned}$$

then, Y can be computed as following table, where $X_0 = x_0 + x_2\rho + x_4\rho^2, X_1 = x_1 + x_3\rho + x_5\rho^2$ and $Y_0 = y_0 + y_2\rho + y_4\rho^2, Y_1 = y_1 + y_3\rho + y_5\rho^2$ ($x_i, y_i \in \mathbb{F}_{3^n}$) for $i = 0, 1, \ldots, 5$.

		Case of $n \equiv 1 \pmod 6$
y_0	$=$	$1 + z_0 + z_1 - bz_2 - bz_3$
y_1	$=$	$z_1 + z_4 + bz_5 - z_0 - bz_6$
y_2	$=$	$z_7 - z_2 - z_3 - z_5 - z_6$
y_3	$=$	$bz_0 + z_3 + z_8 - z_2 - bz_1 - bz_4$
y_4	$=$	$bz_2 + bz_3 + bz_7 - bz_5 - bz_6$
y_5	$=$	$bz_3 + bz_8 - bz_2$

		Case of $n \equiv 5 \pmod 6$
y_0	$=$	$1 + z_0 + z_1 + bz_2 + bz_3$
y_1	$=$	$z_1 + z_4 - bz_5 - z_0 + bz_6$
y_2	$=$	$z_5 + z_6 - z_7$
y_3	$=$	$-bz_0 + z_3 + z_8 - z_2 + bz_1 + bz_4$
y_4	$=$	$bz_2 + bz_3 + bz_7 - bz_5 - bz_6$
y_5	$=$	$-bz_3 - bz_8 + bz_2$

proof Refer Appendix A. □

From Lemma 2 the proposed algorithm can be obtained. We describe the explicit algorithm of the proposed

```
Algorithm 3: Proposed Final Exponentiation of $\eta_T$ Pairing
input: $A = (a_0, a_1, a_2, a_3, a_4, a_5) \in \mathbb{F}^*_{3^{6n}}$,   $b' \in \{-1, 1\}$
output: $A^W \in \mathbb{F}^*_{3^{6n}}$ for $W = (3^{3n} - 1)(3^n + 1)(3^n + 1 - b'3^{(n+1)/2})$
$B \leftarrow A^{3^{3n}-1}$  (in $\mathbb{F}_{3^{6n}}$)                  (Eq.(5))
$C \leftarrow B^{3^n+1} = \Lambda(B)$  (in $T_2(\mathbb{F}_{3^{3n}})$)          (Lemma 2)
$D \leftarrow C^{3^n+1} = \Lambda(C)$  (in $T_2(\mathbb{F}_{3^{3n}})$)          (Lemma 2)
$E \leftarrow C$
for $i \leftarrow 0$ to $(n-1)/2$ do
    $E \leftarrow E^3$  (in $\mathbb{F}_{3^{6n}}$)
end for
if $(b' = 1)$ then return $D \cdot \overline{E}$  (in $\mathbb{F}_{3^{6n}}$)   (Cost: $66M_1 + (3n+3)C_1 + I_3$)
else return $D \cdot E$  (in $\mathbb{F}_{3^{6n}}$)               (Cost: $66M_1 + (3n+3)C_1 + I_3$)
```

scheme in Algorithm 3. Note that although a computation of X^{3^n+1} takes $9M_1$ only for $X \in T_2(\mathbb{F}_{3^n})$ due to this lemma, X^{3^n+1} takes $18M_1$ for arbitrary $X \in \mathbb{F}^*_{3^{6n}}$ due to Eq. (3).

Proposition 1: Algorithm 3 requires $66M_1+(3n+3)C_1+I_3$, where M_1, C_1, I_3 are the cost of multiplication in $\mathbb{F}_{3^n}$, cubing in $\mathbb{F}_{3^n}$, and inversion in $\mathbb{F}_{3^{3n}}$, respectively.

proof The computation of $B = A^{3^{3n}-1}$ is as expensive as that of the final exponentiation for the Duursma-Lee algorithm, namely $30M_1 + I_3$ from Eq. (6). The calculations of C and D are performed by a powering to the $(3^n + 1)$-th power. The calculation of E is performed by $(n + 1)/2$ cubings (its cost is $(n + 1)/2 \cdot C_6 = (3n + 3)C_1$). We have to calculate E^{-1} in the case of $b' = 1$, which requires no cost due to Lemma 1. Hence the proposed algorithm of computing the final exponentiation for the η_T pairing needs $(30M_1 + I_3) + (3n + 3)C_1 + 2C_T + M_6$, where $C_T = 9M_1$ is the cost of powering to $(3^n + 1)$-th in $T_2(\mathbb{F}_{3^{3n}})$. We thus obtain the cost estimation of this proposition. □

4.3 How to Apply GPS Encoding to η_T Pairing

In this section we explain how to apply the GPS encoding to the η_T pairing.

The GPS encoding utilizes the arithmetic of the image of the Duursma-Lee algorithm, $\mathcal{G} = \mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^{3^{3n}+1}$. However, the image of the η_T pairing is included in $\mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^{3^n+1+b'3^{(n+1)/2}}$. Hence we cannot directly apply the GPS encoding to the η_T pairing, we need a modification. Let $\alpha \cdot \beta^{3^n+1+b'3^{(n+1)/2}} = \eta_T(P, Q)$ for $\alpha, \beta \in \mathbb{F}_{3^{6n}}$. Then $\eta_T(P, Q)^V$ with $V = (3^n + 1)(3^n + 1 - b'(3^{(n+1)/2}))$ is regarded as an element in $\mathcal{G}$ since $\eta_T(P, Q)^V = \alpha^V \cdot (\beta^{3^n+1+b'3^{(n+1)/2}})^V = \alpha^V \cdot \beta^{3^{3n}+1}$. Therefore we propose to apply the GPS encoding to $\eta_T(P, Q)^V$. In order to compute the powering by V we have to compute in $\mathbb{F}_{3^{6n}}$ since $\eta_T(P, Q)$ is contained neither in $\mathcal{G}$ nor $T_2(\mathbb{F}_{3^{3n}})$. We have a relationship $\eta_T(P, Q)^V = CD^{-b'}$, where $C = B^{3^n+1}, D = B^{3^{(n+1)/2}}$, and $B = \eta_T(P, Q)^{3^n+1}$. Algorithm 4 shows the GPS encoding for the η_T pairing.

Steps 1 ∼ 4 in Algorithm 4 compute a powering by V, and Step 5 is the same process as the original GPS encoding.

```
Algorithm 4: Modified GPS Encoding for $\eta_T$ Pairing
input: $A \in \mathbb{F}^*_{3^{6n}}/(\mathbb{F}^*_{3^{6n}})^{3^n+1-b'3^{(n+1)/2}}$
output: GPS encoding of $A \in \mathbb{F}^*_{3^{6n}}$
1.  $B \leftarrow A^{3^n+1}$  (in $\mathbb{F}_{3^{6n}}$)
2.  $C \leftarrow B^{3^n+1}$  (in $\mathbb{F}_{3^{6n}}$)
3.  $D \leftarrow B^{3^{(n+1)/2}}$  (in $\mathbb{F}_{3^{6n}}$)
4.  if $b' = 1$ then $E \leftarrow C \cdot D^{-1}$  (in $\mathbb{F}_{3^{6n}}$)
    else $E \leftarrow C \cdot D$  (in $\mathbb{F}_{3^{6n}}$)
5.  return $E_0/E_1$, where $E = E_0 + E_1\sigma$  (in $\mathbb{F}_{3^{3n}}$)
    ( cost:  $90M_1 + (3n+3)C_1 + 2I_3$    if $b' = 1$
             $60M_1 + (3n+3)C_1 + I_3$     if $b' = -1$ )
```

We estimate the computational cost of Algorithm 4. Recall that $X^{3^n} \in \mathbb{F}_{3^{6n}}$ is computed virtually for free (see [3] or Appendix B). Therefore the cost of computing $X^{3^n+1} = X^{3^n} \cdot X$ is just $M_6 = 18M_1$. The total costs of both Step 1 and 2 are $36M_1$. The cost of $B^{3^{(n+1)/2}}$ is $((n+1)/2) \cdot C_6 = (3n+3)C_1$. The cost of $C \cdot D^{-1}$ is $M_6 + I_6 = 48M_1 + I_3$ and the cost of $C \cdot D$ is $M_6 = 18M_1$. The computation of E_0/E_1 which is same as the original GPS encoding takes $6M_1 + I_3$. Consequently the GPS encoding for the η_T pairing is $90M_1 + (3n + 3)C_1 + 2I_1$ if $b' = 1$ and $60M_1 + (3n + 3)C_1 + I_1$ if $b' = -1$.

4.4 Comparison

Here we compare the computational cost of the proposed scheme with other schemes.

The computational cost of an exponentiation with cubings and multiplications by bit is $2nM_6/3 + (n - 1)C_6 = 12nM_1 + 6(n - 1)C_1$ on average. The previously fastest method using Frobenius and no Torus proposed by [2] for computing the final exponentiation requires $10M_6 + (n + 3)C_6/2 + I_6 = 210M_1 + (3n + 9)C_1 + I_3$.

The center part of Table 2 concludes the computational cost of a final exponentiation and the GPS encooding for the η_T and the Duursma-Lee algorithm using M_1, C_1 and I_3. In order to easily understand the result in Table 2 we present the cost estimation only using M_1 in the right part of Table 2. We use the cost relationship among C_1, M_1, I_3 for several n's in Table 3 [9].

Note that there is another relationship among them [8]. A trinomial basis is used in [9], and a normal basis is used in [8] for a basis of $\mathbb{F}_{3^n}$ over $\mathbb{F}_3$. If the normal basis is used, then C_1 becomes virtually for free, however, M_1 becomes considerably higher.

First we discuss for $n = 97$ corresponding standard security. The cost of the final exponentiation appeared in [2] is $267.6M_1$, which is about 25% of the total cost $1071.9M_1$

Table 2 Comparison of the cost of several bilinear pairing algorithms.

η_T pairing ($\eta_T(P,Q)^{3^{(n+1)/2}}$)		computational cost	cost estimation using M_1				
			$n = 97$	163	193	239	353
Proposed final exponentiation		$\mathbf{66M_1} + \mathbf{(3n+3)C_1} + \mathbf{I_3}$	122.7	118.9	122.6	130.1	126.9
(Algorithm 3)	total cost	$(7.5n + 74.5)M_1 + (8n+8)C_1 + I_3$	927.1	1411.4	1647.2	2007.7	2855.6
Modified GPS encoding ($b' = 1$)		$\mathbf{90M_1} + \mathbf{(3n+3)C_1} + \mathbf{2I_3}$	162.5	158.8	162.1	172.1	168.1
(Algorithm 4)	total cost	$(7.5n + 98.5)M_1 + (8n+8)C_1 + 2I_3$	966.8	1451.3	1686.7	2049.8	2896.8
Modified GPS encoding ($b' = -1$)		$\mathbf{60M_1} + \mathbf{(3n+3)C_1} + \mathbf{I_3}$	116.7	112.9	116.6	124.1	120.9
(Algorithm 4)	total cost	$(7.5n + 68.5)M_1 + (8n+8)C_1 + I_3$	921.1	1405.4	1641.2	2001.7	2849.6
Ordinary final exponentiation		$210M_1 + (3n+9)C_1 + I_3$	267.6	263.3	267.0	274.4	271.1
([2])	total cost	$(7.5n + 236.5)M_1 + (8n+8)C_1 + I_3$	1071.9	1555.8	1791.6	2152.1	2999.9

Duursma-Lee algorithm		computational cost	cost estimation using M_1				
			$n = 97$	163	193	239	353
Final exponentiation		$30M_1 + I_3$	45.7	46.0	45.5	48.0	47.2
([10])	total cost	$(15n + 30)M_1 + (10n+2)C_1 + I_3$	1636.3	2613.4	3077.1	3785.9	5487.4
GPS encoding		$6M_1 + I_3$	21.7	22.0	21.5	24.0	23.2
([9])	total cost	$(15n + 6)M_1 + (10n+2)C_1 + I_3$	1612.3	2589.4	3053.1	3761.9	5463.4

Table 3 Relationship among C_1, M_1 and I_3.

n	97	163	193	239	353
C_1	$0.1395M_1$	$0.0750M_1$	$0.0707M_1$	$0.0639M_1$	$0.0411M_1$
I_3	$15.73M_1$	$15.97M_1$	$15.47M_1$	$18.05M_1$	$17.21M_1$

of the η_T pairing. On the other hand, the computational cost of our proposed final exponentiation is $122.7M_1$ and the total takes $927.1M_1$, namely it is about 13% of the whole η_T pairing. Therefore, the proposed scheme can compute the η_T pairing about 16% faster than the previously known algorithms. If $b' = -1$, the cost of the GPS encoding and the total take $116.7M_1$ and $921.1M_1$, respectively. Hence, the GPS encoding for η_T is able to compute the η_T pairing moreover faster.

We notice that the larger extension degree n is, the smaller the ratio of the cost of final exponentiation is. Then the larger n is, the smaller the improvement rate is. For example, the improvement rate of whole cost of the η_T pairing is about 5% only for $n = 353$. However, it is non-negligible. Final exponentiation used torus or the GPS encoding for η_T gives more than 2 times of efficient final exponentiation for any n.

5. Conclusion

In this paper we presented some new efficient algorithms for the final exponentiation of the η_T pairing using powering by $3^n + 1$ and inversion algorithms in torus. Moreover we modified the GPS encoding to apply to the η_T pairing. The total cost of computing the η_T pairing with $n = 97$ has become about 16% faster than the previously known methods.

Acknowledgments

This work was supported by the New Energy and Industrial Technology Development Organization (NEDO), Japan. The authors would like to thank Steven Galbraith for his comments on the previous version.

References

[1] P. Barreto, H. Kim, B. Lynn, and M. Scott, “Efficient algorithms for pairing-based cryptosystems,” CRYPTO 2002, LNCS 2442, pp.354–368, 2002.

[2] P. Barreto, S. Galbraith, C. ÓhÉigeartaigh, and M. Scott, “Efficient pairing computation on supersingular abelian varieties,” Des. Codes Cryptogr., vol.42, no.3, pp.239–271, Springer-Verlag, 2007.

[3] J.-L. Beuchat, M. Shirase, T. Takagi, and E. Okamoto, “An algorithm for the η_T pairing calculation in characteristic three and its hardware implementation,” 18th IEEE International Symposium on Computer Arithmetic, ARITH-18, pp.97–104, 2007.

[4] D. Boneh and M. Franklin, “Identity based encryption from the Weil pairing,” SIAM J. Comput., vol.32, no.3, pp.586–615, 2003.

[5] D. Boneh, C. Gentry, and B. Waters, “Collusion resistant broadcast encryption with short ciphertexts and private keys,” CRYPTO 2005, LNCS 3621, pp.258–275, 2005.

[6] D. Boneh, B. Lynn, and H. Shacham, “Short signature from the Weil pairing,” J. Cryptol., vol.17, no.4, pp.297–319, 2004.

[7] I. Duursma and H. Lee, “Tate pairing implementation for hyperelliptic curves $y^2 = x^p - x + d$,” ASIACRYPT 2003, LNCS 2894, pp.111–123, 2003.

[8] R. Granger, D. Page, and M. Stam, “Hardware and software normal basis arithmetic for pairing-based cryptography in characteristic three,” IEEE Trans. Comput., vol.54, no.7, pp.852–860, July 2005.

[9] R. Granger, D. Page, and M. Stam, “On small characteristic algebraic tori in pairing-based cryptography,” LMS J. Comput. Math., vol.9, pp.64–85, 2006.

[10] T. Kerins, W. Marnane, E. Popovici, and P. Barreto, “Efficient hardware for the Tate pairing calculation in characteristic three,” CHES 2005, LNCS 3659, pp.412–426, 2005.

[11] S. Kwon, “Efficient Tate pairing computation for supersingular elliptic curves over binary fields,” Cryptology ePrint Archive, Report 2004/303, 2004.

[12] R. Ronan, C. ÓhÉigeartaigh, C. Murphy, T. Kerins, and P. Barreto, “Hardware implementation of the η_T pairing in characteristic 3,” Cryptology ePrint Archive, Report 2006/371, 2006.

[13] C. Shu, S. Kwon, and K. Gaj, “FPGA accelerated Tate pairing based cryptosystems over binary fields,” Cryptology ePrint Archive, Report 2006/179, 2006.

Appendix A: Proofs of Lemma 2

Lemma 2. *Let* $n \equiv 1, 5 \pmod 6$. *For* $X = X_0 + X_1\sigma \in T_2(\mathbb{F}_{3^{3n}})$ *we can compute* $Y = \Lambda(X) = X^{3^n+1} = Y_0 + Y_1\sigma$ *with 9 multiplications in* $\mathbb{F}_{3^n}$ *as follows:*
Let $z_0 \sim z_8$ *be defined as*

$$\begin{aligned}
&z_0 = x_0x_4, \quad z_1 = x_1x_5, \quad z_2 = x_2x_4,\\
&z_3 = x_3x_5, \quad z_4 = (x_0 + x_1)(x_4 - x_5),\\
&z_5 = x_1x_2, \quad z_6 = x_0x_3,\\
&z_7 = (x_0 + x_1)(x_2 + x_3),\\
&z_8 = (x_2 + x_3)(x_4 - x_5),
\end{aligned}$$

then, Y can be computed as following table, where $X_0 = x_0 + x_2\rho + x_4\rho^2, X_1 = x_1 + x_3\rho + x_5\rho^2$ *and* $Y_0 = y_0 + y_2\rho + y_4\rho^2, Y_1 = y_1 + y_3\rho + y_5\rho^2$ $(x_i, y_i \in \mathbb{F}_{3^n})$ *for* $i = 0, 1, \ldots, 5$.

	Case of $n \equiv 1 \pmod 6$	
y_0	$=$	$1 + z_0 + z_1 - bz_2 - bz_3$
y_1	$=$	$z_1 + z_4 + bz_5 - z_0 - bz_6$
y_2	$=$	$z_7 - z_2 - z_3 - z_5 - z_6$
y_3	$=$	$bz_0 + z_3 + z_8 - z_2 - bz_1 - bz_4$
y_4	$=$	$bz_2 + bz_3 + bz_7 - bz_5 - bz_6$
y_5	$=$	$bz_3 + bz_8 - bz_2$

	Case of $n \equiv 5 \pmod 6$	
y_0	$=$	$1 + z_0 + z_1 + bz_2 + bz_3$
y_1	$=$	$z_1 + z_4 - bz_5 - z_0 + bz_6$
y_2	$=$	$z_5 + z_6 - z_7$
y_3	$=$	$-bz_0 + z_3 + z_8 - z_2 + bz_1 + bz_4$
y_4	$=$	$bz_2 + bz_3 + bz_7 - bz_5 - bz_6$
y_5	$=$	$-bz_3 - bz_8 + bz_2$

proof We prove Lemma 2 for $n \equiv 1 \pmod 6$. The proof for $n \equiv 5 \pmod 6$ is omitted since it is almost same for $n \equiv 1 \pmod 6$.

Note that $\rho^{3^n} = \rho + b$, $(\rho^2)^{3^n} = \rho^2 - b\rho + 1$ in the case of $n \equiv 1 \pmod 6$. For $X = X_0 + X_1\sigma \in T_2(\mathbb{F}_{3^{3n}})$ we have the relationship:

$$\begin{aligned}
X^{3^n+1} &= X_0^{3^n+1} + X_1^{3^n+1} + (X_0^{3^n}X_1 - X_1^{3^n}X_0)\sigma,\\
X_0^{3^n} &= (x_0 + bx_2 + x_4) + (x_2 - bx_4)\rho + x_4\rho^2, && \text{(A· 1)}\\
X_0^{3^n+1} &= (x_0^2 + bx_0x_2 + x_0x_4 - bx_2x_4 - x_4^2)\\
&\quad +(-x_0x_2 - bx_0x_4 + bx_2^2)\rho && \text{(A· 2)}\\
&\quad +(-x_0x_4 + x_2^2 - x_4^2)\rho^2.
\end{aligned}$$

We similarly see

$$\begin{aligned}
X_1^{3^n} &= (x_1 + bx_3 + x_5) + (x_3 - bx_5)\rho + x_5\rho^2, && \text{(A· 3)}\\
X_1^{3^n+1} &= (x_1^2 + bx_1x_3 + x_1x_5 - bx_3x_5 - x_5^2)\\
&\quad +(-x_1x_3 - bx_1x_5 + bx_3^2)\rho && \text{(A· 4)}\\
&\quad +(-x_1x_5 + x_3^2 - x_5^2)\rho^2.
\end{aligned}$$

$X_0^2 + X_1^2 = 1$ is satisfied since $X = X_0 + X_1\sigma \in T_2(\mathbb{F}_{3^{3n}})$. This derives the following equation.

$$\begin{aligned}
&(x_0 + x_2\rho + x_4\rho^2)^2 + (x_1 + x_3\rho + x_5\rho^2)^2\\
&= 1\ (= 1 + 0\rho + 0\rho^2).
\end{aligned}$$

This equation gives

$$\left\{\begin{aligned}
&x_0^2 + x_1^2\\
&\quad = 1 + bx_2x_4 + bx_3x_5\\
&x_2^2 + x_3^2\\
&\quad = x_0x_4 + x_1x_5 - bx_0x_2\\
&\qquad -bx_1x_3 - bx_2x_4 - bx_3x_5\\
&x_4^2 + x_5^2\\
&\quad = bx_0x_2 + bx_1x_3 + bx_2x_4 + bx_3x_5.
\end{aligned}\right. \qquad \text{(A· 5)}$$

By Eqs. (A· 2), (A· 4), (A· 5)

$$\begin{aligned}
&X_0^{3^n+1} + X_1^{3^n+1}\\
&= y_0 + y_2\rho + y_4\rho^2\\
&= (1 + x_0x_4 + x_1x_5 - bx_2x_4 - bx_3x_5)\\
&\quad +(x_0x_2 + x_1x_3 - x_2x_4 - x_3x_5)\rho\\
&\quad +(bx_0x_2 + bx_1x_3 + bx_2x_4 + bx_3x_5)\rho^2.
\end{aligned}$$

And by Eqs. (A· 1), (A· 3), (A· 5)

$$\begin{aligned}
&X_0^{3^n}X_1 - X_1^{3^n}X_0\\
&= y_1 + y_3\rho + y_5\rho^2\\
&= (bx_1x_2 + x_1x_4 - bx_0x_3 - x_0x_5)\\
&\quad +(bx_0x_5 + x_3x_4 - bx_1x_4 - x_2x_5)\rho\\
&\quad +(bx_3x_4 - bx_2x_5)\rho^2.
\end{aligned}$$

Moreover, we define $z_0, \ldots, z_8$ by

$$\begin{aligned}
&z_0 = x_0x_4,\ z_1 = x_1x_5,\ z_2 = x_2x_4,\ z_3 = x_3x_5,\\
&z_4 = (x_0 + x_1)(x_4 - x_5),\ z_5 = x_1x_2,\ z_6 = x_0x_3,\\
&z_7 = (x_0 + x_1)(x_2 + x_3),\ z_8 = (x_2 + x_3)(x_4 - x_5).
\end{aligned}$$

Then we have

$$\begin{aligned}
y_0 &= 1 + x_0x_4 + x_1x_5 - bx_2x_4 - bx_3x_5\\
&= 1 + z_0 + z_1 - bz_2 - bz_3,\\
y_1 &= bx_1x_2 + x_1x_4 - bx_0x_3 - x_0x_5\\
&= z_1 + z_4 + bz_5 - z_0 - bz_6,\\
y_2 &= x_0x_2 + x_1x_3 - x_2x_4 - x_3x_5\\
&= z_7 - z_2 - z_3 - z_5 - z_6,\\
y_3 &= bx_0x_5 + x_3x_4 - bx_1x_4 - x_2x_5\\
&= bz_0 + z_3 + z_8 - z_2 - bz_1 - bz_4,\\
y_4 &= bx_0x_2 + bx_1x_3 + bx_2x_4 + bx_3x_5\\
&= bz_2 + bz_3 + bz_7 - bz_5 - bz_6,\\
y_5 &= bz_3z_4 - bz_2z_5\\
&= bz_3 + bz_8 - bz_2.
\end{aligned}$$

Consequently Lemma 2 is showed. □

Appendix B: Powering by 3^n and 3^n-th Root in $\mathbb{F}_{3^n}$

This section explains that a powering by 3^n or 3^n-th root in $\mathbb{F}_{3^{6n}}$ is computed virtually for free. Recall that $n \equiv 1, 5$ (mod 6) is a necessary condition of the Duursma-Lee algorithm and the η_T pairing. We deal with only for $n \equiv 1 \pmod 6$, however, the discussion for $n \equiv 5 \pmod 6$ becomes almost same. It follows that

$$\sigma^{3^n} = \begin{cases} \sigma & \text{if } n \equiv 0 \pmod 2\\ -\sigma & \text{if } n \equiv 1 \pmod 2 \end{cases},$$

$$\rho^{3^n} = \begin{cases} \rho & \text{if } n \equiv 0 \pmod 3\\ \rho + b & \text{if } n \equiv 1 \pmod 3\\ \rho - b & \text{if } n \equiv 2 \pmod 3 \end{cases}.$$

If $n \equiv 1 \pmod 6$ then we have

$$\sigma^{3^n} = -\sigma,\ \rho^{3^n} = \rho + b,\ (\rho^2)^{3^n} = \rho^2 - b\rho + 1.$$

B.1 Powering by 3^n

Let $Y = X^{3^n}$ for $X \in \mathbb{F}^*_{3^{6n}}$, where $X = x_0 + x_1\sigma + x_2\rho + x_3\sigma\rho + x_4\rho^2 + x_5\sigma\rho^2$ and $Y = y_0 + y_1\sigma + y_2\rho + y_3\sigma\rho + y_4\rho^2 + y_5\sigma\rho^2$ for some $x_i,\ y_i \in \mathbb{F}_{3^n}$. Then we have

$$\left\{\begin{array}{lcl} y_0 & = & x_0 + bx_2 + x_4 \\ y_1 & = & -x_1 - bx_3 - x_5 \\ y_2 & = & x_2 - bx_4 \\ y_3 & = & -x_3 + bx_5 \\ y_4 & = & x_4 \\ y_5 & = & -x_5 \end{array}\right. \qquad (\mathrm{A}\cdot 6)$$

since

$$\begin{array}{l} (x_0 + x_1\sigma + x_2\rho + x_3\sigma\rho + x_4\rho^2 + x_5\sigma\rho^2)^{3^n} \\ \quad = x_0 + x_1(\sigma)^{3^n} + x_2(\rho)^{3^n} + x_3(\sigma\rho)^{3^n} + x_4(\rho^2)^{3^n} \\ \qquad + x_5(\sigma\rho^2)^{3^n} \\ \quad = x_0 + x_1(-\sigma)^{3^n} + x_2(\rho + b) + x_3(-\sigma\rho - b\sigma) \\ \qquad + x_4(\rho^2 - b\rho + 1) + x_5(\sigma\rho^2 + b\sigma\rho - \sigma) \\ \quad = (x_0 + bx_2 + x_4) + (-x_1 - bx_3 - x_5)\sigma \\ \qquad + (x_2 - bx_4)\rho + (-x_3 + bx_5)\sigma\rho \\ \qquad + x_4\rho^2 - x_5\sigma\rho^2. \end{array}$$

Note that $x_i^{3^n} = x_i$ and $y_i^{3^n} = y_i$ since $x_i,\ y_i \in \mathbb{F}_{3^n}$. Therefore a powering by 3^n is computed virtually for free.

B.2 3^n-th Root

Let $Y = \sqrt[3^n]{X}$ for $X \in \mathbb{F}^*_{3^{6n}}$, where $X = x_0 + x_1\sigma + x_2\rho + x_3\sigma\rho + x_4\rho^2 + x_5\sigma\rho^2$ and $Y = y_0 + y_1\sigma + y_2\rho + y_3\sigma\rho + y_4\rho^2 + y_5\sigma\rho^2$ for some $x_i,\ y_i \in \mathbb{F}_{3^n}$. Note that a 3^n-th root operation in characteristic three is uniquely determined. We have

$$\left\{\begin{array}{lcl} x_0 & = & y_0 + by_2 + y_4 \\ x_1 & = & -y_1 - by_3 - y_5 \\ x_2 & = & y_2 - by_4 \\ x_3 & = & -y_3 + by_5 \\ x_4 & = & y_4 \\ x_5 & = & -y_5 \end{array}\right. \qquad (\mathrm{A}\cdot 7)$$

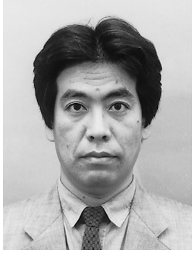
by Eq. (A·6) since $X = Y^{3^n}$. Solving Eq. (A·7) for each y_i gives

$$\left\{\begin{array}{lcl} y_0 & = & x_0 - bx_2 + x_4 \\ y_1 & = & -x_1 + bx_3 - x_5 \\ y_2 & = & x_2 + bx_4 \\ y_3 & = & -x_3 - bx_5 \\ y_4 & = & x_4 \\ y_5 & = & -x_5. \end{array}\right.$$

Therefore a 3^n-th root operation is computed virtually for free.

Masaaki Shirase received the B.Sc. in mathematics from Ibaraki University in 1994, and M.I.S. and Dr.I.S. degrees from JAIST (Japan Advanced Institute of Science and Technology) in 2003 and 2006, respectively. He is currently a Post Doctor in the School of Systems Science Information at Future University-Hakodate. His research interests are algorithm and implementation of cryptography.

Tsuyoshi Takagi received the B.Sc. and M.Sc. degrees in mathematics from Nagoya University in 1993 and 1995, respectively. He had engaged in the research on network security at NTT Laboratories from 1995 to 2001. He received the Dr.rer.nat degree from Technische Universität Darmsradt in 2001. He was an Assistant Professor in the Department of Computer Science at Technische Universität Darmstadt until 2005. He is currently an Associate Professor in the School of Systems Science Information at Future University-Hakodate. His current research interests are information security and cryptography. Dr. Takagi is a member of International Association for Cryptologic Research (IACR).



Eiji Okamoto received his B.S., M.S. and Ph.D. degree in electronics engineering from the Tokyo Institute of Technology in 1973, 1975 and 1978, respectively. He worked and studied communication theory and cryptography for NEC central research laboratories since 1978. Then he became a professor at JAIST from 1991, and at Toho University from 1999 until 2002. He is currently a professor at Graduate School of Systems and Information Engineering, University of Tsukuba. His research interests include cryptography and information security.