Implementing Pairing-Based Cryptosystems in USB Tokens

Zhaohui Cheng

Olym Information Security Technology Ltd.
zhaohui_cheng@hotmail.com

Abstract. In the last decade, pairing-based cryptography has been the most intensively studied subject in the cryptography field. Various optimization techniques have been developed to speed up the pairing computation. However, implementing a pairing-based cryptosystem in resource constrained devices has been less tried. Moreover, due to progress on solving the discrete logarithm problem, those implementations are no longer safe to use. In this paper, we report an implementation of a couple of pairing-based cryptosystems at a high security level on a 32-bit microcontroller in a USB token. It shows that USB tokens supporting secure pairing-based cryptosystems are viable.

1 Introduction

Since the seminal work of Joux's tripartite key agreement protocol [22] and Sakai-Ohgishi-Kasahara's non-interactive identity-based key agreement [30], particularly after Boneh-Franklin [6] introduced identity-based encryption with pairings, there have been a flood of innovative work to create various new cryptosystems using pairings, including short signatures [11, 32], identity-based encryptions [5, 7], identity-based signatures [15, 12], attribute-based encryptions [18], etc. We refer the interested reader to the book "Identity-Based Cryptography" [23] for more references.

Meanwhile, because of its great complexity, computing pairing efficiently at proper security levels is also of great interest. During the past ten years, many optimizations to the Miller algorithm [20] have been proposed to speed up cryptographic-friendly pairings [10, 9, 19, 25, 27, 31, 3, 17]. Although, great progress has been made, pairing computation complexity is still much greater than the traditional cryptographic operations such as the elliptic curve point scalar or even the big size integer modular multiplication at the same security level. This explains why most of the work focuses on algorithm implementation on generic CPUs and only a few attempts to compute pairings in resource constrained devices [8, 28, 16, 26] have been reported.

In [8], Bertoni et al. described an implementation of Tate [10] pairing on a supersingular curve over a 512-bit prime base field on a ST22 processor aiming at RSA-1024 bit security level. With the increase of computation power, now the digital society is moving towards a higher security level. It is generally verified that the Tate pairing on supersingular curves defined over prime fields is

not a promising choice for high security levels, as it requires a large base field. Scott et al. attempted to improve the pairing efficiency on this platform by using Eta pairing [9] on a supersingular curve over a 379-bit binary field. This implementation clearly shows better performance and was considered to be easier to scale to higher security levels. Unfortunately recent progress [1] on computing the discrete logarithm problem (DLP) shows it is unwise to use supersingular curves over characteristic 2 or 3 fields to implement pairing-based cryptosystems. In [16], Devegili et al. implemented the Ate pairing [19] on the Barreto-Naehrig curves [13] aiming at the RSA-3072 security level, but it takes approximately 3 seconds to complete the computation. The performance of several implementations on 8-bit or 16-bit microcontrollers such as on MSP430 [26] is even worse because the computation capability of those chips is weaker.

Instead of proposing new hardware implementation for better performance or giving a slow implementation on an unsuitable architecture, we choose a 32-bit microcontroller which is mainly used to make USB tokens for security applications such as executing ECC or RSA algorithms. We report an efficient implementation of a couple of identity-based cryptosystems on the chip. Particularly, we give a full implementation of identity-based encryption scheme SK-KEM [7] and identity-based signature scheme BLMQ-IBS [12], both are standardized in [21]. For the implementation, we not only need to choose proper curve parameters concerning the schemes but also consider various restrictions posed by the chosen chip. We shall make use of the existing mathematic hardware module, which boosts medium size integer modular multiplications, to implement pairing at a high security level. The final product as a secure USB token can complete all the required cryptographic operations in just over half a second.

The paper is organized as follows. In the next section, pairing is briefly introduced. The implemented identity-based cryptosystems are reviewed in Section 3. The chosen chip is described in Section 4. Curve parameters for the cryptosystems are presented in Section 5. The details of implementation and computation performance analysis are reported in Section 6.

2 Pairings

A pairing is a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ where $\mathbb{G}_1, \mathbb{G}_2$ are additive groups and $\mathbb{G}_3$ is a multiplicative group. All three groups have prime order r. For cryptographic purpose, a pairing should be well defined with bilinearity, non-degenerate and easy to compute. In particular, for all $(P, Q) \in \mathbb{G}_1 \times \mathbb{G}_2$ and for all $(a, b) \in \mathbb{Z}_r \times \mathbb{Z}_r$,

$$e([a]P, [b]Q) = e(P, Q)^{ab}.$$

Let $E(\mathbb{F}_p)$ be an elliptic curve defined over field $\mathbb{F}_p$. Let k be the least positive integer such that $r \mid p^k - 1$ and $r^2 \nmid p^k - 1$. Such integer k is called the embedding degree of r with regard to $\mathbb{F}_p$. For every $Q \in E(\mathbb{F}_{p^k})$ and integer s, let $f_{s,Q}$ be the $\mathbb{F}_{p^k}$-rational function with divisor

$$(f_{s,Q}) = s(Q) - ([s]Q) - (s-1)(\infty)$$

where ∞ represents the infinity point.

On elliptic curves several pairings satisfying the cryptographic requirements can be defined. So far the most efficient one is the optimal Ate pairing [19, 31]. The Ate pairing over $\mathbb{G}_2 \times \mathbb{G}_1$ can be defined by

$$a(Q, P) = f_{t-1,Q}(P)^{(p^k-1)/r},$$

where $\mathbb{G}_1 = E[r] \cap \mathrm{Ker}(\pi_p - [1]) = E(\mathbb{F}_p)[r]$ and $\mathbb{G}_2 = E[r] \cap \mathrm{Ker}(\pi_p - [p]) \subseteq E(\mathbb{F}_{p^k})[r]$ with the Frobenius endomorphism $\pi_p : E \to E$ given by $\pi_p(x, y) = (x^p, y^p)$. t is the trace of Frobenius of the curve and $\mathbb{G}_3 = \mathbb{F}^*_{p^k}/(\mathbb{F}^*_{p^k})^r$.

To compute pairing, the Miller algorithm [20] is used to evaluate function $f_{t-1,Q}$ at point P. The complexity of the process is mainly decided by the Miller iterations t. Vercauteren [31] showed the Miller iterations may be further reduced for certain curves to define the optimal Ate pairing. More details of such pairing with some chosen curve parameters will be given in Section 5.

3 Identity-Based Cryptosystems

We implement two identity-based cryptosystems SK-KEM [7] and BLMQ-IBS [12], both use the Sakai-Kasahara key generation algorithm [29] to generate identity private keys. The key generation algorithm proceeds as follow.

Setup $\mathbb{G}_{\mathtt{ID}}(1^k)$**.** On input 1^k, the algorithm works as follows:

1. Generate three cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ of prime order r and a bilinear pairing map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$. Pick random generator $P_1 \in \mathbb{G}^*_1, P_2 \in \mathbb{G}^*_2$.
2. Pick a random $s \in \mathbb{Z}^*_r$ and compute $P_{pub} = [s]P_1$.
3. Pick a cryptographic hash function

$$H_1 : \{0,1\}^* \to \mathbb{Z}^*_r$$

4. Output the master public key $M_{\mathfrak{pk}} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, \hat{e}, P_1, P_2, P_{pub}, H_1)$ and the master secret key $M_{\mathfrak{sk}} = s$.

Extract $\mathbb{X}_{\mathtt{ID}}(M_{\mathfrak{pk}}, M_{\mathfrak{sk}}, \mathtt{ID}_A)$**.** Given an identifer string $\mathtt{ID}_A \in \{0,1\}^*$ of entity A, $M_{\mathfrak{pk}}$ and $M_{\mathfrak{sk}}$, the algorithm returns $D_A = [\frac{1}{s+H_1(\mathtt{ID}_A)}]P_2$.

SK-KEM is an identity-based key encapsulation mechanism with the encapsulate algorithm $\mathbb{E}_{\mathtt{ID-KEM}}$ and decapsulate algorithm $\mathbb{D}_{\mathtt{ID-KEM}}$ as shown in Table 1, where

$$H_2 : \mathbb{G}_3 \to \{0,1\}^n,$$
$$H_3 : \{0,1\}^n \to \mathbb{Z}^*_r,$$
$$H_4 : \{0,1\}^n \to \{0,1\}^{l_d},$$

for data encapsulation mechanism key length l_d and random variable length n.

BLMQ-IBS is an identity-based signature scheme with the **Sign** and **Verify** algorithm as shown in Table 2, where $H_2 : \{0,1\}^* \times \mathbb{G}_3 \to \mathbb{Z}^*_r$ and M is the signed message.

The SK key generation algorithm bases its security on the following the ℓ-SDH assumption [7, 12].

Table 1. SK-KEM

$\mathbb{E}_{\mathtt{ID-KEM}}(M_{\mathtt{pt}}, \mathtt{ID}_A)$	$\mathbb{D}_{\mathtt{ID-KEM}}(M_{\mathtt{pt}}, \mathtt{ID}_A, D_A, \langle C_1, C_2 \rangle)$
– $m \leftarrow \{0,1\}^n$ – $x \leftarrow H_3(m)$ – $Q_A \leftarrow P_{pub} + [H_1(\mathtt{ID}_A)]P_1$ – $C_1 = [x]Q_A$ – $C_2 = m \oplus H_2(\hat{e}(P_1, P_2)^x)$ – $K \leftarrow H_4(m)$ – Return $(K, \langle C_1, C_2 \rangle)$	– $\alpha \leftarrow \hat{e}(C_1, D_A)$ – $m = C_2 \oplus H_2(\alpha)$ – $x \leftarrow H_3(m)$ – $Q_A \leftarrow P_{pub} + [H_1(\mathtt{ID}_A)]P_1$ – $U \leftarrow [x]Q_A$ – If $C_1 \neq U$, return $\perp$ – $K \leftarrow H_4(m)$ – Return K

Table 2. BLMQ-IBS

Sign$(M_{\mathtt{pt}}, D_A, M)$	**Verify**$(M_{\mathtt{pt}}, \mathtt{ID}_A, M, \langle h, S \rangle)$
– $x \leftarrow \mathbb{Z}_r^*$ – $y \leftarrow \hat{e}(P_1, P_2)^x$ – $h \leftarrow H_2(M, y)$ – $S = [x + h]D_A$ – Return $\langle h, S \rangle$	– $Q_A \leftarrow P_{pub} + [H_1(\mathtt{ID}_A)]P_1$ – $y' \leftarrow \hat{e}(Q_A, S) \cdot \hat{e}(P_1, P_2)^{-h}$ – $h' \leftarrow H_2(M, y')$ – If $h \neq h'$, return Failed – Return OK

Assumption 1 **ℓ-Strong Diffie-Hellman (ℓ-SDH)** *Let P be an element of prime order r in an Abelian group. For a positive integer ℓ, and $\alpha \leftarrow \mathbb{Z}_r^*$, given $([\alpha]P, [\alpha^2]P, \ldots, [\alpha^\ell]P)$, computing $(h, [\frac{1}{\alpha+h}]P)$ for some $h \in \mathbb{Z}_r^*$ is hard.*

Cheon presented an algorithm showing that the computational complexity of ℓ-SDH can be reduced by $O(\sqrt{\ell})$ from that of the DLP in the group [14]. Here we restate Cheon's two main results:

Theorem 1 *Let P be an element of prime order r in an Abelian group. Suppose that ℓ is a positive divisor of $r-1$. If $P, P_1 = [\alpha]P$ and $P_\ell = [\alpha^\ell]P$ are given, α can be computed in $O(\log r \cdot (\sqrt{(r-1)/\ell} + \sqrt{\ell}))$ group operations using $O(\max\{\sqrt{(r-1)/\ell}, \sqrt{\ell}\})$ memory.*

Theorem 2 *Let P be an element of prime order r in an Abelian group. Suppose that ℓ is a positive divisor of $r+1$. If $P_i = [\alpha^i]P$ for $i = 1, 2, \ldots, 2\ell$ are given, α can be computed in $O(\log r \cdot (\sqrt{(r+1)/\ell} + \ell))$ group operations using $O(\max\{\sqrt{(r+1)/\ell}, 2\ell\})$ memory.*

We assume that the adversary can obtain private keys because the adversary may be the user of the system and may also compromise other users' private

keys. By assuming an adversary $\mathcal{A}$ has gathered ℓ different pairs of public/private keys $(h_i, \frac{1}{s+h_i}P_2)$ with $h_i = H_1(\mathtt{ID}_i)$, using Cheon's algorithm $\mathcal{A}$ can recover the master secret key in the following manner.

- Randomly sample $h_0 \in \mathbb{Z}_r^*$ different from h_i for $1 \leq i \leq \ell$.
- Set $\alpha = s + h_0$ which $\mathcal{A}$ does not know, and

$$Q = [\frac{1}{(s+h_1)\cdots(s+h_\ell)}]P_2.$$

- For $j = 0, \ldots, \ell - 1$, $\mathcal{A}$ computes

$$[\alpha^j]Q = [\frac{(s+h_0)^j}{(s+h_1)\cdots(s+h_\ell)}]P_2 = [\sum_{i=1}^{\ell} \frac{c_{ij}}{s+h_i}]P_2$$

  where $c_{ij} \in \mathbb{Z}_r$ are computable from h_i's.
- Given $[\alpha^j]Q$ for $0 \leq j \leq \ell - 1$, use Cheon's algorithm to compute α and so $s = \alpha - h_0$.

By Theorem 2 the complexity of the above attack to recover the master secret key s in SK-KEM/BLMQ-IBS is with $O(\log r \cdot (\sqrt{2(r+1)/\ell} + \ell/2))$ group operations using $O(\max\{\sqrt{2(r+1)/\ell}, \ell\})$ memory. Note that if there is a divisor d of $r-1$ with $d \leq \ell$ and $d \approx \ell$, then the attack is still working with similar computational complexity by Theorem 1.

To defend Cheon's attack on SK key generation algorithm, it is better to have both $r-1$ and $r+1$ with small divisors as few as possible. In Section 5, we show such curve parameters can be found.

4 The AisinoChip's AC4384 Architecture

AC4384 is a microcontroller supplied by AisinoChip [2]. The chip has a 32-bit RISC core designed specifically for secure applications including USB tokens or smart cards. This core has a 32-bit load/store architecture and is able to complete 32x16 multiplication in a single cycle. It equips with 16 32-bit general purpose registers and can complete a branch execution in two cycles. The chip has 4K CPU cache, 32K SDRAM and 384K eflash. The maximum clock speed is 100MHz.

For the purpose of high speed security applications such as ECC or RSA, the chip is incorporated with a mathematic module which is capable of computing 192 to 1024-bit integer modular multiplication at high speed. The module supports modular multiplication pre-computation. The pre-computation is done once for every different modular and the pre-computed value is loaded before each modular multiplication. This module shall be used to boost the underlining field multiplication in pairing.

Table 3 shows the time of main operations in a 256-bit prime field. The modular addition and subtraction are implemented in assembly code. From the table we can see the timing ratio between modular multiplication and addition or subtraction with random values is about 3.4.

Table 3. Timing of 256-bit Prime Field Operations

Operation	Condition	Timing	Timing Ratio
$A+B \mod P$	$A+B<P$	0.0024ms	1
$A+B \mod P$	$A+B \geq P$	0.0047ms	$\simeq 2$
$A-B \mod P$	$A \geq B$	0.0024ms	1
$A-B \mod P$	$A<B$	0.0047ms	$\simeq 2$
$A \times B \mod P$	pre-computation	0.012ms	5
$A \times B \mod P$	no pre-computation	0.024ms	10

5 Curve Parameters

For pairing-based cryptosystems, one should keep the right balance between the intended security level and system efficiency. Pairing may be used to convert DLP in $\mathbb{G}_1$ or $\mathbb{G}_2$ into the corresponding problem in $\mathbb{G}_3$. For pairings defined on elliptic curves, this has serious security implications. There exist algorithms for DLP in the finite fields running much faster than the best known general algorithms for the elliptic curve DLP. This requires $\mathbb{G}_3$ to be large enough to make sure the DLP in $\mathbb{G}_3$ has the same complexity as those in $\mathbb{G}_1$ and $\mathbb{G}_2$. On the other hand, pairing computation involves many operations in $\mathbb{G}_3$ and if $\mathbb{G}_3$ is too large, the computation would become very slow.

Barreto-Naehrig curves [13] defined over a 256-bit prime field is the de facto choice to implement pairing-based cryptosystems at the RSA-3072 security level. The standard Weierstrass representation of the curves is

$$Y^2 = X^3 + B.$$

The characteristic p of the prime field, the prime group order r of $E(\mathbb{F}_p)[r]$, the trace t of the Frobenius of the curves are parameterised by the variable u as follows:

$$\begin{aligned} p(u) &= 36u^4 + 36u^3 + 24u^2 + 6u + 1 \\ r(u) &= 36u^4 + 36u^3 + 18u^2 + 6u + 1 \\ t(u) &= 6u^2 + 1 \end{aligned}$$

The optimal Ate pairing on E is defined as [31]

$$a_{\text{opt}}(Q,P) = [f_{z,Q}(P) \cdot l_{[z]Q,\pi_p(Q)}(P) \cdot l_{[z]Q+\pi_p(Q),-\pi_p^2(Q)}(P)]^{(p^{12}-1)/r}$$

where $z = 6u+2$ and $l_{Q_1,Q_2}(P)$ is the equation of the line corresponding to the addition of Q_1 and Q_2 and its evaluation at P.

A Barreto-Naehrig curve admits a sextic twist $E'(\mathbb{F}_{p^2})$. The D-type sextic twist is defined as $Y^2 = X^3 + B/\xi$, where $\xi \in \mathbb{F}_{p^2}$ is an element that is neither a square nor a cube in $\mathbb{F}_{p^2}$ [16].

Regarding to the implemented cryptosystems, to defend Cheon's attack, we may choose r such that both $r-1$ and $r+1$ have small divisors as few as possible. At the same time to keep the Miller iterations few, $\mid z \mid$ should have hamming

Table 4. Curve Parameters

– **Barreto-Naehrig Curve 1:** p is 264-bit prime.

$$
\begin{aligned}
B &= 2\\
u &= 18000000002840543\\
p(u) &= \text{B640000004C6A1FDAB8C03C8A47B7DABE6C3DC8}\\
&\quad \text{D9E7C76F1D30469A1D3A7840E9B}\\
r(u) &= \text{B640000004C6A1FDAB8C03C8A47B7DABD943DC8}\\
&\quad \text{D9E4F2E931D0443A8D4D04DF165}\\
t(u) &= \text{D800000002D485EB60025F8FED7361D37}\\
r(u)-1 &= 2^2 \times 3 \times 5 \times \text{68D1EC4A0419(47-bit)}\times\\
&\quad \text{4CCCCCCCCD4D9AA7(63-bit)}\times\\
&\quad \text{18BA64EDD57110C5DBB8D755A70745054D0AB9(149-bit)}\\
r(u)+1 &= 2 \times 3 \times 73 \times \text{30E236F939FA7D(54-bit)}\times\\
&\quad \text{7BF6E71655DDCA91C1(71-bit)}\times\\
&\quad \text{1EDB6DB6DBD537B3E924E9148FA2C4D51(129-bit)}\\
|\ z\ | &= 9000000000F181F94\\
\text{HW}(|\ u\ |) &= 10\\
\text{NAF-length}(|\ u\ |) &= 10\\
\text{HW}(|\ z\ |) &= 16\\
\text{NAF-length}(|\ z\ |) &= 10
\end{aligned}
$$

– **Barreto-Naehrig Curve 2:** p is a 254-bit prime.

$$
\begin{aligned}
B &= 3\\
u &= 4000000001E11061\\
p(u) &= \text{24000000043A64DAD02FABEC83B0BDA0A0A60F7F}\\
&\quad \text{7E1723468B1BFE04FF448327}\\
r(u) &= \text{24000000043A64DAD02FABEC83B0BDA040A60F7F7}\\
&\quad \text{873F2238B06CE0DADEEE6A1}\\
t(u) &= 6000000005A3312300152FF751559C87\\
r(u)-1 &= 2^5 \times 3 \times 7 \times \text{924924924D70257 (60-bit)}\times\\
&\quad \text{18000000021D326D800FE3F982A38E58BACBA39D99}\\
&\quad \text{F77C7(185-bit)}\\
r(u)+1 &= 2 \times 5 \times 29 \times \text{13C1F0BB9E04B(49-bit)}\times\\
&\quad \text{894161E4B4F7A84E65(72-bit)}\times\\
&\quad \text{3000000002D19892400A97FBAE4DFF67(126-bit)}\\
|\ z\ | &= 1800000000B466248\\
\text{HW}(|\ u\ |) &= 10\\
\text{NAF-length}(|\ u\ |) &= 8\\
\text{HW}(|\ z\ |) &= 13\\
\text{NAF-length}(|\ z\ |) &= 13
\end{aligned}
$$

weight (HW) as low as possible to keep its non-adjacent form (NAF) short. This in turn requires $|\ u\ |$ to have low hamming weight. Shorter NAF of $|\ u\ |$ also benefits the final exponentiation in pairing computation.

For the strict RSA-3072 security level, one may choose slightly larger field to compensate the complexity deduction due to the attack. On the other hand, to

benefit the computation with the lazy reduction technique [3], a 254-bit prime p was proposed to implement pairings. We also give a 254-bit prime curve example, although our implementation on AC4384 does not use the technique.

Table 4 gives two curve parameter examples. One is defined over a 264-bit field for better security, the other is defined over a 254-bit field for better performance. A few more curve parameter sets with $\mid u \mid$ having lower hamming weights are presented in Appendix.

According to [24], while the special form of p can be used to construct a more efficient number field sieve, the complexity of DLP in $\mathbb{G}_3$ reduced to $L_{p^{12}}(1/3, (\frac{80}{9})^{1/3})$ is still greater than the one of the integer factorization problem. The parameters of Curve 1 provides security at the RSA-3072 security level against Cheon's attack if an attacker collects less than 2^{47} private keys. For the parameters of Curve 2, if an attacker collects less than 2^{49} private keys, the cryptosystems may still stand above the AES-122 bit security level against Cheon's attack.

6 Implementation Issues

Since Barreto et al.'s major improvement [10] to the Miller algorithm, during the years, steady improvements have been made, one over the other. Now pairings on Barreto-Naehrig curves at the RSA-3072 security level may be computed on general CPUs within a millisecond [3].

Our implementation makes use of those techniques suitable for the chosen architecture with a few exceptions. First, as the chip has a hardware module for modular multiplications (but no inversion), we use the homogeneous projective coordinate and do not consider the lazy reduction optimization [3]. Second, on the chip the timing of modular square (SQR) and modular multiplication (MUL) is the same and the ratio between MUL and addition (ADD) or subtraction (SUB) (3.4 as shown in Table 3) is not high. Hence, those tricks intended to make use the timing difference between SQR and MUL should be avoided if they increase ADD or SUB operations. Third, when the efficiency of the implemented cryptosystems is above the acceptable level, reducing the code size and storage requirement become the first priority. Those speed optimizations requiring extra complicated functions or extra memory is excluded.

Following lists the optimization techniques to the Miller algorithm adopted in the implementation:

- Use the BKLS algorithm [10] which removes denominators from the Miller algorithm.
- Represent the first input of a_{opt} on the sextic twist of the Barreto-Naehrig curve, i.e, Q now is in $E'(\mathbb{F}_{p^2})[r]$ [16]. Q is untwisted back to $E(\mathbb{F}_{p^{12}})$ during the line evaluation $l_{\cdot,\cdot}(\cdot)$ in Algorithm 1.
- Represent $\mathbb{F}_{p^{12}}$ as a tower of finite extensions as suggested in [16]:

$$\begin{aligned}\mathbb{F}_{p^2} &= \mathbb{F}_p[X]/(X^2-\beta), \text{where } \beta \text{ is a non-quadratic in } \mathbb{F}_p\\ \mathbb{F}_{p^6} &= \mathbb{F}_{p^2}[Y]/(Y^3-\xi), \text{where } \xi \text{ is neither a square nor a cube in } \mathbb{F}_{p^2}\\ \mathbb{F}_{p^{12}} &= \mathbb{F}_{p^6}[Z]/(Z^2-\mu), \text{where } \mu \text{ is a non-quadratic in } \mathbb{F}_{p^6}\end{aligned}$$

For the two curves listed in Section 5, β, ξ and μ's value is listed in Table 5.

Table 5. Parameters for Extension Field Representation

	β	ξ	μ
Curve 1	-2	$-1-\sqrt{\beta}$	$\sqrt[3]{\xi}$
Curve 2	-1	$1+\sqrt{\beta}$	$\sqrt[3]{\xi}$

- Use the efficient formulas proposed in [4] to evaluate $l_{.,.}(\cdot)$ in Algorithm 1, particularly pre-computed $-P$ is used.
- Use the fact that the value of $l_{.,.}(\cdot)$ in Algorithm 1 is sparse in $\mathbb{F}_{p^{12}}$ to improve the efficiency of field multiplication in line 4, 6, 8, 13 and 15 [3].
- Use the method in [27] to compute the final exponentiation in line 16 in Algorithm 1. The implementation does not adopt the slightly faster method in [17] for compatibility reason. Exponentiation to power of p are computed using Frobenius function with only one pre-computation [16].
- Use NAF to represent both $\mid z \mid$ in Algorithm 1 and $\mid u \mid$ in the final exponentiation computation.
- Replace any inversion in $\mathbb{F}_{p^{12}}$ with conjugation *conj* if possible as suggested in [3].

Algorithm 1: BKLS Algorithm for Optimal Ate Pairing on B-N Curve

Input: $P \in \mathbb{G}_1, Q \in \mathbb{G}_2', u, \mid z \mid = \mid 6u+2 \mid = (1, z_{s-1}, \ldots, z_0)_{\text{NAF}}$
Output: $a_{\text{opt}}(Q, P)$

1 $Z \leftarrow Q, f \leftarrow 1$;
2 **for** $i \leftarrow s_1$; $i \geq 0$; $i--$ **do**
3 $\quad f \leftarrow f^2$;
4 $\quad f \leftarrow f \cdot l_{Z,Z}(-P), Z \leftarrow [2]Z$;
5 $\quad$ **if** $z_i = 1$ **then**
6 $\quad\quad f \leftarrow f \cdot l_{Z,Q}(P), Z \leftarrow Z + Q$;
7 $\quad$ **if** $z_i = -1$ **then**
8 $\quad\quad f \leftarrow f \cdot l_{Z,-Q}(P), Z \leftarrow Z - Q$;
9 **if** $u < 0$ **then**
10 $\quad f \leftarrow conj(f)$;
11 $\quad Z = -Z$;
12 $T \leftarrow \pi_p(Q)$;
13 $f \leftarrow f \cdot l_{Z,T}(P), Z \leftarrow Z + T$;
14 $T \leftarrow -\pi_p(T)$;
15 $f \leftarrow f \cdot l_{Z,T}(P), Z \leftarrow Z + T$;
16 $f \leftarrow f^{(p^{12}-1)/r}$;
17 **return** f;

The performance of the optimal Ate pairing and point scalar is measured on two platforms: the AC4384 microcontroller and the i7-4650U 1.70GHz (boosted to 2.3GHz). On the i7 machine, prime field elements are in Montgomery representation and underlying integer operations are completed with the GMP library v5.1.2. The whole library is compiled using GCC v4.4.7 with option $-\mathbf{O2}$. On AC4384 microcontroller the code is compiled in favor of small code size. Hence here instead of going after a speed record, we merely show that implementation of IBC on such type of chips may achieve reasonably good performance.

The pairing is computed with at most 131 variables in $\mathbb{F}_p$, each is stored in a 36-byte array. The total 4716 bytes are reserved in advance from a global memory shared with other routines implemented in the USB token.

SK-KEM uses identity private keys in $\mathbb{G}_2$ to reduce ciphertext size and improves encryption/decryption speed. BLMQ-IBS may use identity private keys in $\mathbb{G}_1$ for smaller signature size and faster signing operation. If a user has only one identity private key used in both schemes, it is better to map the private key in $\mathbb{G}_2$ because the signing operation in BLMQ-IBS has no heavy pairing operation and the point scalar in $\mathbb{G}_2$ can be completed fairly fast. In this case, the point scalar in $\mathbb{G}_2$ may be computed with the same $l_{\cdot,\cdot}(\cdot)$ in Algorithm 1 but without evaluating on point P to reduce the code size.

Table 6 lists the timing of optimal Ate pairing and point scalar in $\mathbb{G}_2$ over Curve 1 and 2. Table 7 lists the timing of SK-KEM decryption and BLMQ-IBS signing operation over Curve 1 and 2 with identity private keys generated in $\mathbb{G}_2$.

Table 6. Timing of Group Operations

	a_{opt}^{254} Pairing	a_{opt}^{264} Pairing	Point Scalar in $\mathbb{G}_2^{254}$	Point Scalar in $\mathbb{G}_2^{264}$
AC4384	0.49s	0.55s	0.24s	0.27s
i7-4650U	1.65ms	1.86ms	0.993ms	1.16ms

Table 7. Timing of Decrypt/Sign Operations

	SK-KEM$_{\mathbb{G}_2}^{254}$ Dec	SK-KEM$_{\mathbb{G}_2}^{264}$ Dec	BLMQ-IBS$_{\mathbb{G}_2}^{254}$ Sign	BLMQ-IBS$_{\mathbb{G}_2}^{264}$ Sign
AC4384	0.57s	0.69s	0.41s	0.49s
i7-4650U	1.73ms	1.92ms	1.65ms	2.04ms

7 Conclusion

Pairing-based cryptosystems have been intensively studied for more than ten years. Several identity-based schemes from pairings have already been standardized and widely deployed. Many optimization techniques have been developed to improve such systems' efficiency. In this paper, by carefully choosing curve

parameters and adopting many those state-of-art improvements, we have successfully implemented two important identity-based cryptosystems on a 32-bit microcontroller. The final product as a USB token shows identity-based cryptography is able to perform decently well at a high security level in a resource constrained device.

Acknowledge

We'd like to thank Mike Scott for his many comments and valuable suggestions.

References

1. G. Adj, A. Menezes, T. Oliveira and F. Rodríguez-Henríquez. Weakness of $F_{3^{6*509}}$ for discrete logarithm cryptography. Cryptology ePrint Archive: Report 2013/446.
2. http://www.aisinochip.com/product/ac_723.html
3. D. F. Aranha, K. Karabina, P. Longa, C. Gebotys and J. López. Faster explicit formulas for computing pairings over ordinary curves. In *Proc. of Advances in Cryptology - Eurocrypt 2011*, LNCS 6632, pp. 48–68, 2001.
4. D. F. Aranha, P. Barreto, P. Longa, and J. Ricardini. The realm of the pairings. Cryptology ePrint Archive: Report 2013/722.
5. D. Boneh and X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In *Proc. of Advances in Cryptology - Eurocrypt 2004*, LNCS 3027, pp. 223–238, 2004.
6. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. In *Proc. of Advances in Cryptology - Crypto 2001*, LNCS 2139, pp. 213–229, 2001.
7. M. Barbosa, L. Chen, Z. Cheng, M. Chimley, A. Dent, P. Farshim, K. Harrison, J. Malone-Lee, N.P. Smart and F. Vercauteren. SK-KEM : an identity-based KEM. Sumission to IEEE P1363.3, 2006.
8. G. M. Bertoni, L. Chen, P . Fragneto, K. A. Harrison, and G. Pelosi. A pairing SW implementation for Smart-Cards. In *Journal of Systems and Software*, Volume 81 Issue 7, pp. 1240–1247, 2008.
9. P. Barreto, S. Galbraith, C. Ó'hÉigeartaigh and M. Scott. Efficient pairing computation on supersingular abelian varieties. *Des Codes Crypt* (2007) 42:239–271.
10. P. Barreto, H. Kim, B. Lynn and M. Scott. Efficient implementation of pairing-based cryptosystems. In *Journal of Cryptology*, 17(4): 321–34, 2004.
11. D. Boneh, B. Lynn and H. Shacham. Short signatures from the weil pairing. In *Journal of Cryptology* 17: 297-319, 2004.
12. P. Barreto, B. Libert, N. McCullagh and J.-J. Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. In *Proc. of Advances in Cryptology - Asiacrypt 2005*, LNCS 3788, pp. 515–532, 2005.
13. P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In *Proc. of Selected Areas in Cryptography 2005*, LNCS 3897, pp. 319-331, 2006.
14. J. H. Cheon. Security analysis of the strong Diffie-Hellman problem. In *Proc. of Advances in Cryptology - Eurocrypt 2006*, LNCS 4004, pp. 1–11, 2006.
15. J. C. Cha and J. H. Cheon. An identity-based signature from gap Diffie-Hellman groups. In *Proc. of Public Key Cryptography - PKC 2003*, LNCS 2567, pp. 18–30, 2003.

16. A. J. Devegili, M. Scott, R. Dahab. Implementing cryptographic pairings over Barreto-Naehrig curves. Cryptology ePrint Archive: Report 2007/390.
17. L. Fuentes-Castañeda, E. Knapp and F. Rodríguez-Henríquez. Faster hashing to G2. In *Proc. of Selected Areas in Cryptography 2011*, pp. 412–430, 2011
18. V. Goyal, O. Pandey, A. Sahai and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proc. of ACM on Computer and Communications Security*, pp. 89–98, 2006.
19. F. Hess, N.P. Smart and F. Vercauteren. The Eta pairing revisited. In *IEEE Transactions on Information Theory*, Volume: 52, Issue: 10, pp. 4595 – 4602, 2006.
20. V.S. Miller. The Weil pairing, and its efficient calculation. In *Journal of Cryptology*, 17(4):235–261, 2004
21. IEEE P1363. Identity-based public-key cryptography using pairings. 2009.
22. A. Joux. A one-round protocol for tripartite Diffie-Hellman. In *Proc. of Algorithm Number Theory Symposium – ANTS-IV*, LNCS 1838, pp. 385–394, 2000.
23. M.Joye and G. Neven. Identity-based cryptography. IOS press 2009.
24. A. Joux and C. Pierrot. The special number field sieve in F_{p^n}, application to pairing-friendly constructions. Cryptology ePrint Archive: Report 2013/582.
25. E. Lee, H. Lee and C. Park, Efficient and generalized pairing computation on abelian varieties. In *IEEE Transactions on Information Theory*, Volume 55, pp. 1793–1803, 2009.
26. C. Porto Lopes Gouvêa and J. López. Software implementation of pairing-based cryptography on sensor networks using the MSP430 microcontroller. In *Proc. of Progress in Cryptology - Indocrypt 2009*, LNCS 5922, pp. 248–262, 2009.
27. M. Scott, N. Benger, M. Charlemagne, L. J. Dominguez Perez and E. J. Kachisa. On the final exponentiation for calculating pairings on ordinary elliptic curves. In *Proc. of Pairing-Based Cryptography - Pairing 2009*, LNCS 5671, pp. 78–88, 2009.
28. M. Scott, N. Costigan, and W. Abdulwahab. Implementing cryptographic pairings on smartcards. In *Proc. of Cryptographic Hardware and Embedded Systems - CHES 2006*, LNCS 4249, pp. 134–147, 2006.
29. R. Sakai and M. Kasahara. ID based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive, Report 2003/054, 2003.
30. R. Sakai, K. Ohgishi and M. Kasahara. Cryptosystems based on pairing. In *Proc. of 2000 Symposium on Cryptography and Information Security*, Japan, 2000.
31. F. Vercauteren. Optimal pairings. In *IEEE Transactions on Information Theory*, Volume: 56, Issue: 11, pp. 455 – 461, 2010.
32. F. Zhang, R. Safavi-Naini and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In *Proc. of Public Key Cryptography - PKC 2004*, LNCS 2947, pp. 277–290, 2004.

Appendix

Here we give a couple of extra curve parameter sets of which $\mid u \mid$ has slightly low hamming weights and $r(u) - 1$ or $r(u) + 1$ has divisors of around 40 bits. The implementation of optimal Ate pairing on these curve parameters on the i7 CPU is slower than on the ones given in Section 5. If one is willing to accept parameters with $r(u) - 1$ or $r(u) + 1$ having divisors of around 30 bits, she may use those parameters resulting in better system efficiency such as those with $u = -$ 40000FEFF7FFFFF6.

– **Barreto-Naehrig Curve 3:** p is 264-bit prime.

$$
\begin{aligned}
B &= 5 \\
u &= \text{-18000200000013804} \\
p(u) &= \text{B6403CC0079A50C3B29813F9F84313D3F36D74ECB9C9} \\
&\quad \text{F13987248B1DB77E88CC69} \\
r(u) &= \text{B6403CC0079A50C3B29813F9F84313D3E5ED72ACB9B1} \\
&\quad \text{DB493D50851DAE94CE4C09} \\
t(u) &= \text{D800240001815F049D4060008E9BA8061} \\
r(u)-1 &= 2^3 \times 3^2 \times \text{200002AAAAAAC4AB(62-bit)} \times \\
&\quad \text{14400510006C315F9CBA1917E923B4A9109C293CF609A} \\
&\quad \text{8292B(197-bit)} \\
r(u)+1 &= 2 \times 5 \times \text{297B8FF5B95(42-bit)} \times \\
&\quad \text{29A7F38A869F523593FA651(90-bit)} \times \\
&\quad \text{2B333A6666B3796752A67999B61F219AD(130-bit)} \\
|\, z \,| &= \text{90000C00000075016} \\
\text{HW}(|\, u \,|) &= 8 \\
\text{NAF-length}(|\, u \,|) &= 7 \\
\text{HW}(|\, z \,|) &= 12 \\
\text{NAF-length}(|\, z \,|) &= 11 \\
\beta &= -5 \\
\xi &= \sqrt{\beta} \\
\mu &= \sqrt[3]{\xi}
\end{aligned}
$$

– **Barreto-Naehrig Curve 4:** p is a 254-bit prime.

$$
\begin{aligned}
B &= 5 \\
u &= \text{-400010000000F108} \\
p(u) &= \text{240024000D82209306E179BB4521C1FD7FCDF4058082A} \\
&\quad \text{F1C0194453316D2D7D1} \\
r(u) &= \text{240024000D82209306E179BB4521C1FD1FCDC4057A7F} \\
&\quad \text{DC034CCE452DC5327651} \\
t(u) &= \text{600030000602D318B4C6000551A06181} \\
r(u)-1 &= 2^4 \times 3 \times 7 \times \text{19864F3D291 (41-bit)} \times \\
&\quad \text{124929249249697(57-bit)} \times \text{B35CE76183E5F192B(68-bit)} \times \\
&\quad \text{1578F9956E01AB6113375(81-bit)} \\
r(u)+1 &= 2 \times 11 \times 13 \times \text{17A8B7604589170D(61-bit)} \times \\
&\quad \text{27F3D6A82B6CE201(62-bit)} \times \\
&\quad \text{8BA32E8BABA7048106F17464D3D2023(124-bit)} \\
|\, z \,| &= \text{1800060000005A62E} \\
\text{HW}(|\, u \,|) &= 8 \\
\text{NAF-length}(|\, u \,|) &= 6 \\
\text{HW}(|\, z \,|) &= 14 \\
\text{NAF-length}(|\, z \,|) &= 13 \\
\beta &= -5 \\
\xi &= \sqrt{\beta} \\
\mu &= \sqrt[3]{\xi}
\end{aligned}
$$

– **Barreto-Naehrig Curve 5:** p is a 254-bit prime.

$$
\begin{aligned}
B &= 2 \\
u &= \text{-40000FEFF7FFFFF6} \\
p(u) &= 240023\text{DBFB65022FB6D6B42EF2DF822F34ECBEEA7A2} \\
&\quad 7\text{C120863C9C54B006147D} \\
r(u) &= 240023\text{DBFB65022FB6D6B42EF2DF822ED4EC8F1A8C3} \\
&\quad 3\text{C13884C414D0F0061225} \\
t(u) &= 60002\text{FCFEDF3FFE801788783C0000259} \\
r(u)-1 &= 2^2 \times 3 \times 200007\text{F7FBFFFFFB (62-bit)} \times \\
&\quad 180011\text{EDFB770057C1AAF589E768012691F7B793E7FFE} \\
&\quad 619\text{(189-bit)} \\
r(u)+1 &= 2 \times 193 \times 54\text{BC6217D(35-bit)} \times 290\text{DB508C7F(42-bit)} \times \\
&\quad 4\text{AF782A8D59(43-bit)} \times 417\text{BD1FB7B63B(51-bit)} \times \\
&\quad 1774\text{D71493CDEA971C7B(77-bit)} \\
|\, z \,| &= 180005\text{F9FCFFFFFC2} \\
\text{HW}(|\, u \,|) &= 41 \\
\text{NAF-length}(|\, u \,|) &= 6 \\
\text{HW}(|\, z \,|) &= 39 \\
\text{NAF-length}(|\, z \,|) &= 10 \\
\beta &= -2 \\
\xi &= \sqrt{\beta} \\
\mu &= \sqrt[3]{\xi}
\end{aligned}
$$