PAPER Special Section on Information Theory and Its Applications

A More Compact Representation of XTR Cryptosystem*

Masaaki SHIRASE[†a)], ***Member***, **Dong-Guk HAN**[††], ***Nonmember***, **Yasushi HIBINO**[†††], ***Member***, **Howon KIM**[††], ***Nonmember***, ***and*** **Tsuyoshi TAKAGI**[†], ***Member***

SUMMARY XTR is one of the most efficient public-key cryptosystems that allow us to compress the communication bandwidth of their ciphertext. The compact representation can be achieved by deploying a subgroup $\mathbb{F}_{q^2}$ of extension field $\mathbb{F}_{q^6}$, so that the compression ratio of XTR cryptosystem is 1/3. On the other hand, Dijk *et al*. proposed an efficient public-key cryptosystem using a torus over $\mathbb{F}_{q^{30}}$ whose compression ratio is 4/15. It is an open problem to construct an efficient public-key cryptosystem whose compression ratio is smaller than 4/15. In this paper we propose a new variant of XTR cryptosystem over finite fields with characteristic three whose compression ratio is 1/6. The key observation is that there exists a trace map from $\mathbb{F}_{q^6}$ to $\mathbb{F}_q$ in the case of characteristic three. Moreover, the cost of compression and decompression algorithm requires only about 1% overhead compared with the original XTR cryptosystem. Therefore, the proposed variant of XTR cryptosystem is one of the fastest public-key cryptosystems with the smallest compression ratio.

key words: *cryptography, XTR cryptosystem, finite field, efficient implementation, compact representation*

1. Introduction

In the classical Diffie-Hellman (DH) key exchange scheme, two system parameters are fixed: a large prime number q and a generator g of the multiplicative group of the basic prime field $\mathbb{F}_q$. In the basic DH scheme the two parties each send a random power of g to the other party. Assuming both parties know q and g, each party transmits about $\log_2(q)$ bits to the other party.

In [6], ElGamal suggested that finite extension fields can be used instead of prime fields, but no direct computational or communication advantages where implied. In [14], Schnorr proposed a variant of the classical Diffie-Hellman scheme, in which g does not generate the whole multiplicative group of the prime field $\mathbb{F}_q$, but only a small subgroup of which the order contain relatively small compared to q. This considerably reduces the computational cost of the DH scheme, but has no effect on the number of bits to be exchanged.

After that, it has tried to make use of traces to represent and calculate powers of elements of a subgroup of a finite field to achieve efficient and compact subgroup representation. The LUC cryptosystem uses the trace over $\mathbb{F}_q$ to represent elements of the order $q+1$ subgroup of $\mathbb{F}^*_{q^2}$ [16]. Compared to the traditional representation LUC leads to a factor 2 reduction in the representation size. The variant described in [7] uses the subgroup of order q^2+q+1 of $\mathbb{F}^*_{q^3}$ instead, but as a result sizes are reduced by only a factor 1.5. In [2], Brouwer *et al*. introduced for the first time how the use of finite extension fields and subgroups can be combined in such a way that the number of bits to be exchanged is reduced by a factor 3. More specifically, it was shown that elements of an order p subgroup of $\mathbb{F}^*_{q^6}$ can be represented using $2\log_2(q)$ bits if p divides q^2-q+1. Despite its communication efficiency, the method of it is rather troublesome and computationally not particularly efficient.

In 2000 Lenstra-Verheul introduced XTR [9], a cryptosystem using the trace over $\mathbb{F}_{q^2}$ to represent elements of the order q^2-q+1 subgroup of $\mathbb{F}^*_{q^6}$, there by achieving a factor 3 size reduction. Also, the resulting calculations are appreciably faster than using the standard representation. XTR of security equivalent to 1024-bit RSA achieves speed comparable to cryptosystems based on random elliptic curves over random prime fields (ECC) of equivalent security. The corresponding XTR public keys are only about 2∼3 times as large as ECC keys in practical key sizes, assuming global system parameters — without the last requirement the sizes of XTR and ECC public keys are the same. Furthermore, parameter initialization from scratch for XTR takes a negligible amount of computing time, unlike RSA and ECC. Combined with its very easy programmability, this makes XTR an excellent public-key cryptosystem for a very wide variety of environments, ranging from smart cards to web servers.

On the other hand, Rubin-Silverberg proposed a torus-based cryptosystem CEILIDH over $\mathbb{F}_{q^6}$ whose compression rate is same as XTR [13]. Dijk-Woodruff then presented that a torus-based cryptosystem over $\mathbb{F}_{q^n}$ whose compression ratio is asymptotically $\phi(n)/n$ where ϕ is the Euler torsion function [4]. However, the cryptosystem proposed by Dijk-Woodruff is not so efficient as RSA with the key length in practical applications. In 2005 Dijk *et al*. further proposed a relatively efficient public-key cryptosystem using a torus over $\mathbb{F}_{q^{30}}$ whose compression ratio is $\phi(30)/30 = 4/15$

Manuscript received January 28, 2008.
Manuscript revised April 20, 2008.

[†]The authors are with Future University Hakodate (FUN), Hakodate-shi, 041-8655 Japan.
[††]The authors are with Electronics and Telecommunications Research Institute (ETRI), Korea.
[†††]The author is with Japan Advanced Institute of Science and Technology (JAIST), Nomi-shi, 923-1292 Japan.

*The preliminary version of this paper [15] was published at 5th International Conference Applied Cryptography and Network Security, ACNS 2007.

a) E-mail: shirase@fun.ac.jp

DOI: 10.1093/ietfec/e91–a.10.2843

[5]. It is an open problem to construct a practical public-key cryptosystem whose compression ratio is smaller than 4/15.

1.1 Contribution of This Paper

In this paper we present a greatly improved version of XTR that leads to a factor 6 reduction in the representation size compared to the traditional representation. That is to say, we achieve a factor 2 reduction compared to the original XTR. We show that if the characteristic of q is three, *i.e.*, $q = 3^{2k-1}$ for some integer k, then we can use the trace over $\mathbb{F}_q$ to represent elements of the order $q - \sqrt{3q} + 1$ subgroup of $\mathbb{F}^*_{q^6}$. Also, the resulting calculations such as exponentiations are as faster as that of XTR. Given $Tr_{(q^6,q)}(g)$ and n, $Tr_{(q^6,q)}(g^n)$ takes about 1291 multiplications in $\mathbb{F}_q$, which is only about 1% increase compared to the cost of computation of $Tr_{(q^6,q^2)}(h^n)$ for given $Tr_{(q^6,q^2)}(h)$ and n, where the size of n is 160 bits. Therefore, the proposed scheme is one of the fastest public-key cryptosystems with the smallest compression ratio (*i.e.*, 1/6).

In Sect. 2 we describe XTR, and in Sect. 3 we introduce XTR over characteristic three, which achieves a factor 2 reduction in the representation size compared to XTR. Section 4 shows efficient calculations of XTR exponentiation over characteristic three. Applications and comparisons to the original XTR are given in Sect. 5. We then describe conclusion in Sect. 6.

2. XTR

2.1 Description of XTR

XTR uses a subgroup of prime order p of the order q^2-q+1 subgroup of $\mathbb{F}^*_{q^6}$. The latter group is referred to as the *XTR supergroup* denoted as G_{q^2-q+1} and the order p subgroup G_p is referred to as the *XTR group*. The XTR supergroup G_{q^2-q+1} is not contained in any proper subfield of $\mathbb{F}_{q^6}$ due to the following fact.

Fact 1: [10] Let p be a prime factor of $\Phi_m(q)$, where m-th cyclotomic polynomial for a positive integer m not divisible by q. Then the subgroup G_p of $\mathbb{F}^*_{q^m}$ is not contained in any proper subfield of $\mathbb{F}_{q^m}$.

Combined with the choice of p it follows that computing discrete logarithms in G_p is as hard, in general, as it is in $\mathbb{F}^*_{q^6}$ [9].

Before describing XTR more detail, we introduce two definitions about optimal normal basis.

Definition 1: Type I Optimal Normal Basis (Type-I ONB) If $m+1$ is a prime and q is a generator of $\mathbb{F}^*_{m+1}$, then the set $\{\omega^m, \omega^{m-1}, \cdots, \omega^2, \omega\}$ forms an optimal normal basis of type I in $\mathbb{F}_{q^m}$ and called Type-I ONB. Here, ω is the primitive $(m+1)$-th root of unity.

Definition 2: Type II Optimal Normal Basis (Type-II ONB)
If $2m + 1$ is a prime and either of the following two conditions holds,

- q is a primitive root modulo $2m + 1$,
- q is a quadratic residue modulo $2m + 1$ and $q \not\equiv 1 \ mod \ (2m + 1)$,

then the set $\{\beta^m, \beta^{m-1}, \cdots, \beta^2, \beta\}$ forms an optimal normal basis of type II in $\mathbb{F}_{q^m}$ and called Type-II ONB. Here, $\beta = \gamma + \gamma^{-1}$ and γ is the primitive $(2m+1)$-th root of unity.

XTR uses $\mathbb{F}_{q^2}$ arithmetic to achieve $\mathbb{F}_{q^6}$ security, without requiring explicit construction of $\mathbb{F}_{q^6}$. Let q be a prime that is 2 mod 3. It follows that $(X^3 - 1)/(X - 1) = X^2 + X + 1$ is irreducible over $\mathbb{F}_q$ and the zeros α and α^q of it form an Type-I ONB for $\mathbb{F}_{q^2}$ over $\mathbb{F}_q$. In XTR elements of G_p are represented by their trace over $\mathbb{F}_{q^2}$. For $h \in \mathbb{F}^*_{q^6}$ the trace $Tr_{(q^6,q^2)}(h)$ over $\mathbb{F}_{q^2}$ is defined as the sum of the conjugates over $\mathbb{F}_{q^2}$ of h, *i.e.*, $Tr_{(q^6,q^2)}(h) = h + h^{q^2} + h^{q^4} \in \mathbb{F}_{q^2}$. Let p and q be primes with p dividing $q^2 - q + 1$. Also let h be a generate of G_p and let $c = Tr_{(q^6,q^2)}(h)$. Suggested lengths to provide adequate levels of security are $\log_2(q) \approx 170$ and $\log_2(p) \approx 160$.

c_n denotes $Tr_{(q^6,q^2)}(h^n) \in \mathbb{F}_{q^2}$, for some q and h of order p dividing $q^2 - q + 1$ as above. Efficient computation of c_n given q, p and c depends on the recurrence relation

$$c_{u+v} = c_u c_v - c_v^q c_{u-v} + c_{u-2v}, \tag{1}$$

for $u, v \in Z$. It simplifies for $u = v$ to

$$c_{2u} = c_u^2 - 2c_u^q. \tag{2}$$

In [9], Lenstra and Verheul proved that computing c_{u+v} and c_{2u} take four and two multiplications in $\mathbb{F}_q$ respectively, when c_u, c_v, c_{u-v}, and c_{u-2v} are given.

XTR Exponentiation ([9], Algorithm 2.3.7)

INPUT: c and n where $n > 2$
OUTPUT: c_n

1. Compute initial values:
   1.1. $C_3 \leftarrow c$, $C_0 \leftarrow D[C_3]$, $C_1 \leftarrow A[C_0, C_3, C_3, 3]$, and $C_2 \leftarrow D[C_0]$
   1.2. If n is even, n replace $n - 1$.
   Let $n = 2m + 1$ and $m = \sum_{j=0}^{l} m_j 2^j$ with $m_j \in \{0, 1\}$ and $m_l = 1$.
2. for $j = l - 1$ down to 0
   2.1. $T_1 \leftarrow D[C_{m_j}]$
   2.2. $T_2 \leftarrow D[C_{1+m_j}]$
   2.3. if $(m_j = 0)$ then $T_3 \leftarrow A[C_0, C_1, C_3^q, C_2^q]$
   if $(m_j = 1)$ then $T_3 \leftarrow A[C_2, C_1, C_3, C_0^q]$
   2.4. $C_0 \leftarrow T_1$
   2.5. $C_1 \leftarrow T_3$
   2.6. $C_2 \leftarrow T_2$
3. If n is odd then return C_1
   else return C_2

2.2 XTR Exponentiation

In XTR, an algorithm for computing $Tr_{(q^6,q^2)}(h^n)$ given

$Tr_{(q^6,q^2)}(h)$ and a scalar $n \in Z$ is needed like the algorithm for computing h^n in public-key cryptosystems based on discrete logarithm problem. By using two formula (1), (2) above, we define the following two functions called as XTR addition and XTR doubling respectively;

$$A[u, v, w, z] = u \cdot v - v^q \cdot w + z,$$
$$D[u] = u^2 - 2u^q.$$

Theorem 1: ([9], Theorem 2.3.8) Let c and a positive integer n be given. Computing the sum c_n of the n^{th} powers of the roots takes $8\log_2(n)$ multiplications in $\mathbb{F}_q$.

Thus, given the representation $Tr_{(q^6,q^2)}(h) \in \mathbb{F}_{q^2}$ of the conjugates of h, the representation $Tr_{(q^6,q^2)}(h^n) \in \mathbb{F}_{q^2}$ of the conjugates of the n^{th} power of h can be computed at the cost of $8\log_2(n)$ multiplications in $\mathbb{F}_q$, for any integer n.

Denote the above XTR exponentiation with input c and n outputs c_n as

$$\mathsf{XTR\_Exp}[c, n] = c_n.$$

2.3 XTR-DH Key Agreement

XTR can be used in any cryptosystem that relies on the discrete logarithm problem. This section contains a description of an application of XTR that provides confidentiality service, for example Diffie-Hellman key agreement.

Public Parameters : q, p, $c = Tr_{(q^6,q^2)}(h)$

If Alice and Bob want to agree on a secret key K they do the following.

1. Alice selects at random $a \in Z_p$, uses $\mathsf{XTR\_Exp}[c, a] = c_a \in \mathbb{F}_{q^2}$, and sends c_a to Bob.
2. Bob receives c_a from Alice, selects at random $b \in Z_p$, uses $\mathsf{XTR\_Exp}[c, b] = c_b \in \mathbb{F}_{q^2}$, and sends c_b to Alice.
3. Alice receives c_b from Bob, computes $\mathsf{XTR\_Exp}[c_b, a] = c_{ba}$, and determines K based on $c_{ba} := Tr_{(q^6,q^2)}(h^{ba})$.
4. Bob uses $\mathsf{XTR\_Exp}[c_a, b] = c_{ab}$, and determines K based on $c_{ab} := Tr_{(q^6,q^2)}(h^{ab})$.

3. XTR over Characteristic Three

The original XTR uses the trace over $\mathbb{F}_{q^2}$ to represent elements of the order q^2-q+1 subgroup of $\mathbb{F}^*_{q^6}$, thereby achieving a factor 3 size reduction. This section shows that if q is 3 to the odd power then elements in G_{q^2-q+1} can be represented as elements in $\mathbb{F}_q$ using the trace over $\mathbb{F}_q$. It achieves a factor 6 size reduction, which is the half size reduction compared to the original XTR representation.

3.1 New XTR Group

We assume that $q = 3^t$ for any odd integer t, say $t = 2k - 1$. Then, $\sqrt{3q} = 3^k$ is an integer and $q^2 - q + 1$ is factorized as

$$q^2 - q + 1 = (q + \sqrt{3q} + 1)(q - \sqrt{3q} + 1).$$

In this section, we define a new XTR group $G_p = <g>$ which is a subgroup of $G_{q-\sqrt{3q+1}}$, namely, XTR uses a subgroup of prime order p of the order $q - \sqrt{3q} + 1$ subgroup of $\mathbb{F}^*_{q^6}$. The order p subgroup $< g >$ generated by g is referred as the *New XTR group*. Since p does not divide any $q^s - 1$ for $s = 1, 2, 3$, the new XTR group G_p generated by g cannot be embedded in the multiplicative group of any true subfield of $\mathbb{F}_{q^6}$. Combined with the choice of p it follows that computing discrete logarithms in G_p is as hard, in general, as it is in $\mathbb{F}^*_{q^6}$ (cf. [9], Section 5).

$$G_p =< g > \ \lhd G_{q-\sqrt{3q+1}} \ \lhd G_{q^2-q+1} \ \lhd G_{q^3+1} \tag{3}$$

Here, $A \lhd B$ denotes A is a subgroup of B.

From $q = 3^t$ and t is odd it follows q is a generator of $\mathbb{F}^*_5$, so that $\{\omega + \omega^{-1}, \omega^2 + \omega^{-2}\}$ form an Type-II ONB for $\mathbb{F}_{q^2}$ over $\mathbb{F}_q$, where ω is a root of the polynomial $(X^5 - 1)/(X - 1) = X^4 + X^3 + X^2 + X + 1$. For the simplicity, we denote $x = x_1 \cdot (\omega + \omega^{-1}) + x_2 \cdot (\omega^2 + \omega^{-2}) \in \mathbb{F}_{q^2}$ as (x_1, x_2).

Lemma 1: Let $x, y, z \in \mathbb{F}_{q^2}$ with $q = 3^t$ and t is odd.

i. Computing x^q is for free.
ii. Computing x^2 takes two multiplications in $\mathbb{F}_q$.
iii. Computing $x * z - y * z^q$ takes four multiplications in $\mathbb{F}_q$.

Proof 1: Let $x = (x_1, x_2)$, $y = (y_1, y_2)$ and $z = (z_1, z_2) \in \mathbb{F}_{q^2}$ for x_i, y_i, $z_i \in \mathbb{F}_q$, $i \in \{1, 2\}$. From $(\omega + \omega^{-1})^q = \omega^2 + \omega^{-2}$ and $(\omega^2 + \omega^{-2})^q = \omega + \omega^{-1}$, $x^q = (x_2, x_1)$. It follows that q^{th} powering in $\mathbb{F}_{q^2}$ does not require arithmetic operations and can thus be considered to be for free.

From $x^2 = ((x_1 + x_2)(x_1 - x_2) + 2x_1x_2, 2(x_1 + x_2)(x_1 - x_2) + 2x_1x_2)$, x^2 is obtained from two multiplications in $\mathbb{F}_q$.

Finally, to compute $x * z - y * z^q$ four multiplications in $\mathbb{F}_q$ suffice, because it is easily verified that

$$x * z - y * z^q = ((x_2 - 2x_1 + y_2 - y_1) * z_1 + (x_1 - x_2 + 2y_1 - y_2) * z_2) * (\omega + \omega^{-1}) + ((x_2 - x_1 + 2y_2 - y_1) * z_1 + (x_1 - 2x_2 + y_1 - y_2) * z_2) * (\omega^2 + \omega^{-2}).$$

3.2 Compression and Restoration

For some q and g of order p dividing $q - \sqrt{3q} + 1$, define d and e as the trace $Tr_{(q^6,q^2)}(g)$ over $\mathbb{F}_{q^2}$ and the trace $Tr_{(q^6,q)}(g)$ over $\mathbb{F}_q$, respectively. We use the shorthand $d_n = Tr_{(q^6,q^2)}(g^n)$ and $e_n = Tr_{(q^6,q)}(g^n)$, *i.e.*, e_n and d_n are the sum of the conjugates over $\mathbb{F}_{q^2}$ and $\mathbb{F}_q$ of g^n respectively. Immediately, $d = d_1$ and $e = e_1$.

$$\begin{aligned} d_n &= Tr_{(q^6,q^2)}(g^n) = g^n + g^{nq^2} + g^{nq^4} \in \mathbb{F}_{q^2} \\ e_n &= Tr_{(q^6,q)}(g^n) \\ &= g^n + g^{nq} + g^{nq^2} + g^{nq^3} + g^{nq^4} + g^{nq^5} \in \mathbb{F}_q. \end{aligned}$$

3.2.1 Compression

From the definition of d_n and e_n, e_n can be easily derived

from d_n due to the following equation

$$e_n = d_n + d_n^q. \tag{4}$$

For any $d_n = x_1(\omega + \omega^{-1}) + x_2(\omega^2 + \omega^{-2}) \in \mathbb{F}_{q^2}$, we have that $e_n = (x_1 + x_2) * (\omega + \omega^{-1}) + (x_1 + x_2) * (\omega^2 + \omega^{-2}) \in \mathbb{F}_q$ because of Lemma 1-*i*. Note that as $d_n \in \mathbb{F}_{q^2}$ and $d_n \notin \mathbb{F}_q$, $x_1 \neq x_2$, where x_1 and x_2 are in $\mathbb{F}_q$.

Define a compression function with input an element of $\mathbb{F}_{q^2}$ represented by two elements of $\mathbb{F}_q$, say (x_1, x_2), outputs an element of $\mathbb{F}_q$.

$$\mathsf{Compression}[x_1, x_2] = x_1 + x_2.$$

3.2.2 Restoration

Contrary to the compression from d_n to e_n, this section explains how to get d_n from e_n, called it as restoration in this paper.

Lemma 2: The roots of $X^2 - e_n X + e_n^{\sqrt{3q}} \in \mathbb{F}_q[x]$ are d_n and d_n^q.

Proof 2: It is sufficient to prove $d_n * d_n^q = e_n^{\sqrt{3q}}$ because $d_n + d_n^q = e_n$ from equation (4). For simplicity, we prove $d_n * d_n^q = e_n^{\sqrt{3q}}$ when $n = 1$.

$$\begin{aligned} d * d^q &= (g + g^{q^2} + g^{q^4}) \cdot (g^q + g^{q^3} + g^{q^5}) \\ &= g^{1+q} + g^{1+q^3} + g^{1+q^5} + g^{q^2+q} + g^{q^2+q^3} \\ &\quad + g^{q^2+q^5} + g^{q^4+q} + g^{q^4+q^3} + g^{q^4+q^5} \\ &= g^{\sqrt{3q}} + (g^q)^{\sqrt{3q}} + (g^{q^2})^{\sqrt{3q}} + (g^{q^3})^{\sqrt{3q}} \\ &\quad + (g^{q^4})^{\sqrt{3q}} + (g^{q^5})^{\sqrt{3q}} + 3 \\ &= (g + g^q + g^{q^2} + g^{q^3} + g^{q^4} + g^{q^5})^{\sqrt{3q}} \\ &= e^{\sqrt{3q}}. \end{aligned}$$

The third equality is derived from the series of subgroups in equation (3), that is to say, $g^{q+1} = g^{\sqrt{3q}}$ (from $< g > \lhd G_{q-\sqrt{3q}+1}$) and $g^{q^3+1} = 1$ (from $< g > \lhd G_{q^3+1}$).

From Lemma 2, we can find two roots d_n and d_n^q by solving the quadratic formula, which are

$$\{d_n,\ d_n^q\} = \frac{e_n \pm \sqrt{e_n^2 - 4e_n^{\sqrt{3q}}}}{2}. \tag{5}$$

Let $e_n = z \in \mathbb{F}_q$ and the roots of the quadratic equation be $\{(x_1, x_2), (x_2, x_1)\}$, where $x_1, x_2 \in \mathbb{F}_q$ and $x_1 \neq x_2$. Actually, $z = x_1 + x_2$. Define a restoration function with input e_n, outputs $\{d_n, d_n^q\} \in \mathbb{F}_{q^2}$.

$$\mathsf{Restoration}[e_n] = \{d_n, d_n^q\}.$$

Remark 1: In Sec. 3, we have explained how to compute e_n from e_1 and any integer n. In some cryptographic protocols, we also need to compute e_{mn} from e_m and any integer n, where m is an integer arbitrary selected. We can compute it in the same way in XTR. To show that, it is enough to show $< g > = < g^m >$ and g^m has the same order as g, because e_m is $Tr_{q^6,q}(g^m)$ by the definition.

$< g^m > \subset < g >$: Any element contained in $< g^m >$ is represented as $(g^m)^k$ for an integer k. It is equal to $g^{(mk)}$, then it is contained in $< g >$. Consequently $< g^m > \subset < g >$ was shown.

$< g > \subset < g^m >$: Any element contained in $< g >$ is represented as g^k for an integer k. Note that $m \not\equiv 0 \bmod p$ due to the order of g is a prime p and $g^m \neq 1$. Then m is coprime to p. Therefore there are integers α and β such that $n\alpha + p\beta = 1$, which means $k = (m\alpha + p\beta)k$. Note that $g^p = 1$ because p is the order of g. Therefore $g^k = g^{(m\alpha+p\beta)k} = (g^m)^{\alpha k}$ and g^k is contained in $< g^m >$. Consequently $< g > \subset < g^m >$ was shown.

g^m has the same order as g: Because $< g > = < g^m >$ and the order of g is a prime p.

4. Efficient Method of Restoration — Finding d_n and d_n^q from e_n

As we have looked around at the previous section, we need to solve the quadratic formula described in Lemma 2 to extract d_n and d_n^q from e_n. In other words, we have to compute the square root extraction $\sqrt{e_n^2 - 4e_n^{\sqrt{3q}}}$.

In a finite field $\mathbb{F}_{r^s}$ where $r \equiv 3\ (mod\ 4)$ and odd s, the best algorithm known [3], [11] to compute a squire root executes $O(s \log_2 r)$ multiplications in $\mathbb{F}_{r^s}$. By that method, a solution of $X^2 = A$ is given by $X = A^{\frac{r^s+1}{4}}$, assume that A is a quadratic residue. Recently, Barreto *et al.* (c.f. [1], Section 4) presented an improvement to it. The complexity is reduced to $O(\log_2 s + \log_2 r)$ multiplications in $\mathbb{F}_{r^s}$. If the characteristic r is fixed and small compared to s, the complexity is simply $O(\log_2 s)$.

4.1 Square Root Extraction

Let $R = e_n^2 - 4e_n^{\sqrt{3q}} \in \mathbb{F}_q$. Here, $q = 3^{2k-1}$ for any inter k. As d_n or $d_n^q \notin \mathbb{F}_q$ $\sqrt{R}$ is not an element of $\mathbb{F}_q$. Thus, we cannot utilize Barreto *et al.*'s method directly to compute square root of R even if $q \equiv 3\ (mod\ 4)$.

Fact 2: -1 has a square root in $\mathbb{F}_q$ if and only if $q \equiv 1\ (mod\ 4)$.

As $q \equiv 3\ (mod\ 4)$, $\sqrt{-1} \notin \mathbb{F}_q$, but in $\mathbb{F}_{q^2}$.

Lemma 3: $\sqrt{-R} \in \mathbb{F}_q$, where $-R = 2e_n^2 + e_n^{\sqrt{3q}}$.

Proof 3: Let $\mathbb{F}_q^* = < g_1 >$. $\sqrt{g_1^n} = g_1^{n/2} \in \mathbb{F}_q$ if n is even and $\sqrt{g_1^n}$ is not in $\mathbb{F}_q$ if n is odd for $g_1^n \in \mathbb{F}_q^*$. From $(g_1^{(q-1)/2})^2 = 1$ and g_1 is a generator of $\mathbb{F}_q$, $g_1^{(q-1)/2} = -1$. We confirm easily that $(q-1)/2$ is odd if $q = 3^{2k-1}$. Then we see that $R = g_1^{n_1}$ for some odd n_1 since $\sqrt{R} \notin \mathbb{F}_q$. Hence $-R = R \cdot (-1) = g_1^{n_1+(q-1)/2}$ and $n_1 + (q-1)/2$ is even. Therefore, $\sqrt{-R} \in \mathbb{F}_q$.

From Lemma 3, one of $\sqrt{-R}$ is $(-R)^{\frac{q+1}{4}}$ and it is efficiently computed by using the idea of Barreto *et al.* [1]. The basic idea is as follows.

They noticed that, if $q = 3^{2k-1}$ for some k:

$$\frac{q+1}{4} = \frac{3^{2k-1}+1}{4} = 6 \cdot \sum_{i=0}^{k-2} (3^2)^i + 1,$$

so that

$$(-R)^{(q+1)/4} = \left[((-R)^2)^{\sum_{i=0}^{k-2}(3^2)^i}\right]^3 \cdot (-R).$$

The quantity $((-R)^2)^{\sum_{i=0}^{k-2}(3^2)^i}$ is efficiently computed in an analogues fashion to Itoh-Teechai-Tsujii inversion [8], based on the Frobenius map in characteristic three. Let $A \in \mathbb{F}_q$. Then, one can compute $A^{\sum_{i=0}^{k-2}(3^2)^i}$ with no more than $\lfloor \log_2(k-1) \rfloor + HW(k-1) - 1$ multiplications in $\mathbb{F}_q$. Here, $\lfloor \cdot \rfloor$and $HW(\cdot)$ denote the maximum integer less than its operand and the Hamming weight of its operand respectively. Thus, we need at most $\lfloor \log_2(k-1) \rfloor + HW(k-1) + 1$ multiplications in $\mathbb{F}_q$ to compute $(-R)^{(q+1)/4}$ in total.

Next we must find $\sqrt{-1} \in \mathbb{F}_{q^2}$ to compute $\sqrt{R} = \sqrt{-R} \cdot \sqrt{-1}$.

Lemma 4: $\sqrt{-1} \in \mathbb{F}_{q^2}$ is $(\omega + \omega^{-1}) - (\omega^2 + \omega^{-2})$ or $-(\omega + \omega^{-1}) + (\omega^2 + \omega^{-2})$.

Proof 4: It is easily checked that $\left((\omega+\omega^{-1})-(\omega^2+\omega^{-2})\right)^2 = -1$ because $1 + \omega + \omega^2 + \omega^3 + \omega^4 = 0$ and $\omega^5 = 1$.

Lemma 5: For any $x = x_1(\omega + \omega^{-1}) + x_2(\omega^2 + \omega^{-2}) \in \mathbb{F}_{q^2}$, $x \cdot \sqrt{-1}$ is free.

Proof 5: Because $\left(x_1(\omega + \omega^{-1}) + x_2(\omega^2 + \omega^{-2})\right) \cdot \sqrt{-1} = -x_2(\omega + \omega^{-1}) + x_1(\omega^2 + \omega^{-2})$.

4.2 Computation of d_n and d_n^q

Thanks to the Eq. (5) and the results of the previous section, for given $e_n \in \mathbb{F}_q$

$$\begin{aligned} \{d_n, d_n^q\} &= \frac{e_n \pm \sqrt{R}}{2} \\ &= 2 \cdot (e_n \pm \sqrt{-R} \cdot \sqrt{-1}) \\ &= 2e_n \pm (2e_n^2 + e_n^{\sqrt{3q}})^{\frac{q+1}{4}} \cdot \sqrt{-1}. \end{aligned} \quad (6)$$

Table 1 shows the number of multiplications in $\mathbb{F}_q$ required to compute equation (6), where as customary we do not count the cost of additions and subtractions in $\mathbb{F}_q$.

For efficient computation of $\sqrt{3q}$-th power of e_n ($\in \mathbb{F}_q$), *i.e.*, $e_n^{\sqrt{3q}}$, we should select q such that $\mathbb{F}_q$ has optimal normal basis (ONB) over $\mathbb{F}_3$. As $q = 3^{2k-1}$, $\sqrt{3q} = 3^k$. Thus, $\sqrt{3q}$-th power is performed by shift of coefficients when $\mathbb{F}_q$ has ONB over $\mathbb{F}_3$. However, $\mathbb{F}_q$ never has Type-I ONB over $\mathbb{F}_3$ since $2k$ is not prime. Therefore, we should check whether $\mathbb{F}_q$ has Type-II ONB over $\mathbb{F}_3$ or not for given k. For example, we may select $k = 56, 71, 111$, and 120, which satisfy that $\mathbb{F}_q$ has Type-II ONB over $\mathbb{F}_3$.

Table 1 The number of multiplications in $\mathbb{F}_q$ for computation of d_n and d_n^q, where $q = 3^t$ and $t = 2k - 1$ for some integer k.

Operation	# of multiplications in $\mathbb{F}_q$
e_n^2	1
$e_n^{\sqrt{3q}}$	free using Type-II ONB
$(2e_n^2 + e_n^{\sqrt{3q}})^{\frac{q+1}{4}}$	$\lfloor \log_2(k-1) \rfloor + HW(k-1) + 1$
$(2e_n^2 + e_n^{\sqrt{3q}})^{\frac{q+1}{4}} \cdot \sqrt{-1}$	0
$2e_n \pm (2e_n^2 + e_n^{\sqrt{3q}})^{\frac{q+1}{4}} \cdot \sqrt{-1}$	$\lfloor \log_2(k-1) \rfloor + HW(k-1) + 2$

Note that a multiplication $(2e_n^2 + e_n^{\sqrt{3q}})^{\frac{q+1}{4}} * \sqrt{-1}$ is free from Lemma 5.

Theorem 2: Given e_n for any integer n, computing d_n and d_n^q take about $\lfloor \log_2(k-1) \rfloor + HW(k-1) + 2$ multiplications in $\mathbb{F}_q$ under assumption that $\mathbb{F}_q$ has Type-II ONB over $\mathbb{F}_3$.

5. Compressed XTR Exponentiation

In this section it is shown how e_n can be computed based on e_1 and an arbitrary integer n.

Restoration - compute d_1 and d_1^q from Restoration$[e_1]$. Between $\{d_1, d_1^q\}$ choose one of them at random, denoted d'.
XTR exponentiation - compute d'_n from XTR_Exp$[d', n]$ described in Sect. 2.2.
Compression - compute Compression$[d'_n] = d'_n + (d'_n)^q$. Actually, Compression$[d'_n] = e_n$.

At the compression step, we can easily check $d'_n + (d'_n)^q = e_n$. d' is one of $\{d_1, d_1^q\}$. If $d' = d_1$ then it is trivial because of the definition of e_n. Otherwise, *i.e.*, $d' = d_1^q$ then $d'_n + (d'_n)^q = d_n^q + d_n$ because $d_n \in \mathbb{F}_{q^2}$, which concludes the justification of the compression step.

Denote the above XTR exponentiation over characteristic three with input e_1 and n outputs e_n as

$$\mathsf{XTR\_Exp}_3[e_1, n] = e_n.$$

Theorem 3: Let e_1 and a positive integer $n \in Z_p$ be given. Assume that $\mathbb{F}_q$ has Type-II ONB over $\mathbb{F}_3$. Then, computing e_n takes about $8 \log_2(n) + \lfloor \log_2(k-1) \rfloor + HW(k-1) + 2$ multiplications in $\mathbb{F}_q$.

Proof 6: Immediate from Theorem 2, XTR Exponentiation algorithm ([9], Algorithm 2.3.7), and Lemma 1.

5.1 Application to XTR-DH

In this section we describe XTR version Diffie-Hellman key agreement over characteristic three.

Public Parameters : $q(= 3^{2k-1})$, p, $Tr_{(q^6,q)}(g) := e$

Suppose that Alice and Bob who both have access to the XTR public key data, want to agree on a shared secret key K. This can be done using the following XTR version.

1. Alice selects at random $a \in Z_p$, uses $\mathsf{XTR\_Exp}_3[e, a] = e_a \in \mathbb{F}_q$, and sends e_a to Bob.
2. Bob receives e_a from Alice, selects at random $b \in Z_p$, uses $\mathsf{XTR\_Exp}_3[e, b] = e_b \in \mathbb{F}_q$, and sends e_b to Alice.
3. Alice receives e_b from Bob, computes $\mathsf{XTR\_Exp}_3[e_b, a] = e_{ba}$, and determines K based on $e_{ba} = Tr_{(q^6,q)}(g^{ba})$.
4. Bob uses $\mathsf{XTR\_Exp}_3[e_a, b] = e_{ab}$, and determines K based on $e_{ab} = Tr_{(q^6,q)}(g^{ab})$.

5.2 Comparison to Original XTR

In this section, we compare XTR over characteristic three to the original XTR. Let $\mathbb{XTR}$ and $\mathbb{XTR}_3$ denote the original XTR [9] and XTR over characteristic three respectively.

5.2.1 XTR Group G_p

$\mathbb{XTR}$: XTR group $G_p =<h>$ is a subgroup of G_{q^2-q+1}, where $h \in \mathbb{F}^*_{q^6}$.

- p and q are prime, and $q \equiv 2$ (*mod* 3).

$\mathbb{XTR}_3$: XTR group $G_p =< g >$ is a subgroup of $G_{q-\sqrt{3q+1}}$, where $g \in \mathbb{F}^*_{q^6}$.

- p is prime and $q = 3^{2k-1}$.

Note that suggested lengths to provide adequate levels of security are $\log_2(q) \approx 170$ and $\log_2(p) \approx 160$.

5.2.2 XTR Exponentiation

$\mathbb{XTR}$: For given $Tr_{(q^6,q^2)}(h)$ and $n \in Z_p$ computing $Tr_{(q^6,q^2)}(h^n)$ takes $8\log_2(n)$ multiplications in $\mathbb{F}_q$.

- $\mathbb{F}_{q^2}$ has Type-I ONB over $\mathbb{F}_q$.

$\mathbb{XTR}_3$: For given $Tr_{(q^6,q)}(g)$ and $n \in Z_p$ computing $Tr_{(q^6,q)}(g^n)$ takes $8\log_2(n)+\lfloor\log_2(k-1)\rfloor+HW(k-1)+2$ multiplications in $\mathbb{F}_q$.

- $\mathbb{F}_{q^2}$ has Type-II ONB over $\mathbb{F}_q$.
- $\mathbb{F}_q$ has Type-II ONB over $\mathbb{F}_3$.

Denote by $|\cdot|$ the bit length of " $\cdot$ ". In the proposed $\mathbb{XTR}_3$, we have to select k such that $\mathbb{F}_q$ has Type-II ONB over $\mathbb{F}_3$, and both order $|p|$ of the subgroup and order $|q^6|$ of the whole group are large enough. Therefore, we cannot construct the proposed $\mathbb{XTR}_3$ with arbitrary size of p unlike the original $\mathbb{XTR}$. The security of 1024 bits (or 2048 bits) RSA cryptosystem corresponds to that of the discrete logarithm problem in the 160 bits (or 224 bits) subgroup, respectively [12].

In order to estimate the efficiency of the proposed $\mathbb{XTR}_3$, we try to choose several ks in the following. $k = 56$ provides $(|p|, |q^6|)$ closest to $(160, 1024)$, namely $(|p|, |q^6|) = (156, 1056)$, however this $|p|$ is a bit smaller than 160. $k = 71$ is the smallest k such that $|p| \geq 160$ and $|q^6| \geq 1024$. $k = 111$ provides $(|p|, |q^6|)$ closest to $(224, 2048)$, namely $(|p|, |q^6|) = (225, 2101)$, and that also satisfies $|p| \geq 224$ and $|q^6| \geq 2048$. $k = 120$ corresponds to a larger parameter $(|p|, |q^6|) = (378, 2273)$.

Table 2 Suitable k and costs of $\mathbb{XTR}$ and $\mathbb{XTR}_3$.

k	$\lvert p\rvert$	$\lvert q^6\rvert$	Cost of $\mathbb{XTR}$	Cost of $\mathbb{XTR}_3$
56	156	1056	1248	1260
71	193	1341	1544	1555
111	225	2101	1792	1806
120	378	2273	3024	3038

Table 3 The sizes of public key data of $\mathbb{XTR}$ and $\mathbb{XTR}_3$.

k	$\lvert q\rvert$	$\lvert p\rvert$	Public key data size of $\mathbb{XTR}$	Public key data size of $\mathbb{XTR}_3$
56	176	156	684	508
71	224	193	865	641
111	351	225	1278	927
120	379	378	1515	1136

Table 2 shows the cost of $\mathbb{XTR}$ and $\mathbb{XTR}_3$ for the above ks. Note that the cost of $\mathbb{XTR}$ requires $8|p|$ multiplications in $\mathbb{F}_q$, because the size of scalar n is equal to $|p|$. On the other hand, the cost of $\mathbb{XTR}_3$ additionally requires some overheads appeared in Table 1. Here we estimate that the cost of $\mathbb{XTR}_3$ increases by 1% compared with that of $\mathbb{XTR}$ for the above ks.

5.2.3 Communication Overhead

The communication overhead of XTR-DH in $\mathbb{XTR}_3$ is about *half* of XTR-DH proposed in [9] and *one six* of traditional implementations of the Diffie-Hellman protocol that are based on subgroups of multiplicative groups of finite fields, and that achieves the same level of security.

5.2.4 Size of Public Key Parameter

In $\mathbb{XTR}$, the public key data are q, p, and $Tr_{(q^6,q^2)}(h)$. Thus, the total length is $3|q| + |p|$. However, in the case of $\mathbb{XTR}_3$, the public key data are $q(= 3^{2k-1})$, p, and $Tr_{(q^6,q)}(g)$, and the total length of it is $2|q| + |p|$. Table 3 shows the sizes of public key data of $\mathbb{XTR}$ and $\mathbb{XTR}_3$. The size of public key data of $\mathbb{XTR}_3$ is reduced by about 26% compared with $\mathbb{XTR}_3$ for these ks.

6. Conclusion

In this paper we presented a new variant of XTR cryptosystem with a compact representation of the ciphertext. The compression ratio of the ciphertext becomes 1/6, which is the smallest among the previously known practical public-key cryptosystems. The computational overhead of the proposed scheme over the original XTR is only about 1%. Therefore, the proposed scheme is one of the fastest public-key cryptosystems with the smallest compression ratio.

It is a further research topic to construct a practical

public-key cryptosystem that achieves the compression ratio smaller than 1/6.

Acknowledgements

The work reported in this paper was supported by the IT R&D program of MIC/IITA. [2005-S088-04, Development of Security technology for Secure RFID/USN Service].

References

[1] P. Barreto, H.Y. Kim, B. Lynn, and M. Scott, "Efficient algorithms for pairing-based cryptosystems," Crypto 2002, LNCS 2442, pp.354–369, 2002.
[2] A. Brouwer, R. Pellikaan, and E.R. Verheul, "Doing more with fewer bits," Asiacrypt'99, LNCS 1716, pp.321–332, 1999.
[3] H. Cohen, A Course in Computational Algebraic Number Theory, Springer, 1993.
[4] M. van Dijk and D. Woodruff, "Asymptotically optimal communication for torus-based cryptography," Crypto 2004, LNCS 3152, pp.157–178, 2004.
[5] M. van Dijk, R. Granger, D. Page, K. Rubin, A. Silverberg, M. Stam, and D. Woodruff, "Practical cryptography in high dimensional Tori," Eurocrypt 2005, LNCS 3494, pp.234–250, 2005.
[6] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Trans. Inf. Theory, vol.31, no.4, pp.469–472, 1985.
[7] G. Gong and L. Harn, "Public key cryptosystems based on cubic finite field extensions," IEEE Trans. Inf. Theory, vol.45, no.7, pp.2601–2605, 1999.
[8] T. Itoh, O. Teechai, and S. Tsujii, "A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases," Inf. Comput., vol.78, pp.171–177, 1988.
[9] A. Lenstra and E. Verheul, "The XTR public key system," Crypto 2000, LNCS 1800, pp.1–20, 2000.
[10] A. Lenstra, "Using cyclotomic polynomials to construct efficient discrete logarithm cryptosystems over finite fields," ACISP'97, LNCS 1270, pp.127–138, 1997.
[11] A. Menezes, P. van Oorschot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.
[12] National Institute of Standards and Technology, Special Publication 800-56: Recommendation on key establishment schemes, Draft 2.0, 2003.
[13] K. Rubin and A. Silverberg, "Torus-based cryptography," Crypto 2003, LNCS 2729, pp.349–365, 2003.
[14] C. Schnorr, "Efficient signature generation by smart cards," J. Cryptol., vol.4, pp.161–174, 1991.
[15] M. Shirase, D.-G. Han, Y. Hibino, H.W. Kim, and T. Takagi, "Compressed XTR," ACNS 2007, LNCS 4521, pp.420–431, 2007.
[16] P. Smith and C. Skinner, "A public-key cryptosystem and a digital signature system based on the Lucas function analogue to discrete logarithms," Asiacrypt'94, LNCS 917, pp.357–364, 1995.
[17] M. Stam and A. Lenstra, "Speeding up XTR," Asiacrypt 2001, LNCS 2248, pp.125–143, 2001.

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 Postdoctoral in the School of System Science Information at Future University-Hakodate. His research interests are algorithm and implementation of cryptography.

Dong-Guk Han received his B.S. degree in mathematics from Korea University in 1999, and his M.S. degrees in mathematics from Korea University in 2002, respectively. He received Ph.D. of engineering in Information Security from Korea University in 2005. He was a Post.Doc. in Future University-Hakodate, Japan. After finishing the doctor course, he had been an exchange student in Dep. of Computer Science and Communication Engineering in Kyushu University in Japan from April 2004 to March 2005. Now, he is a senior researcher in Electronics and Telecommunications Research Institute (ETRI) from June 2006. He is a member of KIISC, IEEK, and IACR.

Yasushi Hibino is a professor in School of Information Science at Japan Advanced Institute of Science and Technology (JAIST). He received B.S. and M.S. degrees from Tokyo Institute of Technology, Tokyo, 1970 and 1972 respectively, and a Ph.D. degree in computer engineering from same institution in 1995. He worded as a researcher in Electrical Communication Laboratory of Nippon Telegraph and Telephone (Public) Corporation from 1972 to 1992, where he engaged in development of a Lisp Machine ELIS. He joined JAIST in 1993 and his current research is focused on wave pipeline architecture. He is a member of IEEE, ACM and IPSJ.

Howon Kim received his B.S.E.E. degree from KyungPook National University, DaeGu, Korea, in 1993 and the M.S. and Ph.D. degrees in Electronic and Electrical Engineering from Pohang University of Science and Technology (POSTECH), Pohang, Korea, in 1995 and 1999, respectively. From July 2003 to June 2004, he studied at the COSY group at the Ruhr-University of Bochum, Germany. He was a senior member of technical staff at the Electronics and Telecommunications Research Institute (ETRI), DaeJeon, Korea. He is currently working as an assistant professor at the department of computer engineering in Pusan National University, Busan, Korea. His research interests include RFID technology, sensor network, information security and computer architecture. Currently, his main research focus is on mobile RFID technology and sensor network, public key cryptosystem and its security issues. He is a member of the IEEE, IEEE Computer Society, and IACR.

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 Darmstadt in 2001. He was an Assistant Professor in the Department of Computer Science at Technische Universität Darmstadt until 2005. He is currently a Professor in the School of Systems Infomation Science 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).