Software implementation of pairings at the 128-bit security level

Darrel Hankerson, Auburn University
(with D. Aranha, S. Chatterjee, J. López, A. Menezes)

ECC, October 2010

Overview

Optimization and protocol issues for pairings over supersingular curves, in particular for the genus-2 case.

1. Structure of pairings over (hyper)elliptic curves.
2. Eta pairing.
3. Hardware characteristic 2 multiplier. Parallelization for a single paring.
4. Genus-2 supersingular curve.
5. BLS signature scheme with the genus-2 curve.

Pairings from Elliptic Curves

Let E be an elliptic curve defined over $\mathbb{F}_q$.

- Let $n \approx q$ be a prime divisor of $\#E(\mathbb{F}_q)$ with $\gcd(n, q) = 1$.
- Let k be the smallest positive integer with $n \mid q^k - 1$, and suppose that $k > 1$. Then $E[n] \subseteq E(\mathbb{F}_{q^k})$.
- Let $\mathbb{G}_T$ be the order-n subgroup of $\mathbb{F}_{q^k}^*$.

The (reduced/restricted) Tate pairing is

$$t : E(\mathbb{F}_q)[n] \times E[n] \to \mathbb{G}_T$$

defined by

$$t(P, Q) = f_{n,P}(Q)^{(q^k-1)/n}$$

where $f_{n,P}$ is a *Miller function* with divisor $n(P) - n(\infty)$.

Miller's Algorithm for Computing $t(P, Q)$

Let $n = \sum_{i=0}^{d} n_i 2^i$.

1. Set $f \leftarrow 1$, $R \leftarrow P$.
2. For i from d down to 0 do:
   - 2.1 Let ℓ be the tangent line through R, and let v be the vertical line through $2R$.
   - 2.2 $R \leftarrow 2R$.
   - 2.3 $f \leftarrow f^2 \cdot \ell(Q)/v(Q)$.
   - 2.4 If $n_i = 1$ then
     - 2.4.1 Let ℓ be the line through R and P and let v be the vertical line through $R + P$.
     - 2.4.2 $R \leftarrow R + P$.
     - 2.4.3 $f \leftarrow f \cdot \ell(Q)/v(Q)$.
3. Return $f^{(q^k-1)/n}$.

Optimizations

1. Improve the arithmetic in the main loop. Parallelize.
2. Reduce the number of iterations.
3. Improve the arithmetic in the final exponentiation.

Symmetric Pairings

Let $\mathbb{G}_1 = \mathbb{G}_2 = E(\mathbb{F}_q)[n]$.

- Let Φ be an endomorphism on E with $\Phi(\mathbb{G}_1) \neq \mathbb{G}_1$.
- $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ defined by

$$e(P, Q) = t(P, \Phi(Q))$$

is a symmetric (Type 1) pairing.

Most pairing-based protocols can be implemented with symmetric pairings. For the 128-bit security level:

k	Curve	Bitlength of $\mathbb{F}_{q^k}$
2	$Y^2 = X^3 + aX/\mathbb{F}_{p1536}$	3072
4	$Y^2 + Y = X^3 + X/\mathbb{F}_{2^{1223}}$	4892
6	$Y^2 = X^3 - X + 1/\mathbb{F}_{3^{509}}$	4840

Eta Pairing ($k = 4$)

[Barreto, Galbraith, Ó' hÉigeartaigh, Scott]

$$E/\mathbb{F}_2 : Y^2 + Y = X^3 + X$$

$$N = \#E(\mathbb{F}_{2^m}) = 2^m - 2^{(m+1)/2} + 1, \quad m \equiv 3 \pmod 8.$$

- Doubling is cheap: $[2](x, y) = (x^2, y^2 + 1)$.
- Distortion map:

$$\mathbb{F}_{2^{2m}} = \mathbb{F}_{2^m}[u]/(u^2 + u + 1), \quad \mathbb{F}_{2^{4m}} = \mathbb{F}_{2^{2m}}[v]/(v^2 + v + u)$$

$$\Phi(x, y) = (x + u^2, y + ux + v).$$

- Pairing: Let $P, Q \in E(\mathbb{F}_{2^m})$. Then

$$\eta(P, Q) = f_{T,P}(\Phi(Q))^M$$

where

$$T = 2^{(m+1)/2}$$

$$M = (2^{4m} - 1)/N = (2^m + 2^{(m+1)/2} + 1)(2^{2m} - 1)$$

- $\eta(P, Q)$ is a fixed power of the Tate pairing.

Computing $\eta(P, Q)$

Input: $P = (x_1, y_1), Q = (x_2, y_2) \in E(\mathbb{F}_{2^m})$.

1. $z \leftarrow x_1 + 1$.
2. $f \leftarrow z \cdot (x_1 + x_2 + 1) + y_1 + y_2 + (z + x_2)u + v$.
3. For i from 1 to $(m + 1)/2$ do:
   - 3.1 $z \leftarrow x_1$, $x_1 \leftarrow \sqrt{x_1}$, $y_1 \leftarrow \sqrt{y_1}$.
   - 3.2 $g \leftarrow z \cdot (x_1 + x_2) + y_1 + y_2 + x_1 + 1 + (z + x_2)u + v$.
   - 3.3 $f \leftarrow f \cdot g$.
   - 3.4 $x_2 \leftarrow x_2^2$, $y_2 \leftarrow y_2^2$.
4. Return $f^{(2^{2m}-1)(2^m-2^{(m+1)/2}+1)}$.

Cost estimate: $7 \cdot (m + 1)/2$ multiplications in $\mathbb{F}_{2^m}$.

Genus-2 Supersingular Curve

[Barreto, Galbraith, Ó' hÉigeartaigh, Scott]

$$C/\mathbb{F}_2 : Y^2 + Y = X^5 + X^3 + b.$$

- Degree zero divisor class group $J_C(\mathbb{F}_q)$, $q = 2^m$.
- Reduced divisors:
  Degenerate: $(P) - (\infty)$, $P \in C(\mathbb{F}_q)$.
  Non-degenerate: $(P_1) + (P_2) - 2(\infty)$
  Type A: $P_1, P_2 \in C(\mathbb{F}_q) \setminus \{\infty\}$.
  Type B: $P_1 \in C(\mathbb{F}_{q^2}) \setminus C(\mathbb{F}_q)$, $P_2 = \pi(P_1)$.
- Mumford rep: $a, b \in \mathbb{F}_{2^m}[z]$, $\deg(b) < deg(a) \leq 2$.
- $\#J_C(\mathbb{F}_q) \approx q^2$.
- Embedding degree is $k = 12$ ($\#J_C(\mathbb{F}_{q^2}) \mid q^{12} - 1$).

Symmetric Pairings

For the 128-bit security level:

k	Curve	Bitlength of $\mathbb{F}_{q^k}$
2	$Y^2 = X^3 + aX/\mathbb{F}_{p1536}$	3072
4	$Y^2 + Y = X^3 + X/\mathbb{F}_{2^{1223}}$	4892
6	$Y^2 = X^3 - X + 1/\mathbb{F}_{3^{509}}$	4840
12	$Y^2 + Y = X^5 + X^3/\mathbb{F}_{2^{439}}$	5268

$k = 12$ gives relatively small base field.

Eta Pairing on Degenerate Divisors

- Octupling is cheap: If $P = (x, y) \in C(\mathbb{F}_q)$, then
$$8((P) - (\infty)) = (P') - (\infty)$$
where $P' = (x^{64} + 1, y^{64} + x^{128} + 1)$.
- Eta pairing: $D_i = (P_i) - (\infty)$. $\eta(D_1, D_2)$ is a fixed power of the Tate pairing.
- Cost estimate: $69 \cdot (m - 1)/2$ multiplications in $\mathbb{F}_{2^m}$.

Eta Pairing on General Divisors

1. If D_1 is degenerate and $D_2 = (P_1) + (P_2) - 2(\infty)$ is Type A non-degenerate, then
$$\eta(D_1, D_2) = \eta(D_1, (P_1) - (\infty)) \cdot \eta(D_1, (P_2) - (\infty)).$$
2. If D_1 is degenerate (and fixed) and D_2 is Type B non-degenerate, then find a (small) integer c such that $D_2' = D_2 + cD_1$ is Type A. Then
$$\eta(D_1, D_2) = \eta(D_1, D_2')/\eta(D_1, D_1)^c.$$
(1 and 2 not necessarily fastest [Aranha, Beuchat, Detrey, Estibals], but can use common code.)
3. For the general case, Lee & Lee give an alg for $\eta(D_1, D_2)$ using resultant. Cost estimate from mult counts gives factor 4 over degenerate-degenerate case.

Timings

[Barreto, Galbraith, Ó' hÉigeartaigh, Scott, 2007]

Timings (in milliseconds) for the eta pairing at the "1230-bit security level" on a 3 GHz Intel Pentium 4:

Curve	Pairing
$E(\mathbb{F}_{2^{307}})$	3.50
$E(\mathbb{F}_{3^{127}})$	5.36
$C(\mathbb{F}_{2^{103}})$ degenerate	1.87
$C(\mathbb{F}_{2^{103}})$ non-degenerate	6.42

Multiplication in $\mathbb{F}_{2^{103}}$ exploited 128-bit SIMD registers. Other fields used only 32-bit registers.

Timings at 128-bit security level

Timings (in clock cycles) for the eta pairing at the 128-bit security level on an Intel Core2.

Curve	Number of field mults	Pairing (10^6 cycles)
$E/\mathbb{F}_{2^{1223}}$	4,284	19.0
$E/\mathbb{F}_{3^{509}}$	3,570	15.8
$C/\mathbb{F}_{2^{439}}$ (degenerate divisors)	15,111	16.4
$E/\mathbb{F}_{p_{256}}$ (BN[a])	15,093	10
$E/\mathbb{F}_{p_{256}}$ (BN[b])		4.5
$E/\mathbb{F}_{p_{256}}$ (BN[c])	12,785	3.3

[a] R-ate via MIRACL, 2008
[b] Naehrig, Niederhagen, Schwabe
[c] Beuchat, Diaz, Mitsunari, Okamoto, Rodríguez-Henríquez, Teruya

Beneath the timings

Times suggested that genus-2 with the anticipated hardware char 2 multiplier would be competitive with BN[a].

- Naehrig, Niederhagen, Schwabe (BN[b]) used an elegant (redundant) field rep with floating-point arithmetic.
  - SIMD can do 2 floating-point mult simultaneously.
  - ...but operand size is limited by 53-bit mantissa while integer multiplier is relatively fast on 64-bit operands.
  - Bernstein: floating-point on Pentium for point mult on NIST curves. 80-bit regs rather than SIMD. Integer mult is 32-bit.

  So Naehrig et al. seemed surprisingly fast.
- Beuchat et al. (BN[c]): faster times with alg improvements and faster mult with integer multiplier.
  Overhead BN[a] was more than suspected.

Approaches may find application across hardware.

Developments favoring small characteristic

Hardware char 2 multiplier (Intel Core i5, 2009) Pre-release speculation: should give factor > 2 accel in field mult against methods relying on lookup with a few bits.

- Gueron and Kounavis [2008], estimates for point mult on NIST random curve over $\mathbb{F}_{2^{233}}$ (B-233):

Method	acceleration
OpenSSL	0.57X
OpenSSL with enhancements	1X
...and 9-clock HW multiplier	12X
...and 3-clock HW multiplier	37X

- Aranha, Rodríguez-Henríquez [2010] with the real thing:

Accel for NIST random curve over $\mathbb{F}_{2^{233}}$	
Field multiplication	2.1X
Point multiplication	1.7X

Hardware char 2 multiplier (2/2)

Why isn't actual $\approx$ predicted?

1. OpenSSL in the 2008 estimates not written for speed records. Need comparisons against fast versions using 64- or 128-bit registers.
2. L-D "comb" commonly used for field mult is quite good in the 128-bit registers.

   Sanity test: if mult in $\mathbb{F}_{2^{233}}$ is charged as 16 polynomial mult of 64-bit operands, then comb with 128-bit registers is *15 cycles* per such op.

   Can't expect the acceleration factors from [GK, 2008].
3. [Fog] HW multiplier: throughput 1/8, latency 12. (Appears perfect scheduling can do better.) Charged as in item 2, $\mathbb{F}_{2^{233}}$ mul is *7 cycles* each.

   (Karatsuba appears effective even at this field size, and the experiments have 9 HW muls. Regardless, L-D times mean HW won't reach GK estimates.)

Parallelization

Pairing finds Miller function $f_{r,P}$. Strategy to apply multiple cores:

1. Write $r = 2^w r_1 + r_0$ for some w.
2. Let $\ell_{P,Q}$ be line through P and Q and v_P be vertical line through P. Then
$$f_{r,P} = f_{2^w r_1 + r_0, P} = f_{2^w r_1, P} \cdot f_{r_0,P} \cdot \frac{\ell_{2^w r_1 P, r_0 P}}{v_{rP}}.$$
3. Can evaluate $f_{2^w r_1, P}$ as
$$f_{r_1,P}^{2^w} \cdot f_{2^w, r_1 P} \quad \text{or} \quad f_{2^w,P}^{r_1} \cdot f_{r_1, 2^w P}$$
depending on curve, embedding degree, weight of r,...
4. For our case, r_0 is small and $f_{r,P}$ is approx two half-length Miller function calculations.
5. Can apply recursively to exploit more processors.

Parallelization of the η_T pairing

INPUT: $P = (x_P, y_P), Q = (x_Q, y_Q) \in E(\mathbb{F}_{2^m}[r])$, starting point w_i for processor i.
OUTPUT: $\eta_T(P,Q) \in \mathbb{F}^*_{2^{4m}}$.

parallel section(processor i)
Initialize F_i
$x_{Q_i} \leftarrow (x_Q)^{2^{w_i}}$, $y_{Q_i} \leftarrow (y_Q)^{2^{w_i}}$
$x_{P_i} \leftarrow (x_P)^{\frac{1}{2^{w_i}}}$, $y_{P_i} \leftarrow (y_P)^{\frac{1}{2^{w_i}}}$
for $j \leftarrow w_i$ **to** $w_{i+1} - 1$ **do**
  $x_{P_i} \leftarrow \sqrt{x_{P_i}}, y_{P_i} \leftarrow \sqrt{y_{P_i}}, x_{Q_i} \leftarrow x_{Q_i}^2, y_{Q_i} \leftarrow y_{Q_i}^2$
  $u_i \leftarrow x_{P_i} + \alpha, v_i \leftarrow x_{Q_i} + \alpha$
  $g_{0_i} \leftarrow u_i \cdot v_i + y_{P_i} + y_{Q_i} + \beta$, $g_{1_i} \leftarrow u_i + x_{Q_i}$
  $F_i \leftarrow F_i \cdot (g_{0_i} + g_{1_i} s + t)$
end for
$F \leftarrow \prod_{i=0}^{\pi} F_i$
end parallel
return $F^{(2^{2m}-1)(2^m+1\pm 2^{(m+1)/2})}$.

Acceleration for supersingular curve

Pairing with EC over $\mathbb{F}_{2^{1223}}$ (128-bit security level), estimated and experimental on Core 2 (45nm) and Core i5.

	Number of processors			
	1	2	4	8
Estimated accel factor		1.9	3.5	5.8
Core2 time (10^6 cycles)	17.4	9.3	5.1	3.0
Acceleration factor		1.9	3.4	5.8
Core i5 time (10^6 cycles)	7.5	4.3	2.5*	1.7*
Acceleration factor		1.7	3.0	4.5

*Estimate from per-thread data.

- Experimental is close to estimated.
- Thread synchronization (via OpenMP) cost small.
- Parallelization overhead increases with number of processors.

Parallelization for asymmetric pairings

Technique is not specific to symmetric pairings. But:

- r_0 not so small, exponentiation by 2^w not negligible.
- Final exponentiation is a larger portion of pairing cost.

Grabher, Großschädl, Page [2008] obtain factor 1.6 accel for 2 cores on a BN curve.

- OpenMP used to parallelize $\mathbb{F}_{p^{12}}$ arithmetic and for simultaneous $\mathbb{F}_{p^2}$ ops.
- ...but single-thread times are $>$ 4X slower than BN^b, BN^c.
- A little help: Aranha, Karabina, Longa, Gebotys, López reduce cost of squarings in final exponentiation.

More opportunities if parallelization could be applied lower.

- OpenMP can have 3000-cycle sync on basic use.
- 1000 cycles with POSIX threads and spinlocks.
- ...but $\mathbb{F}_p$ multiplication is in hundreds of cycles.

Who wins the speed record?

If the question is "what's the fastest single pairing" then supersingular curves over char 2 fields appear to use multiple cores more efficiently and data suggests competitive with BN curves if given enough cores and hardware multiplier.

But...is multi-core parallelism applied to a single pairing useful?

- Probably not where these processors are targeted.
- Application: weak device with multi-thread capability.

What about genus-2?

Embedding degree 12 can mean parameter-size advantages. Times for particular implementation here are not compelling.

- Aranha, Beuchat, Detrey, Estibals cut Miller loop by 1/3.
- Pairing at security level corresponding to field size of 367 (rather than 439) bits on Core 2 and Core i5 (using hardware multiplier):

Pairing	Core 2	Core i5
Degenerate	5.0	2.5
Mixed	9.3	4.5
General	18.4	8.6

Units: 10^6 cycles

- The competition: Aranha, Rodríguez-Henríquez report BN times of 1.7–2.3 on this platform (128-bit security level).

Pairings for genus-2, especially on degenerate divisors, interesting again.

Arranging for degenerate divisors

BGOS remarked that parameters can be chosen in BF-IBE so that encryption is on degenerate divisors.

- Degenerate is important for speed.
- But...the security argument in the EC setting does not carry [CHM, 2010].

We illustrate security argument for Boneh-Lynn-Shacham signatures.

Boneh-Lynn-Shacham (BLS) Signatures

Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a symmetric pairing, and let P be a fixed generator of $\mathbb{G}$.

1. Key generation for Alice:
   - Private key: $x \in_R [1, n-1]$; Public key: $X = xP$.
2. Signature generation. To sign M, Alice does:
   - Compute $Q = H(M)$, where $H : \{0,1\}^* \to \mathbb{G}$.
   - Compute $S = xQ$.

   Alice's signature on M is S.
3. Signature verification. To verify (M, S), Bob does:
   - Compute $Q = H(M)$.
   - Accept iff $e(P, S) = e(Q, X)$.

Correctness:

$$e(P, S) = e(P, xQ) = e(xP, Q) = e(X, Q) = e(Q, X).$$

BLS Security

DHP Given $X = xP$ and Q, compute xQ.

Claim If DHP in $\mathbb{G}$ is hard and H is a random function, then the BLS signature scheme is secure.

Security argument Given a DHP instance (X, Q):

1. Set challenge public key as X and run BLS forger A.
2. Respond to hash queries $H(M)$ made by A, except for a randomly chosen distinguished query, by selecting $a \in_R [0, n)$ and setting $H(M) = aP$; the response to the distinguished hash query is $H(M^*) = Q$.
3. Respond to signing queries $M \neq M^*$ by setting $S = aX$.
4. If A eventually produces a forged signature S^* on M^*, then we have successfully obtained the solution S^* to the DHP instance (X, Q)

BLS-2

Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be the genus-2 pairing. Let $\mathcal{D}$ denote the set of degenerate divisors in $\mathbb{G}$, and let $\mathcal{P} \in \mathcal{D}$.

1. Key generation. Alice does:
   - Private key: $x \in_R [1, n)$; Public key: $X = x\mathcal{P}$.
2. Signature generation. To sign M, Alice does:
   - Compute $\mathcal{Q} = H(M)$, where $H : \{0, 1\}^* \to \mathcal{D}$.
   - Compute $S = x\mathcal{Q}$.

   Alice's signature on M is S.
3. Signature verification. To verify (M, S), Bob does:
   - Compute $\mathcal{Q} = H(M)$.
   - Accept iff $e(\mathcal{P}, S) = e(\mathcal{Q}, X)$.

DHP*: Given $X = x\mathcal{P}$ and $\mathcal{Q}$, compute $x\mathcal{Q}$.

DHP versus DHP*

Let $\mathcal{P}$ be a fixed generator of G.

DHP: Given $X = x\mathcal{P}$ and Q, compute xQ.

DHP*: Given $X = x\mathcal{P}$ and $\mathcal{Q}$, compute $x\mathcal{Q}$.

DHP and DHP* are computationally equivalent.

Degeneracy-Preserving Multipliers Let $\mathcal{P}$ be an order-n degenerate divisor.

$$\text{DPM} = \{a \in [0, n) : a\mathcal{P} \text{ is degenerate}\}.$$

$8^i\mathcal{P}$ is degenerate. Since $8^{4m} \equiv 1 \pmod{n}$, there are exactly $4m$ degenerate divisors of this form.

Question: Can one efficiently select $a \in_R$ DPM?

BLS-2 Security

DHP*: Given $X = x\mathcal{P}$ and $\mathcal{Q}$, compute $x\mathcal{Q}$.

Claim Suppose that one can efficiently select $a \in_R$ DPM. If DHP* in $\mathbb{G}$ is hard and H is a random function, then the BLS-2 signature scheme is secure.

Security argument Given DHP* instance $(X, \mathcal{Q})$:

1. Set the challenge public key as X and run A.
2. Respond to hash queries $H(M)$ made by A, except for a randomly chosen distinguished query, by selecting $a \in_R$ DPM and setting $H(M) = a\mathcal{P}$; the response to the distinguished query is $H(M^*) = \mathcal{Q}$.
3. Respond to signing queries $M \neq M^*$ with $S = aX$.
4. If A eventually produces a forged signature S^* on M^*, then we have obtained the solution S^* to the DHP* instance $(X, \mathcal{Q})$.

BLS-2 Security

Perhaps: introduce a new problem to circumvent issue with security argument.

DHP$^*_{\mathcal{O}}$: Given $X = x\mathcal{P}$ and $\mathcal{Q}$, plus an oracle which returns random pairs $(\mathcal{R}, x\mathcal{R})$, compute $x\mathcal{Q}$.

Claim: If DHP$^*_{\mathcal{O}}$ in $\mathbb{G}$ is hard and H is a random function, then the BLS-2 signature scheme is secure.

But...assumption that DHP$^*_{\mathcal{O}}$ is hard is rephrasing of the assertion that it is hard to forge signature.

BLS-3: Choose only the parameter $\mathcal{P}$ to be degenerate. Verification: $e(\mathcal{P}, S) = e(Q, X)$.

Claim: BLS-3 is secure if DHP is hard and H is a random function.

Summary for genus-2

In most favorable case (BLS-2), verification is $e(\mathcal{P}, S) = e(\mathcal{Q}, X)$ with $\mathcal{P}$ and $\mathcal{Q}$ degenerate.

- Estimates are that these are factor 2 more expensive than degenerate-degenerate.
- Optimization from Aranha, Beuchat, Detrey, Estibals give factor 1.7 advantage to genus 2 degenerate-degenerate vs EC.

So, genus-2 not exactly compelling for speed in this example.

Genus-2 looks stronger in [ABDE], in part due to 367-bit base field rather than 439-bit (elements fit in 3 rather than 4 128-bit registers).

- Pairing on degenerate divisors is factor 3 faster than pairing over $E(\mathbb{F}_{2^{1223}})$.

Selected references (1/2)

- D. Aranha, K. Karabina, P. Longa, C. Gebotys, and J. López, Faster Explicit Formulas for Computing Pairings over Ordinary Curves. Cryptology ePrint Archive 2010.
- D. Aranha, J.-L. Beuchat, J. Detrey, and N. Estibals, Optimal eta pairing on supersingular genus-2 binary hyperelliptic curves. Cryptology ePrint Archive 2010.
- D. Aranha, J. López, and D. Hankerson, High-speed parallel software implementation of the η_T pairing, CT-RSA 5985:89-105, 2010.
- J.-L. Beuchat, J. Díaz, S. Mitsunari, E. Okamoto, F. Rodríguez-Henríquez, and T. Teruya, High-speed software implementation of the optimal ate pairing over Barreto-Naehrig curves. Pairing 2010, LNCS.
- S. Chatterjee, D. Hankerson, and A. Menezes, On the efficiency and security of pairing-based protocols in the Type 1 and Type 4 settings. WAIFI 2010, LNCS 6087:114–134.

Selected references (2/2)

- A. Fog, Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs, http://www.agner.org, 2010.
- J. Grabher, J. Großschädl, and D. Page, On software parallel implementation of cryptographic pairings. Cryptology ePrint Archive 2008/205.
- S. Gueron and M. Kounavis, A technique for accelerating characteristic 2 elliptic curve cryptography. ITNG 2008.
- E. Lee and Y. Lee, Tate pairing computation on the divisors of hyperelliptic curves of genus 2. Journal of the Korean Mathematical Society 45(4):1057–1073, 2008.
- M. Naehrig, R. Niederhagen, and P. Schwabe, New software speed records for cryptographic pairings. LATINCRYPT 2010, LNCS 6612:109–123.