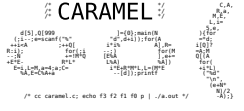
Rennes, November 17, 2010

Compact hardware for computing the Tate pairing over 128-bit-security supersingular curves

Nicolas Estibals

CARAMEL project-team, LORIA, Nancy Université / CNRS / INRIA

Nicolas.Estibals@loria.fr



Nancy-Université

Outline of the talk

- Context
- Choosing curves suited to pairings with 128 bits of security
- Implementation of the field arithmetic
- Results and conclusion

Pairing-based cryptography

- Origin of pairings in cryptography
  - attack against some elliptic curves
    - ⋆ Menezes–Okamoto–Vanstone, 1993
    - ⋆ Frey–Rück, 1994

Pairing-based cryptography

- Origin of pairings in cryptography
  - attack against some elliptic curves
    - ⋆ Menezes–Okamoto–Vanstone, 1993
    - ⋆ Frey–Rück, 1994
- Constructive properties
  - one-round three-party key exchange
    - ⋆ Joux, 2000
  - short digital signature
    - ⋆ Boneh–Lynn–Shacham, 2001
    - ⋆ Zang–Safavi–Naini–Susilo, 2004
  - identity-based encryption
    - ⋆ Boneh–Franklin, 2001
    - ⋆ Sakai–Kasahara, 2001
  - ...

Pairing-based cryptography

- Origin of pairings in cryptography
  - attack against some elliptic curves
    - ⋆ Menezes–Okamoto–Vanstone, 1993
    - ⋆ Frey–Rück, 1994
- Constructive properties
  - one-round three-party key exchange
    - ⋆ Joux, 2000
  - short digital signature
    - ⋆ Boneh–Lynn–Shacham, 2001
    - ⋆ Zang–Safavi–Naini–Susilo, 2004
  - identity-based encryption
    - ⋆ Boneh–Franklin, 2001
    - ⋆ Sakai–Kasahara, 2001
  - ...
- Standardization in progress
  - ISO/IEC 14888-3
  - IEEE P1363.3

Hardware accelerators for pairing computation

- Pairings everywhere!
  - wide range of targets and applications
    - ⋆ low-resource environment (embedded systems, smart card, . . .)
    - ⋆ high-performance computation (bank server, . . .)
  - non-trivial to compute
    - ⋆ complex mathematical structure
    - ⋆ finite field arithmetic
    - ⋆ substantial amount of computation
- Needs in hardware implementation
  - computation not suited to general purpose processor
  - specific target (*e.g.* credit card)

Hardware accelerators for pairing computation

- Pairings everywhere!
  - wide range of targets and applications
    - ⋆ low-resource environment (embedded systems, smart card, ...)
    - ⋆ high-performance computation (bank server, ...)
  - non-trivial to compute
    - ⋆ complex mathematical structure
    - ⋆ finite field arithmetic
    - ⋆ substantial amount of computation
- Needs in hardware implementation
  - computation not suited to general purpose processor
  - specific target (*e.g.* credit card)
- Previous work on FPGA implementations
  - low-security pairings
  - most are performance-oriented designs
- Our goal:
  - AES-128 equivalent security
  - compact accelerator

Tate pairing

- Bilinear pairing:

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$$

Tate pairing

- E elliptic curve over $\mathbb{F}_q$
- ℓ large prime dividing $\#E(\mathbb{F}_q)$
  - in general, $\ell \approx \#E(\mathbb{F}_q)$
  - Hasse's bound : $|\#E(\mathbb{F}_q) - (q+1)| \leqslant 2\sqrt{q}$
  - thus, $\ell \approx q$
- $\mathbb{F}_q$-rational ℓ-torsion of E: $E(\mathbb{F}_q)[\ell] = \{P \in E(\mathbb{F}_q) \mid [\ell]P = \mathcal{O}\}$
- Tate pairing:

$$e : E(\mathbb{F}_q)[\ell] \times \mathbb{G}_2 \to \mathbb{G}_T$$

Tate pairing

- E elliptic curve over $\mathbb{F}_q$
- ℓ large prime dividing $\#E(\mathbb{F}_q)$
  - in general, $\ell \approx \#E(\mathbb{F}_q)$
  - Hasse's bound : $|\#E(\mathbb{F}_q) - (q+1)| \leqslant 2\sqrt{q}$
  - thus, $\ell \approx q$
- $\mathbb{F}_q$-rational ℓ-torsion of E: $E(\mathbb{F}_q)[\ell] = \{P \in E(\mathbb{F}_q) \mid [\ell]P = \mathcal{O}\}$
- Embedding degree: k, the smallest integer s. t. $\ell \mid q^k - 1$
- Tate pairing:

$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \to \mathbb{G}_T$$

Tate pairing

- E elliptic curve over $\mathbb{F}_q$
- ℓ large prime dividing $\#E(\mathbb{F}_q)$
  - in general, $\ell \approx \#E(\mathbb{F}_q)$
  - Hasse's bound : $|\#E(\mathbb{F}_q) - (q+1)| \leqslant 2\sqrt{q}$
  - thus, $\ell \approx q$
- $\mathbb{F}_q$-rational ℓ-torsion of E: $E(\mathbb{F}_q)[\ell] = \{P \in E(\mathbb{F}_q) \mid [\ell]P = \mathcal{O}\}$
- Embedding degree: k, the smallest integer s. t. $\ell \mid q^k - 1$
- Set of ℓ-th root of unity: $\mu_\ell = \{u \in \mathbb{F}_{q^k}^* \mid u^\ell = 1\}$
- Tate pairing:

$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \to \mu_\ell \subset \mathbb{F}_{q^k}^*$$

Compute the Tate pairing

$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \rightarrow \mu_\ell \subset \mathbb{F}_{q^k}^*$$

- Miller's algorithm
- Iterative algorithm
- Complexity:
  - number of iteration proportional to definition field size
  - a multiplication over $\mathbb{F}_{q^k}$ at each iteration

General attacks

$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \to \mu_\ell \subset \mathbb{F}_{q^k}^*$$

- Pollard's ρ on the torsion subgroup $E[\ell]$
  - $\sqrt{\pi\ell/2} \approx \sqrt{\pi q/2}$ group operations
  - complexity exponential in q

General attacks

$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \to \mu_\ell \subset \mathbb{F}_{q^k}^*$$

- Pollard's ρ on the torsion subgroup $E[\ell]$
  - $\sqrt{\pi\ell/2} \approx \sqrt{\pi q/2}$ group operations
  - complexity exponential in q
- Discrete logarithm in finite field multiplicative group $\mathbb{F}_{q^k}^*$
  - small characteristic: FFS $\to L_{q^k}[1/3, \sqrt[3]{32/9}]$
  - large characteristic: NFS $\to L_{q^k}[1/3, \sqrt[3]{64/9}]$
  - complexity subexponential in q^k

General attacks

$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \to \mu_\ell \subset \mathbb{F}_{q^k}^*$$

- Pollard's ρ on the torsion subgroup $E[\ell]$
  - $\sqrt{\pi\ell/2} \approx \sqrt{\pi q/2}$ group operations
  - complexity exponential in q
- Discrete logarithm in finite field multiplicative group $\mathbb{F}_{q^k}^*$
  - small characteristic: FFS $\to L_{q^k}[1/3, \sqrt[3]{32/9}]$
  - large characteristic: NFS $\to L_{q^k}[1/3, \sqrt[3]{64/9}]$
  - complexity subexponential in q^k
- k acts as a cursor to balance the complexity of the two attacks
- Optimal k for the 128-bit security level
  - 15 for small characteristic
  - 12 for large characteristic

Outline of the talk

- Context
- Choosing curves suited to pairings with 128 bits of security
- Implementation of the field arithmetic
- Results and conclusion

Supersingular elliptic curves Vs. Barreto–Naehrig curves

- Definitions:

$$E_{2,b}/\mathbb{F}_2 : \quad y^2 + y = x^3 + x + b$$
$$E_{3,b}/\mathbb{F}_3 : \quad y^2 = x^3 - x + b, b \neq 0$$

- Supersingular curve

$\Rightarrow$ Simpler curve arithmetic (sparser addition and doubling formulae)

- Definition:

$$E_{\alpha,b}/\mathbb{F}_p : \quad y^2 = x^3 + b, b \neq 0,$$
$$p = 36\alpha^4 - 36\alpha^3 + 24\alpha^2 - 6\alpha + 1$$

- Ordinary curve

Supersingular elliptic curves Vs. Barreto–Naehrig curves

- Definitions:

$$E_{2,b}/\mathbb{F}_2: \quad y^2 + y = x^3 + x + b$$
$$E_{3,b}/\mathbb{F}_3: \quad y^2 = x^3 - x + b, b \neq 0$$

- Supersingular curve

  $\Rightarrow$ Simpler curve arithmetic (sparser addition and doubling formulae)

- Distortion map, reduced pairing:

$$\delta : E(\mathbb{F}_q)[\ell] \to E(\mathbb{F}_{q^k})[\ell]$$
$$\hat{e}(P, Q) = e(P, \delta(Q))$$

  $\Rightarrow$ Symmetric pairing (BN cannot be used with all protocols)

- Definition:

$$E_{\alpha,b}/\mathbb{F}_p: \quad y^2 = x^3 + b, b \neq 0,$$
$$p = 36\alpha^4 - 36\alpha^3 + 24\alpha^2 - 6\alpha + 1$$

- Ordinary curve

- No distortion map

Supersingular elliptic curves Vs. Barreto–Naehrig curves

- Definitions:

  $E_{2,b}/\mathbb{F}_2 : \quad y^2 + y = x^3 + x + b$
  $E_{3,b}/\mathbb{F}_3 : \quad y^2 = x^3 - x + b, b \neq 0$

- Supersingular curve

  $\Rightarrow$ Simpler curve arithmetic (sparser addition and doubling formulae)

- Distortion map, reduced pairing:

  $\delta : E(\mathbb{F}_q)[\ell] \rightarrow E(\mathbb{F}_{q^k})[\ell]$
  $\hat{e}(P, Q) = e(P, \delta(Q))$

  $\Rightarrow$ Symmetric pairing (BN cannot be used with all protocols)

- Small characteristic field arithmetic

  $\Rightarrow$ No carry, better suited to hardware implementation

- Definition:

  $E_{\alpha,b}/\mathbb{F}_p : \quad y^2 = x^3 + b, b \neq 0,$
  $p = 36\alpha^4 - 36\alpha^3 + 24\alpha^2 - 6\alpha + 1$

- Ordinary curve

- No distortion map

- Modular arithmetic

Supersingular elliptic curves Vs. Barreto–Naehrig curves

- Definitions:

  $E_{2,b}/\mathbb{F}_2 : \quad y^2 + y = x^3 + x + b$

  $E_{3,b}/\mathbb{F}_3 : \quad y^2 = x^3 - x + b, b \neq 0$

- Supersingular curve

  $\Rightarrow$ Simpler curve arithmetic (sparser addition and doubling formulae)

- Distortion map, reduced pairing:

  $\delta : E(\mathbb{F}_q)[\ell] \rightarrow E(\mathbb{F}_{q^k})[\ell]$

  $\hat{e}(P, Q) = e(P, \delta(Q))$

  $\Rightarrow$ Symmetric pairing (BN cannot be used with all protocols)

- Small characteristic field arithmetic

  $\Rightarrow$ No carry, better suited to hardware implementation

- Small embedding degree ($k = 4$ or 6)

  $\Rightarrow$ Larger field of definition for the same security level. For 128 bits of security:

  $\mathbb{F}_q$ with $q \approx 2^{1150}$ or 3^{500}

- Definition:

  $E_{\alpha,b}/\mathbb{F}_p : \quad y^2 = x^3 + b, b \neq 0,$

  $p = 36\alpha^4 - 36\alpha^3 + 24\alpha^2 - 6\alpha + 1$

- Ordinary curve

- No distortion map

- Modular arithmetic

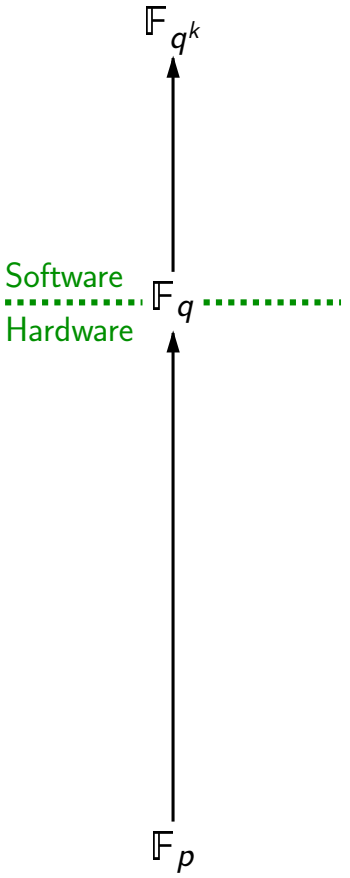
- Optimal embedding degree ($k = 12$)

  $\mathbb{F}_p$ with p a 256-bit prime.

Which field of definition?

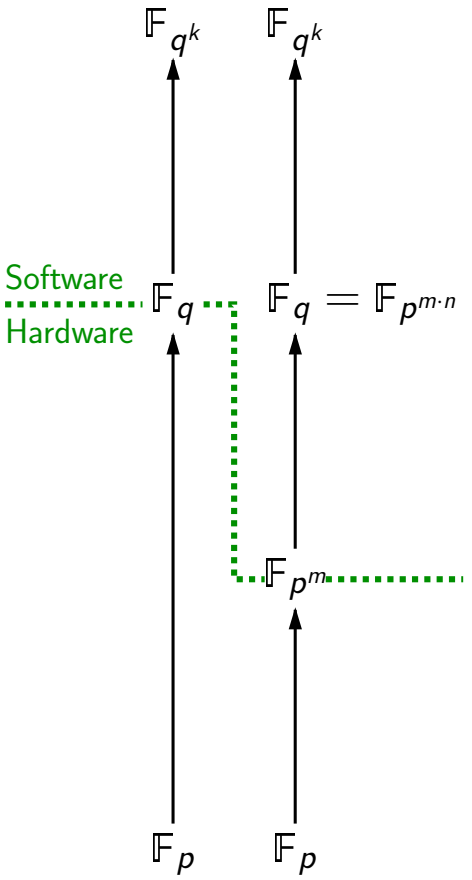
- Arithmetic of $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$:
  - tower field fixed by pairing construction
  - already optimized by previous works
  - Bottleneck in pairing computation: products in $\mathbb{F}_q$

Which field of definition?



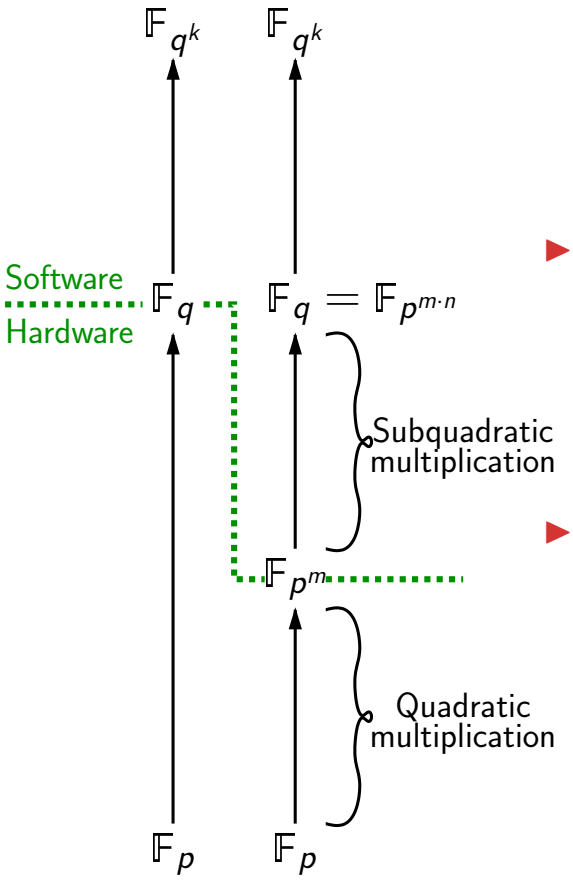
- Arithmetic of $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$:
  - tower field fixed by pairing construction
  - already optimized by previous works
  - Bottleneck in pairing computation: products in $\mathbb{F}_q$
- Arithmetic of $\mathbb{F}_q$
  - traditionally implemented in hardware
  - does not scale to the 128-bit security level
    - ⋆ slow multiplier, or
    - ⋆ large multiplier

Which field of definition?



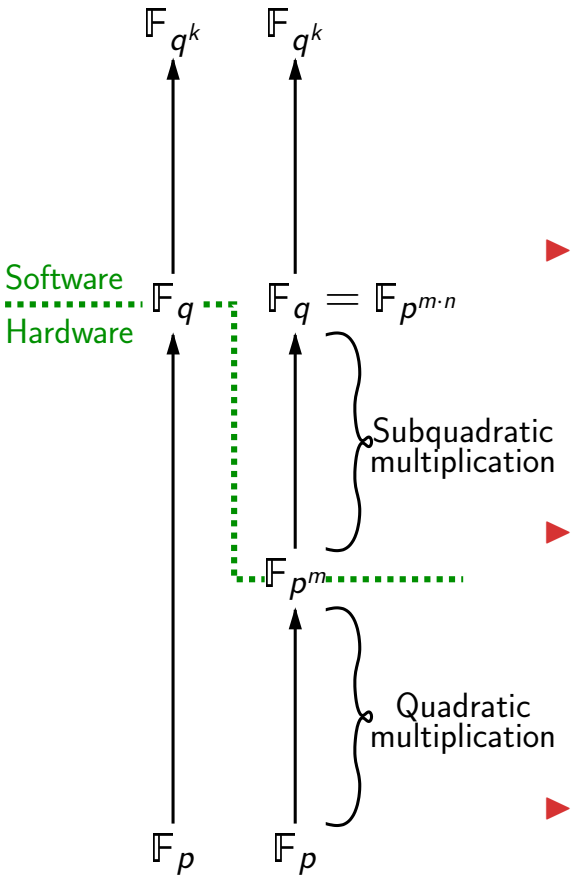
- Arithmetic of $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$:
  - tower field fixed by pairing construction
  - already optimized by previous works
  - Bottleneck in pairing computation: products in $\mathbb{F}_q$
- Arithmetic of $\mathbb{F}_q$
  - traditionally implemented in hardware
  - does not scale to the 128-bit security level
    - ⋆ slow multiplier, or
    - ⋆ large multiplier
- Idea: lower the soft/hardware frontier
  - insert $\mathbb{F}_{p^m}$ in the tower field
  - implement it in hardware

Which field of definition?



- Arithmetic of $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$:
  - tower field fixed by pairing construction
  - already optimized by previous works
  - Bottleneck in pairing computation: products in $\mathbb{F}_q$
- Arithmetic of $\mathbb{F}_q$
  - traditionally implemented in hardware
  - does not scale to the 128-bit security level
    - ⋆ slow multiplier, or
    - ⋆ large multiplier
- Idea: lower the soft/hardware frontier
  - insert $\mathbb{F}_{p^m}$ in the tower field
  - implement it in hardware
  - use subquadratic multiplication algorithm for $\mathbb{F}_q$ over $\mathbb{F}_{p^m}$

Which field of definition?



- Arithmetic of $\mathbb{F}_{q^k}$ over $\mathbb{F}_q$:
  - tower field fixed by pairing construction
  - already optimized by previous works
  - Bottleneck in pairing computation: products in $\mathbb{F}_q$
- Arithmetic of $\mathbb{F}_q$
  - traditionally implemented in hardware
  - does not scale to the 128-bit security level
    - ⋆ slow multiplier, or
    - ⋆ large multiplier
- Idea: lower the soft/hardware frontier
  - insert $\mathbb{F}_{p^m}$ in the tower field
  - implement it in hardware
  - use subquadratic multiplication algorithm for $\mathbb{F}_q$ over $\mathbb{F}_{p^m}$
- Problem:
  - field with composite extension degree
  - allows some supplementary attacks

Weil Descent and Gaudry–Hess–Smart attack

- We now consider:
$$E(\mathbb{F}_{p^{m \cdot n}})[\ell] \text{ with } m \text{ prime and } n \text{ small}$$
- Weil descent (or Weil restriction to scalar) apply:
$$E(\mathbb{F}_{p^{m \cdot n}}) \cong W_E(\mathbb{F}_{p^m}) \cong E(\mathbb{F}_{p^m}) \times B(\mathbb{F}_{p^m})$$
  - $B(\mathbb{F}_{p^m})$ is called the Trace-Zero Variety
  - $E(\mathbb{F}_{p^{m \cdot n}})[\ell]$ is a subgroup of the TZV

Weil Descent and Gaudry–Hess–Smart attack

- We now consider:
$$E(\mathbb{F}_{p^{m \cdot n}})[\ell] \text{ with } m \text{ prime and } n \text{ small}$$
- Weil descent (or Weil restriction to scalar) apply:
$$E(\mathbb{F}_{p^{m \cdot n}}) \cong W_E(\mathbb{F}_{p^m}) \cong E(\mathbb{F}_{p^m}) \times B(\mathbb{F}_{p^m})$$
  - $B(\mathbb{F}_{p^m})$ is called the Trace-Zero Variety
  - $E(\mathbb{F}_{p^{m \cdot n}})[\ell]$ is a subgroup of the TZV
- Gaudry–Hess–Smart attack:
  - $W_E(\mathbb{F}_{p^m})$ might map to $\mathrm{Jac}(\mathcal{C})$, with $\mathcal{C}$ a curve of genus at least n
  - index calculus algorithm: solve DLP in $\tilde{O}((p^m)^{2-\frac{2}{n}})$

Weil Descent and Gaudry–Hess–Smart attack

- We now consider:
$$E(\mathbb{F}_{p^{m \cdot n}})[\ell] \text{ with } m \text{ prime and } n \text{ small}$$
- Weil descent (or Weil restriction to scalar) apply:
$$E(\mathbb{F}_{p^{m \cdot n}}) \cong W_E(\mathbb{F}_{p^m}) \cong E(\mathbb{F}_{p^m}) \times B(\mathbb{F}_{p^m})$$
  - $B(\mathbb{F}_{p^m})$ is called the Trace-Zero Variety
  - $E(\mathbb{F}_{p^{m \cdot n}})[\ell]$ is a subgroup of the TZV
- Gaudry–Hess–Smart attack:
  - $W_E(\mathbb{F}_{p^m})$ might map to $\mathrm{Jac}(\mathcal{C})$, with $\mathcal{C}$ a curve of genus at least n
  - index calculus algorithm: solve DLP in $\tilde{O}((p^m)^{2-\frac{2}{n}})$
  - general case:
    - ⋆ attacks also run in $\tilde{O}((p^m)^{2-\frac{2}{n}})$
    - ⋆ but $\tilde{O}(.)$ notation hide bad dependency in n

Static Diffie–Hellman problem

- Static Diffie–Hellman problem:
  - given $P, [d]P \in E$ where d is secret
  - the oracle $Q \mapsto [d]Q$
  - compute $[d]R$ where R is a random point
- Leakage when reusing private key
  - ElGamal encryption scheme

Static Diffie–Hellman problem

- Static Diffie–Hellman problem:
  - given $P, [d]P \in E$ where d is secret
  - the oracle $Q \mapsto [d]Q$
  - compute $[d]R$ where R is a random point
- Leakage when reusing private key
  - ElGamal encryption scheme
- Index calculus based algorithm (Granger, 2010)
  - construct factor base thanks to the oracle
  - time complexity: $\tilde{O}((p^m)^{1-\frac{1}{n+1}})$
  - $O((p^m)^{1-\frac{1}{n+1}})$ calls to the oracle
- Simple and efficient workaround
  - revoke key after a certain amount of use

Suitable curves for 128-bit security level

				Cost of the attacks (bits)			
p^m	n	b	$\log_2 \ell$	Pollard's ρ	FFS	GHS	SDH
3^{503}	1	1	697	342	**132**	–	–
3^{97}	5	-1	338	163	**130**	245	128
3^{67}	7	-1	612	300	**129**	182	92
3^{53}	11	-1	672	330	**140**	152	77
3^{43}	13	1	764	376	138	**125**	63

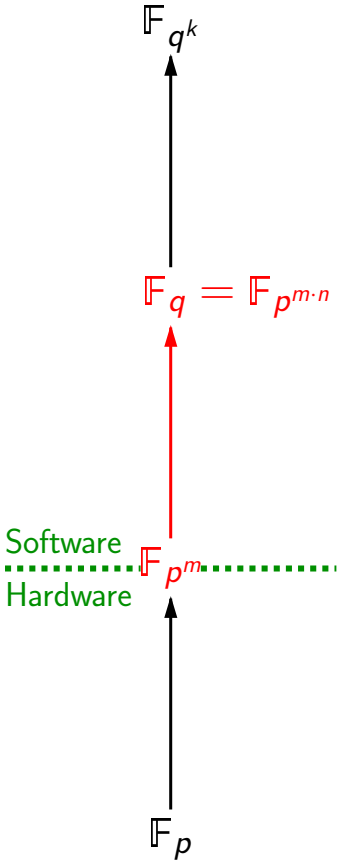
Suitable curves for 128-bit security level

				Cost of the attacks (bits)			
p^m	n	b	$\log_2 \ell$	Pollard's ρ	FFS	GHS	SDH
3^{503}	1	1	697	342	**132**	–	–
3^{97}	5	-1	338	163	**130**	245	128
3^{67}	7	-1	612	300	**129**	182	92
3^{53}	11	-1	672	330	**140**	152	77
3^{43}	13	1	764	376	138	**125**	63
2^{1117}	1	1	1076	531	**128**	–	–
2^{367}	3	1	698	342	**128**	489	275
2^{227}	5	1	733	359	**129**	363	189
2^{163}	7	1	753	370	**129**	279	142
2^{127}	9	1	487	236	**130**	225	114
2^{103}	11	1	922	454	**129**	187	94
2^{89}	13	0	1044	515	164	**130**	82
2^{73}	15	0	492	239	136	**127**	68

Outline of the talk

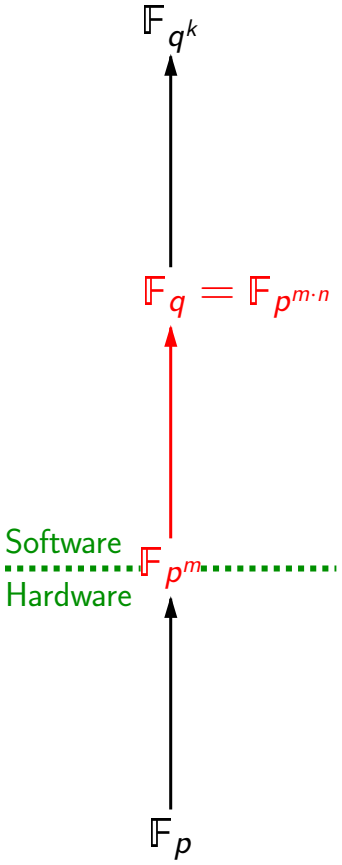
- Context
- Choosing curves suited to pairings with 128 bits of security
- Implementation of the field arithmetic
- Results and conclusion

Arithmetic of the extension field



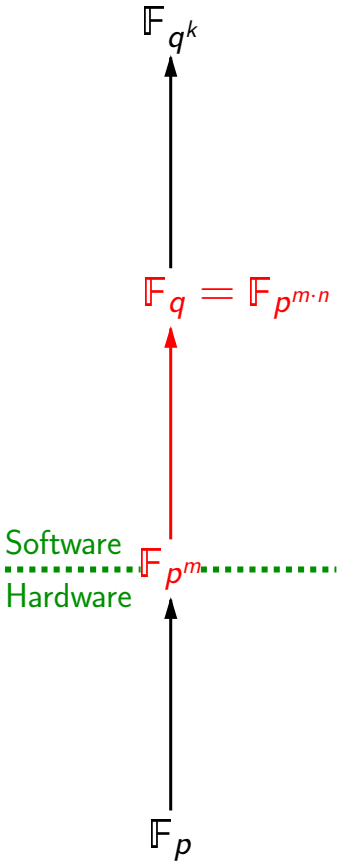
- Polynomial representation: $\mathbb{F}_{p^{m \cdot n}} \cong \mathbb{F}_{p^m}[X]/(f(X))$
  - f irreducible polynomial of degree n
  - f low Hamming weight (tri- or pentanomial)
  - $\mathbb{F}_{p^{m \cdot n}}$ represented by $\mathbb{F}_{p^m}[X]^{<n}$
- Addition:
  - add polynomials coefficient-wise
  - no reduction needed
- Frobenius automorphism: $(.)^p$
  - apply Frobenius on each coefficient
  - linear combination of the coefficients
- Inversion
  - Itoh & Tsujii algorithm (Fermat's little theorem)
  - only need additions, multiplications and Frobenius in $\mathbb{F}_{p^m}$

Multiplication in the extension field



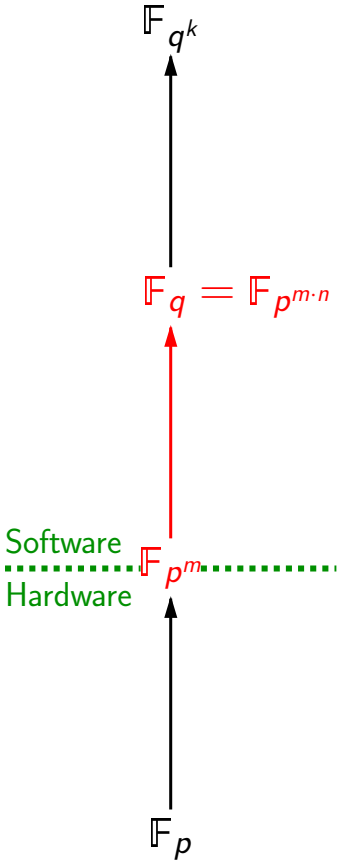
- Bottleneck of pairing computation
- Subquadratic multiplication algorithm
  - Karatsuba
  - Karatsuba with Montgomery's trick
  - Montgomery's formulae
  - CRT-based algorithms
  - ...

Multiplication in the extension field



- Bottleneck of pairing computation
- Subquadratic multiplication algorithm
  - Karatsuba
  - Karatsuba with Montgomery's trick
  - Montgomery's formulae
  - CRT-based algorithms
  - ...
  - Recursive use of those methods

Multiplication in the extension field



- Bottleneck of pairing computation
- Subquadratic multiplication algorithm
  - Karatsuba
  - Karatsuba with Montgomery's trick
  - Montgomery's formulae
  - CRT-based algorithms
  - ...
  - Recursive use of those methods
- Relative cost of addition against multiplication in $\mathbb{F}_{p^m}$ is not necessarily negligible:
  - because $\mathbb{F}_{p^m}$ is a not-so-large field
  - comparison of multiplication algorithms should integrate addition cost

Cost of different multiplication algorithms

Multiplication in $\mathbb{F}_{3^{97 \cdot 5}}$

Algorithm	$\times$	$+$	Ratio
Schoolbook	25	24	0.96
One-level Karatsuba (Montgomery's trick)	21	29	1.38
Recursive Karatsuba	15	39	2.60
Recursive Karatsuba (Montgomery's trick)	14	43	3.07
Montgomery's Karatsuba-like	13	54	4.153
CRT-based	12	53	4.42

Cost of different multiplication algorithms

Multiplication in $\mathbb{F}_{3^{97 \cdot 5}}$

Algorithm	$\times$	$+$	Ratio
Schoolbook	25	24	0.96
One-level Karatsuba (Montgomery's trick)	21	29	1.38
Recursive Karatsuba	15	39	2.60
Recursive Karatsuba (Montgomery's trick)	14	43	3.07
Montgomery's Karatsuba-like	13	54	4.153
CRT-based	12	53	4.42

Multiplication in $\mathbb{F}_{2^{163 \cdot 7}}$

Algorithm	$\times$	$+$	Ratio
Schoolbook	49	48	0.98
One-level Karatsuba (Montgomery's trick)	40	52	1.30
Recursive Karatsuba	25	51	2.04
Recursive Karatsuba (Montgomery's trick)	23	76	3.30
Montgomery's Karatsuba-like	22	84	3.818
CRT-based	22	88	4.05

Total Cost

- Full pairing computation over $E_{3,-1}(\mathbb{F}_{3^{97\cdot 5}})$

	$\times$	$+$	$(.)^3$	$1/.$
$\mathbb{F}_{3^{97\cdot 5}}$	3104	13127	4123	1

Total Cost

- Full pairing computation over $E_{3,-1}(\mathbb{F}_{3^{97\cdot 5}})$

	$\times$	$+$	$(.)^3$	$1/.$
$\mathbb{F}_{3^{97\cdot 5}}$	3104	13127	4123	1
$\mathbb{F}_{3^{97}}$	37248	253185	20615	1

Total Cost

- Full pairing computation over $E_{3,-1}(\mathbb{F}_{3^{97\cdot 5}})$

	$\times$	$+$	$(.)^3$	$1/.$
$\mathbb{F}_{3^{97\cdot 5}}$	3104	13127	4123	1
$\mathbb{F}_{3^{97}}$	37289	253314	21099	–

Total Cost

- Full pairing computation over $E_{3,-1}(\mathbb{F}_{3^{97\cdot 5}})$

	$\times$	$+$	$(.)^3$	$1/.$
$\mathbb{F}_{3^{97\cdot 5}}$	3104	13127	4123	1
$\mathbb{F}_{3^{97}}$	37289	253314	21099	–

- Full pairing computation over $E_{2,1}(\mathbb{F}_{2^{163\cdot 7}})$

	$\times$	$+$	$(.)^2$	$1/.$
$\mathbb{F}_{2^{163\cdot 7}}$	4019	12605	7424	1
$\mathbb{F}_{2^{163}}$	88418	448103	51968	1

Total Cost

- Full pairing computation over $E_{3,-1}(\mathbb{F}_{3^{97\cdot 5}})$

	$\times$	$+$	$(.)^3$	$1/.$
$\mathbb{F}_{3^{97\cdot 5}}$	3104	13127	4123	1
$\mathbb{F}_{3^{97}}$	37289	253314	21099	–

- Full pairing computation over $E_{2,1}(\mathbb{F}_{2^{163\cdot 7}})$

	$\times$	$+$	$(.)^2$	$1/.$
$\mathbb{F}_{2^{163\cdot 7}}$	4019	12605	7424	1
$\mathbb{F}_{2^{163}}$	88509	448361	52782	–

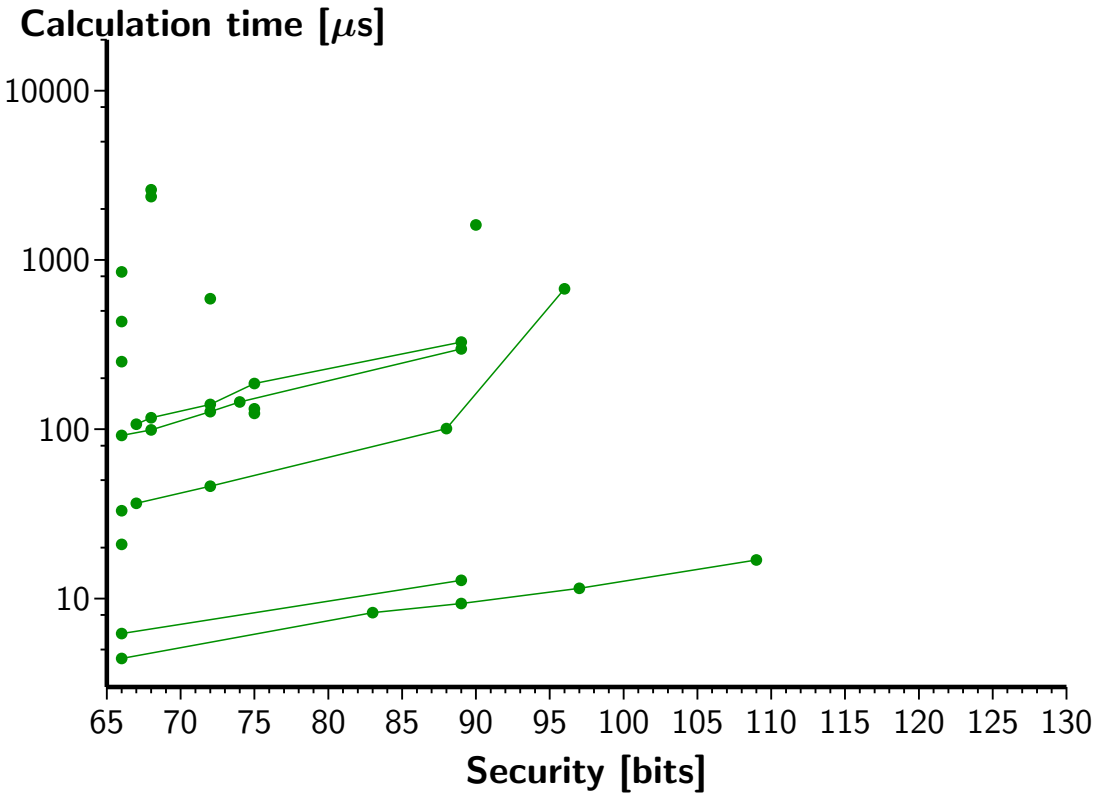
Outline of the talk

- Context
- Choosing curves suited to pairings with 128 bits of security
- Implementation of the field arithmetic
- Results and conclusion

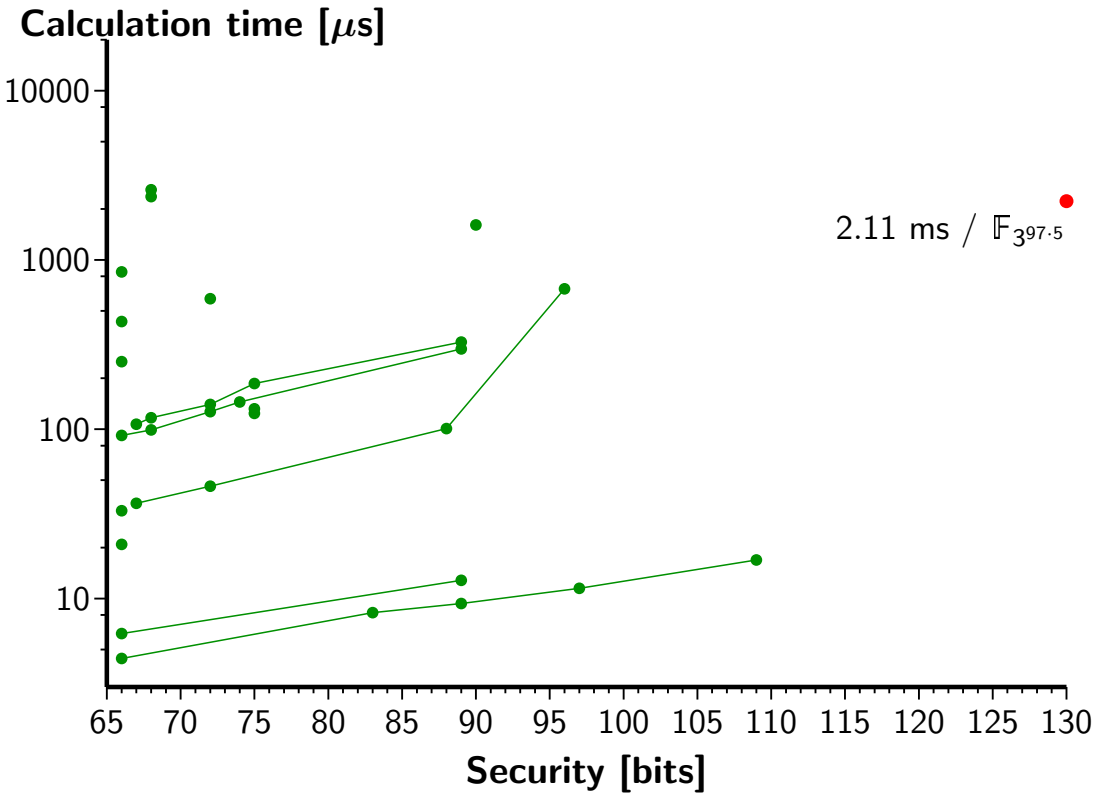
Experimental setup

- Full Tate pairing computation:
  - Miller's loop and
  - final exponentiation
- Implemented for the two following curves:
  - $E_{3,-1}(\mathbb{F}_{3^{97\cdot 5}})$
  - $E_{2,1}(\mathbb{F}_{2^{163\cdot 7}})$
- Prototyped on Xilinx Virtex-4 LX FPGAs
- Post-place-and-route timing and area estimations

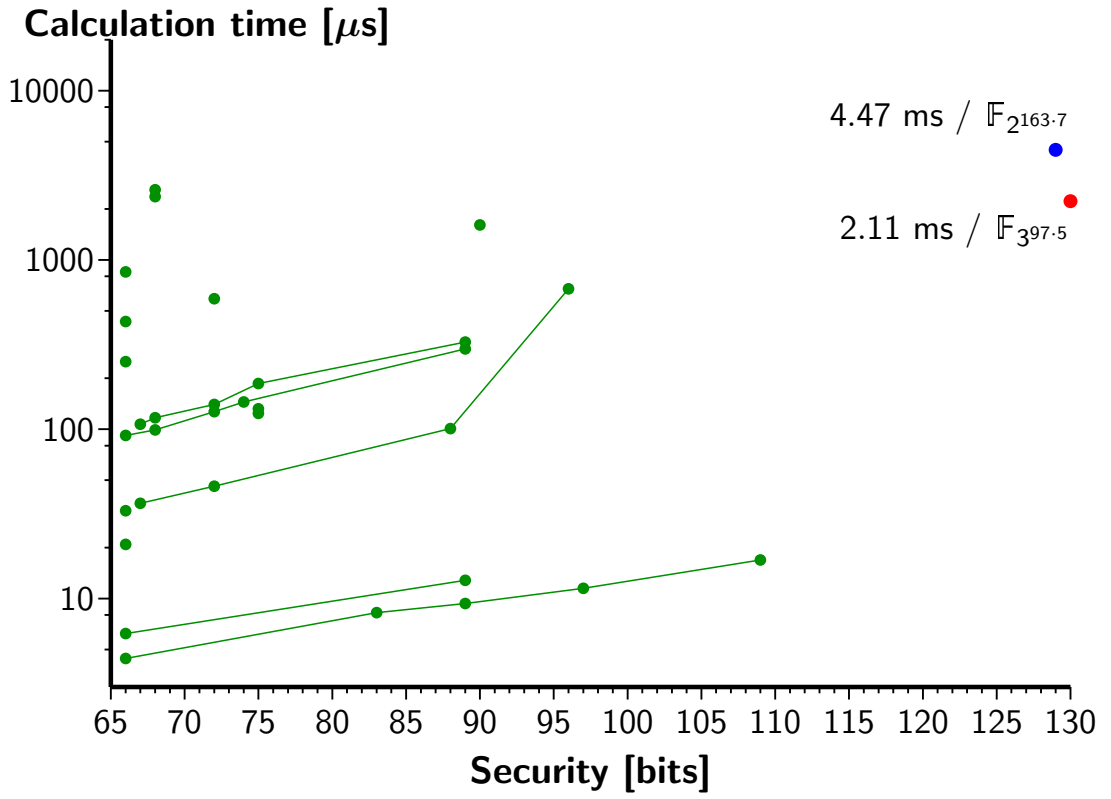
Calculation time



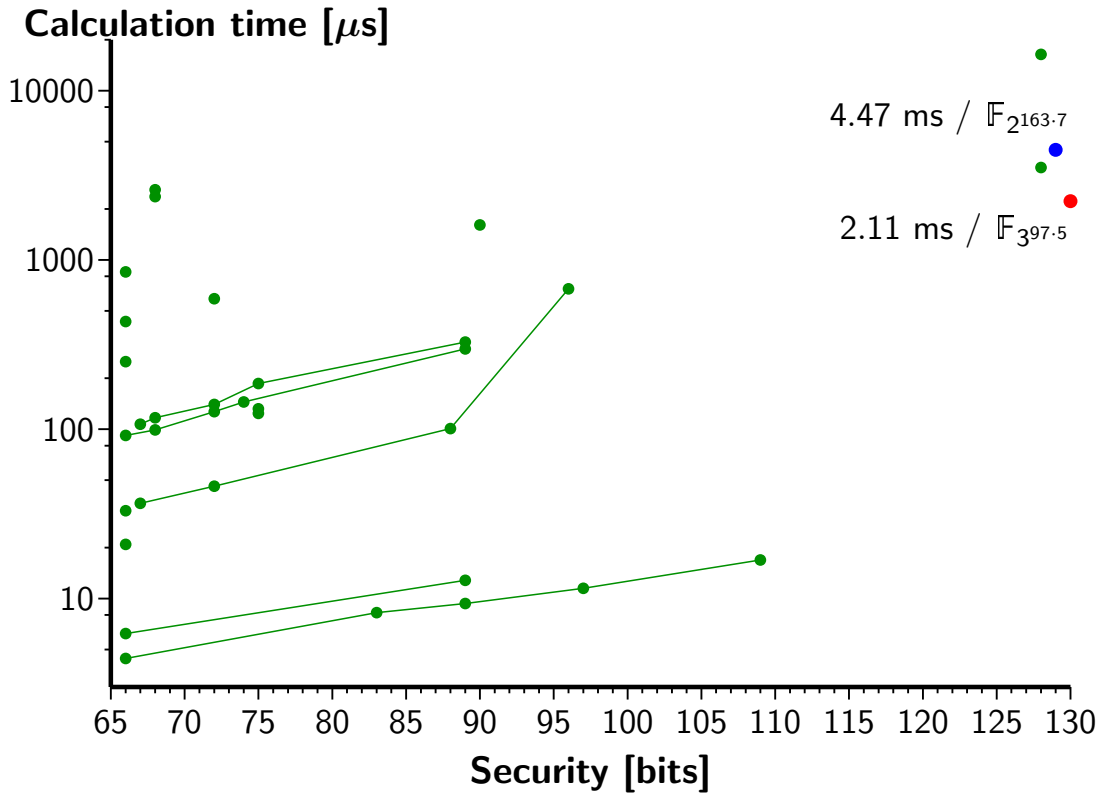
Calculation time



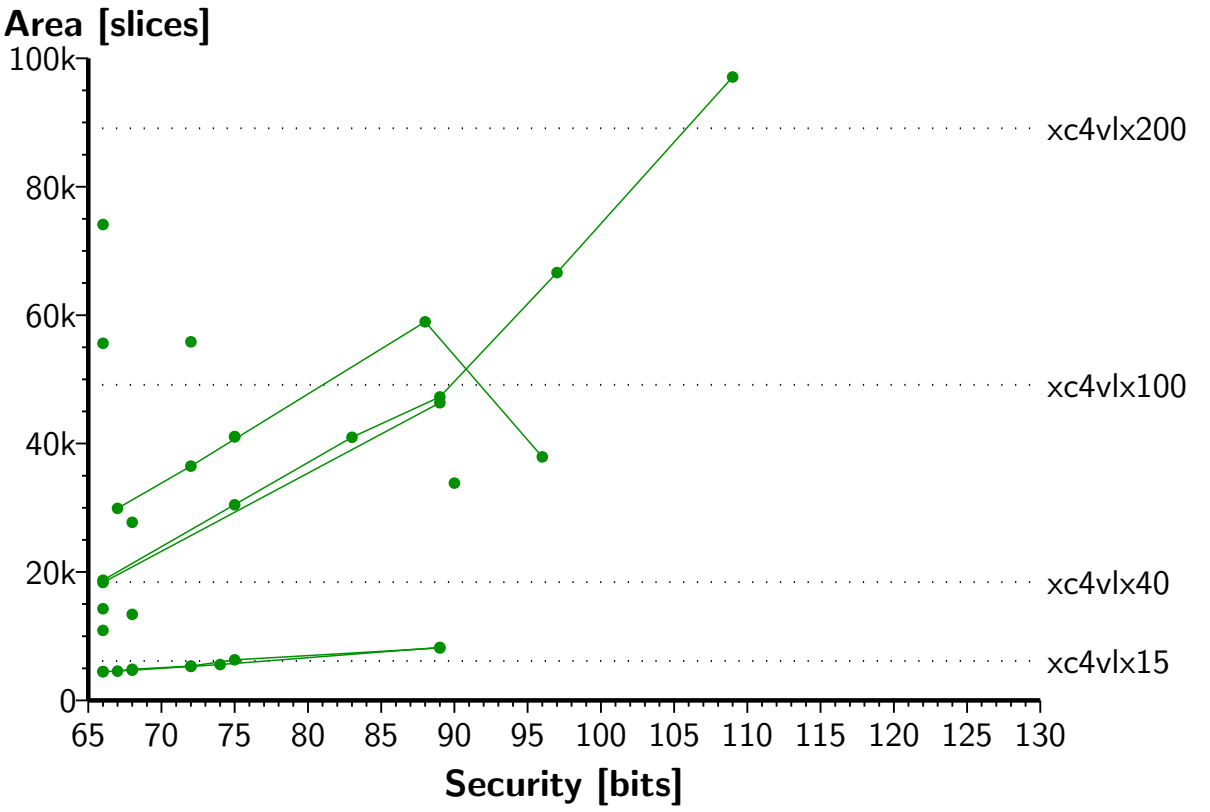
Calculation time



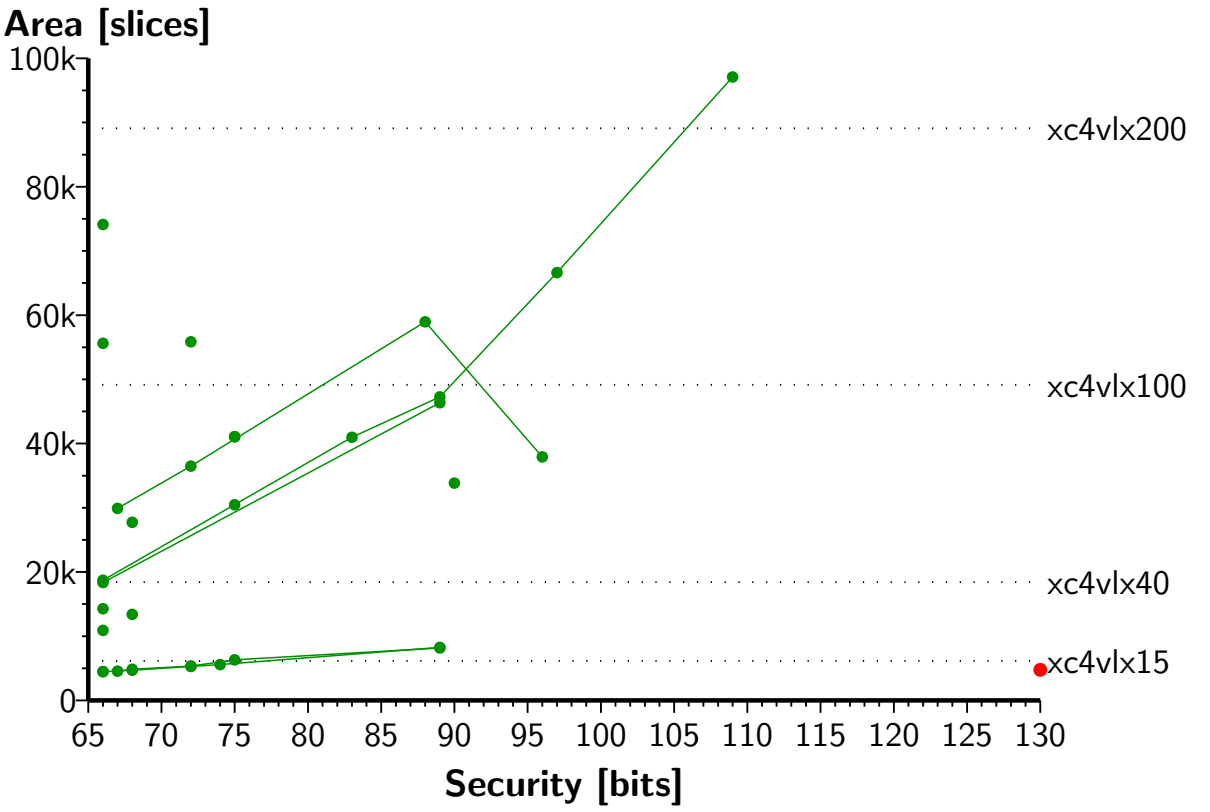
Calculation time



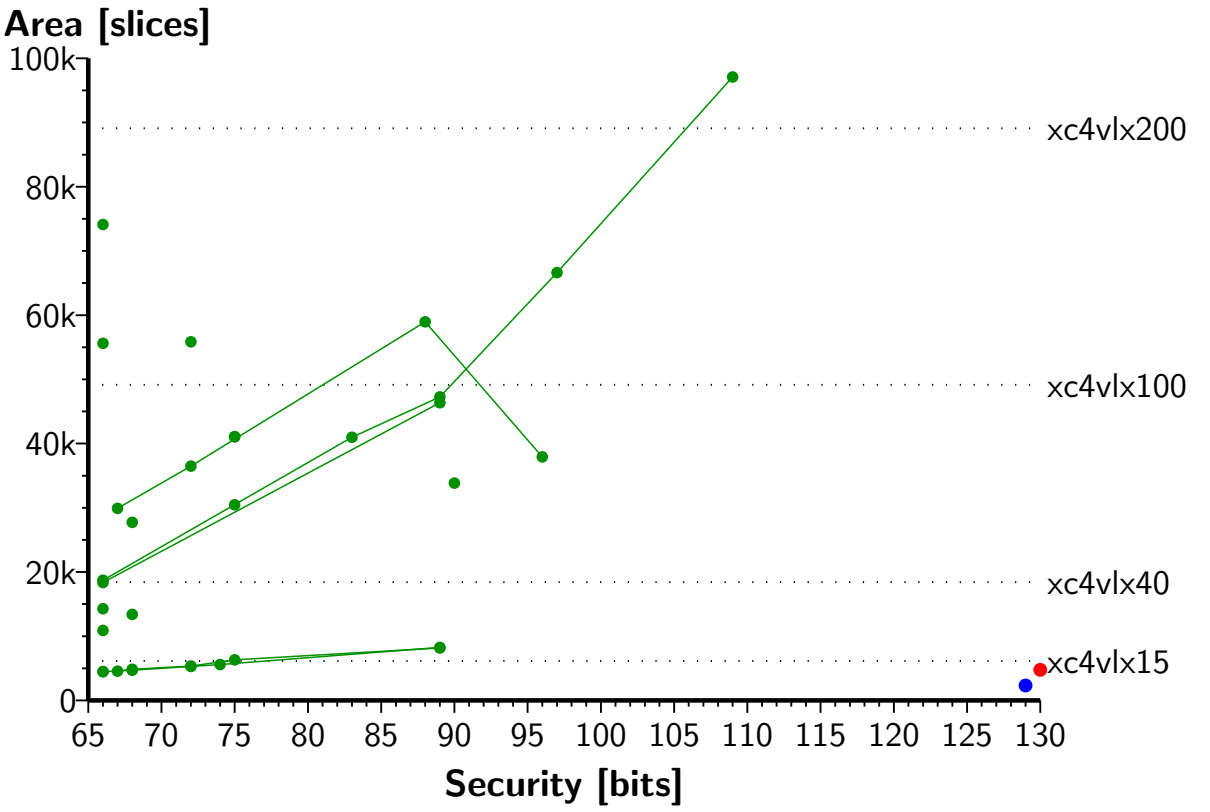
Area


Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc4vlx100
xc4vlx40
xc4vlx15
65
70
75
80
85
90
95
100
105
110
115
120
125
130
Security [bits]

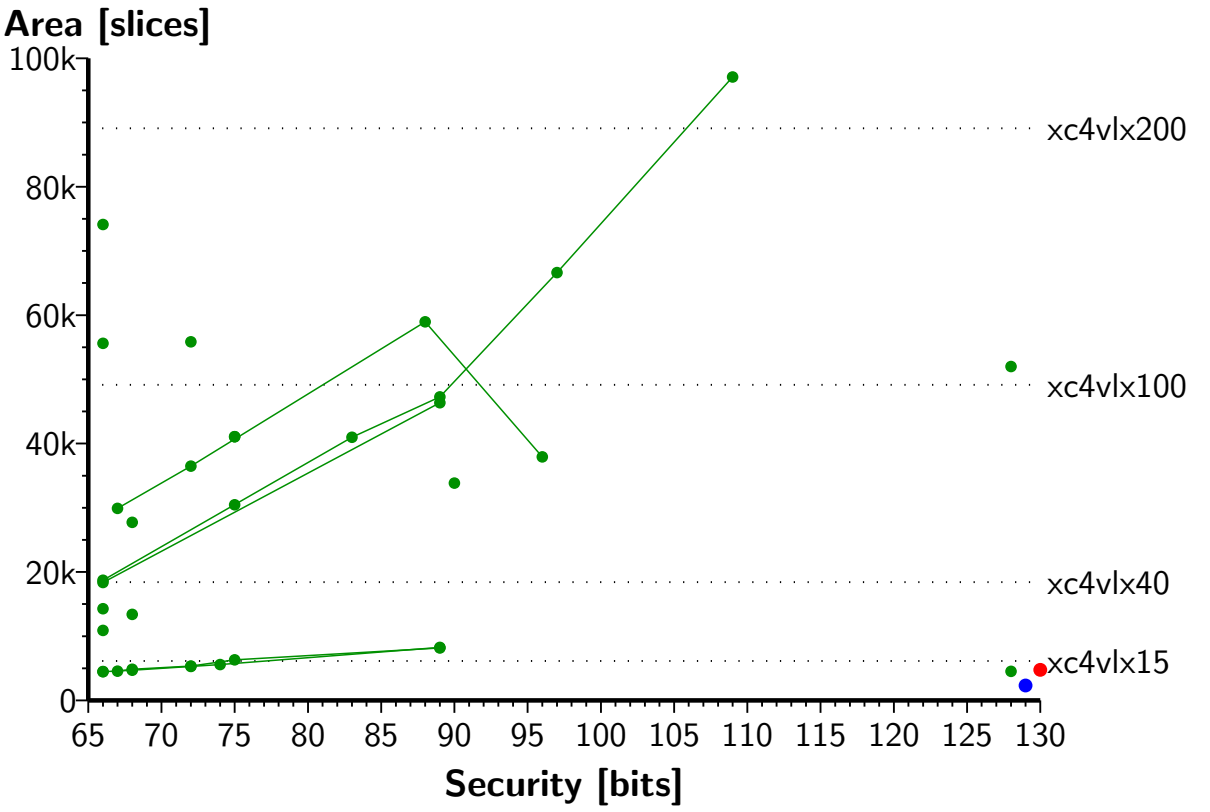
Area


Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc4vlx100
xc4vlx40
xc4vlx15
65
70
75
80
85
90
95
100
105
110
115
120
125
130
Security [bits]

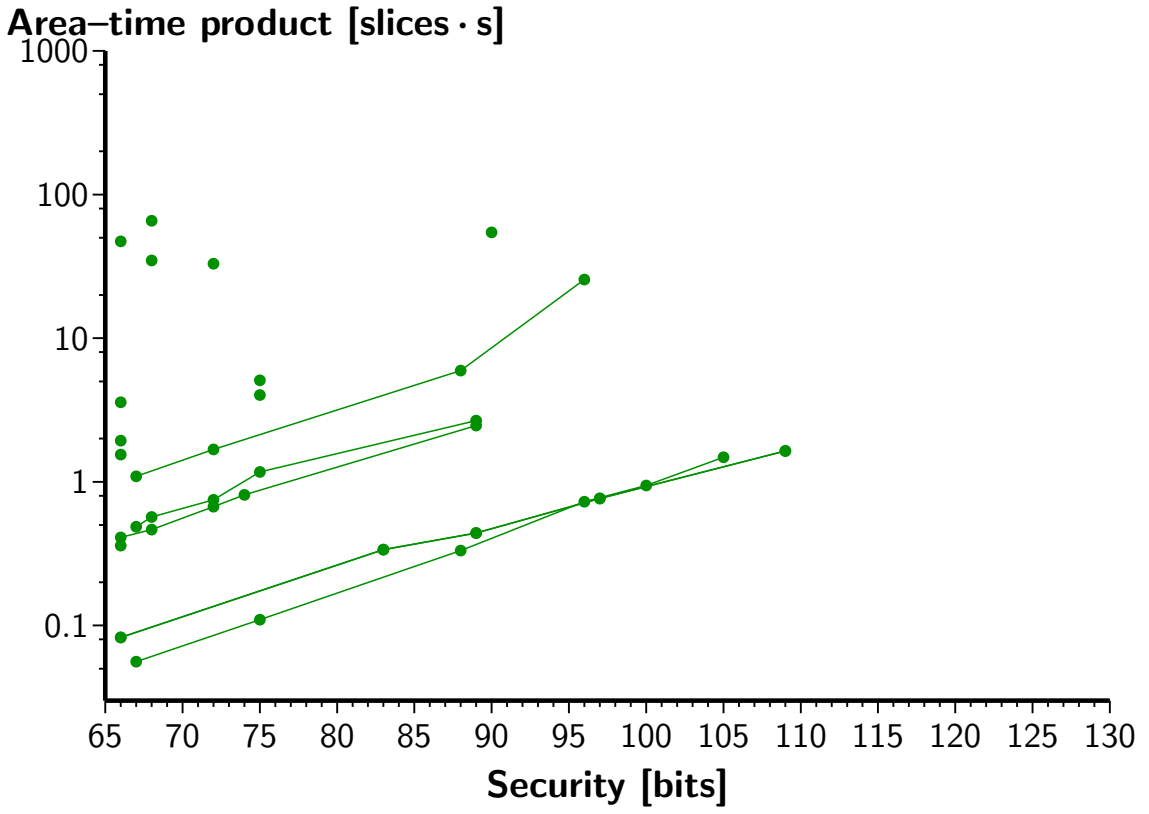
Area


Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc4vlx100
xc4vlx40
xc4vlx15
65
70
75
80
85
90
95
100
105
110
115
120
125
130
Security [bits]

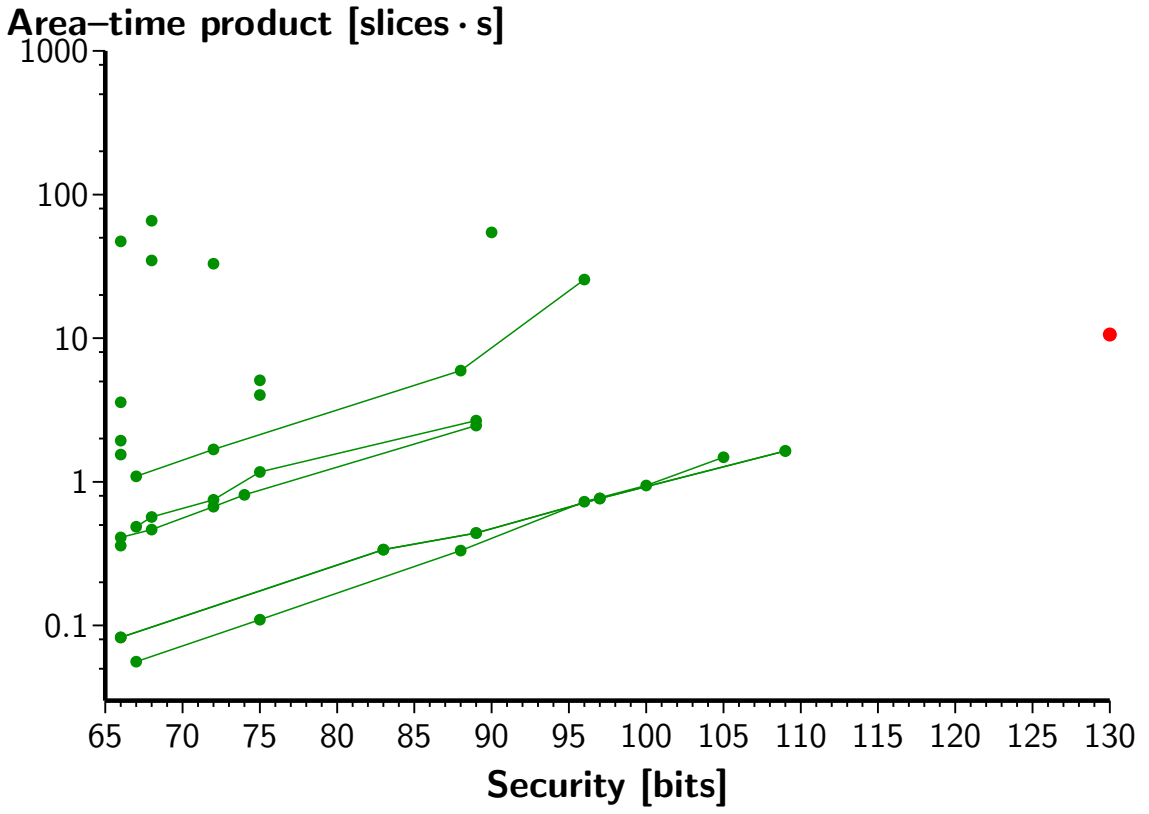
Area


Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc4vlx100
xc4vlx40
xc4vlx15
65
70
75
80
85
90
95
100
105
110
115
120
125
130
Security [bits]

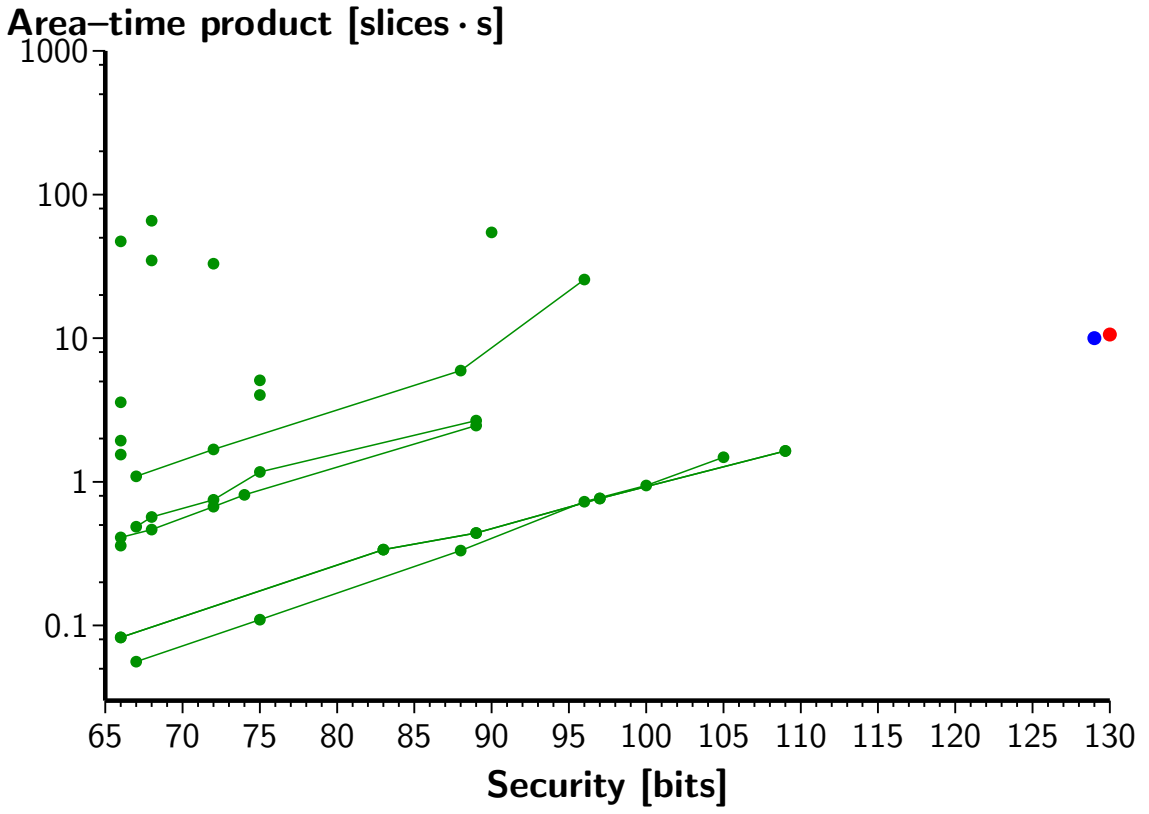
Area–Time product



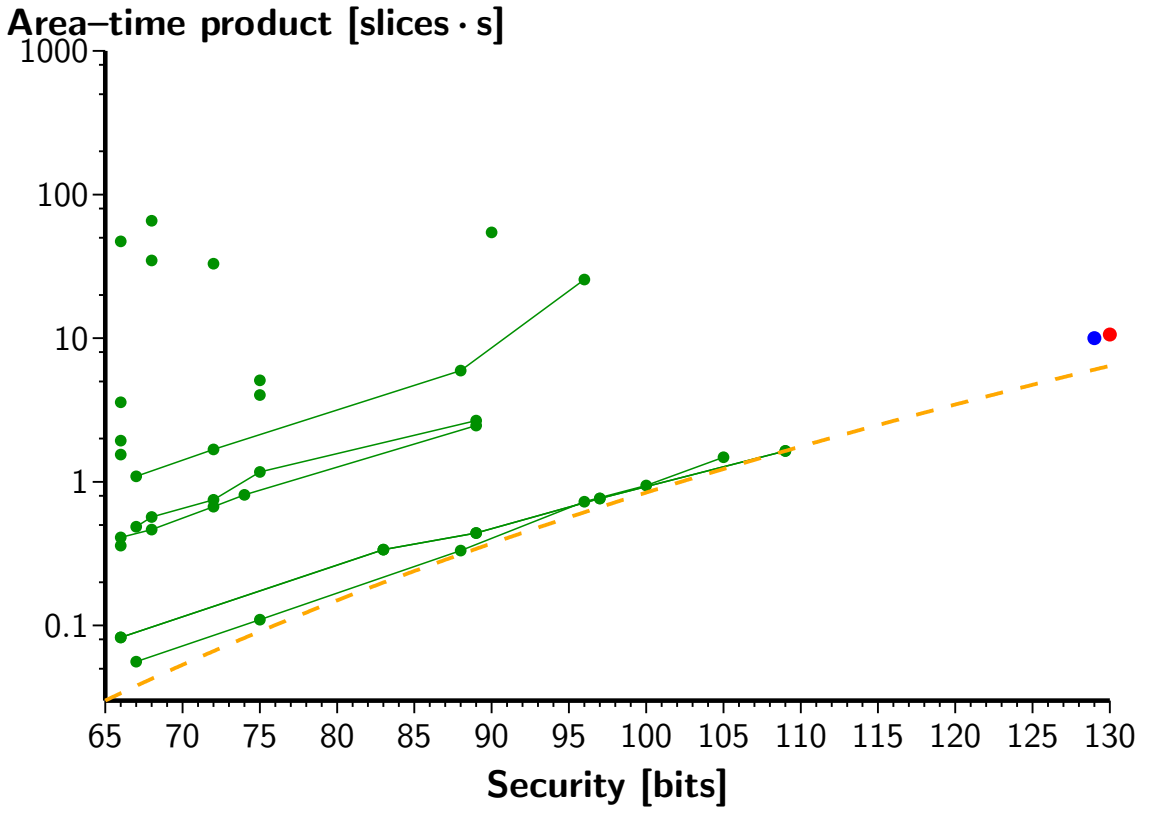
Area–Time product



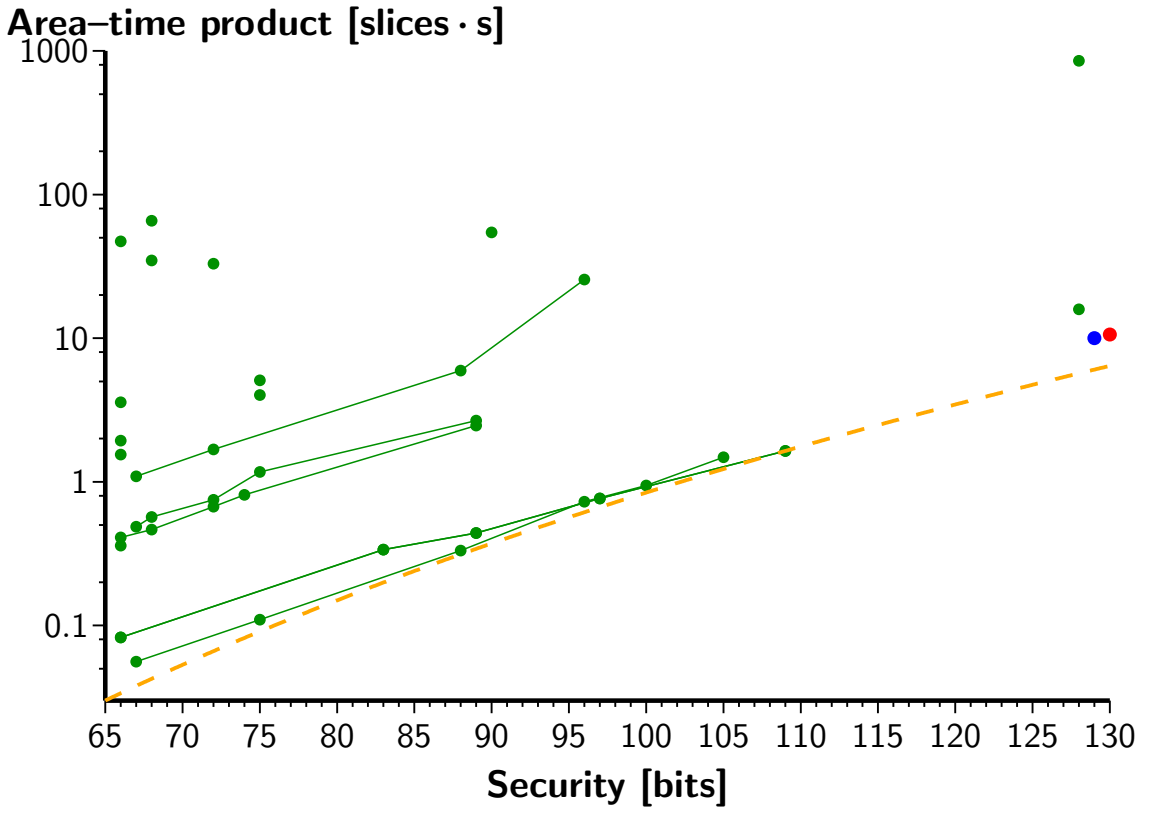
Area–Time product



Area–Time product



Area–Time product



Comparison with ASIC and software

- Our results (Xilinx Virtex-4):
  - Characteristic 3: 2.11 ms on 4755 slices
  - Characteristic 2: 4.47 ms on 2313 slices
- ASIC implementation of pairings over BN-curves with 128 bits of security:
  - 2.91 ms using 113 kGates on a 130 nm ASIC (Fan *et al.*, 2009)
  - 15.8 ms using 97 kGates on a 130 nm ASIC (Kammler *et al.*, 2009)
- Software implementation of pairings at the 128-bit security level
  - over supersingular curves (Beuchat *et al.*, 2009)
    - ⋆ Characteristic 2: 11.9 ms on one core of a 2.4 GHz Intel Core2
    - ⋆ Characteristic 3: 7.59 ms on one core of a 2.4 GHz Intel Core2
  - over BN-curve
    - ⋆ 0.921 ms on one core of a 2.4 GHz Intel Core2 (Aranha *et al.*, 2010)

Conclusion

- Compact, yet reasonably fast, accelerator for pairings with 128 bits of security
  - supersingular elliptic curve
  - low characteristic
  - efficient representation of the field of definition
  - careful security analysis

Conclusion

- Compact, yet reasonably fast, accelerator for pairings with 128 bits of security
  - supersingular elliptic curve
  - low characteristic
  - efficient representation of the field of definition
  - careful security analysis
- Characteristic 2 vs. Characteristic 3
  - field arithmetic efficiency vs. Better embedding degree

  → very close overall performance

Conclusion

- Compact, yet reasonably fast, accelerator for pairings with 128 bits of security
  - supersingular elliptic curve
  - low characteristic
  - efficient representation of the field of definition
  - careful security analysis
- Characteristic 2 vs. Characteristic 3
  - field arithmetic efficiency vs. Better embedding degree

  → very close overall performance
- Genus-1 over field of composite extension degree vs. Genus-2
  - both a way to compute in a smaller base field

  → close overall performance

Perspective and future work

- Implement this pairing on more curves:
  - better understanding of the software/hardware frontier
  - hopefully improve performance
  - try higher security level

Perspective and future work

- Implement this pairing on more curves:
  - better understanding of the software/hardware frontier
  - hopefully improve performance
  - try higher security level
- Exploit Trace-Zero Variety structure:
  - some parallelism in Miller's algorithm (Cesena, 2008)

Perspective and future work

- Implement this pairing on more curves:
  - better understanding of the software/hardware frontier
  - hopefully improve performance
  - try higher security level
- Exploit Trace-Zero Variety structure:
  - some parallelism in Miller's algorithm (Cesena, 2008)
- Pairings over genus-2 supersingular hyperelliptic curve
  - characteristic 2
  - better embedding degree ($k/g = 6$) but still not optimal for 128 bits of security
  - Optimal Eta pairing (ePrint Report 2010/559)
  - change the system of point representation (Work in Progress!)

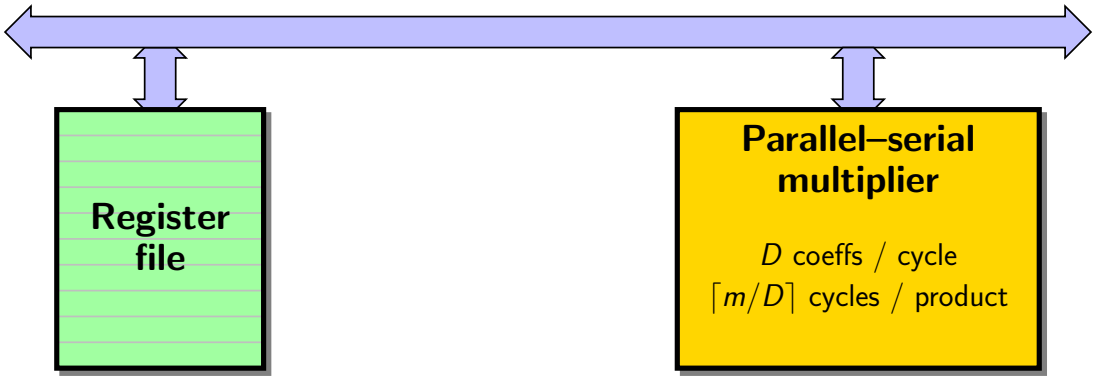
Questions?

A finite field coprocessor

- General-purpose finite-field arithmetic processor

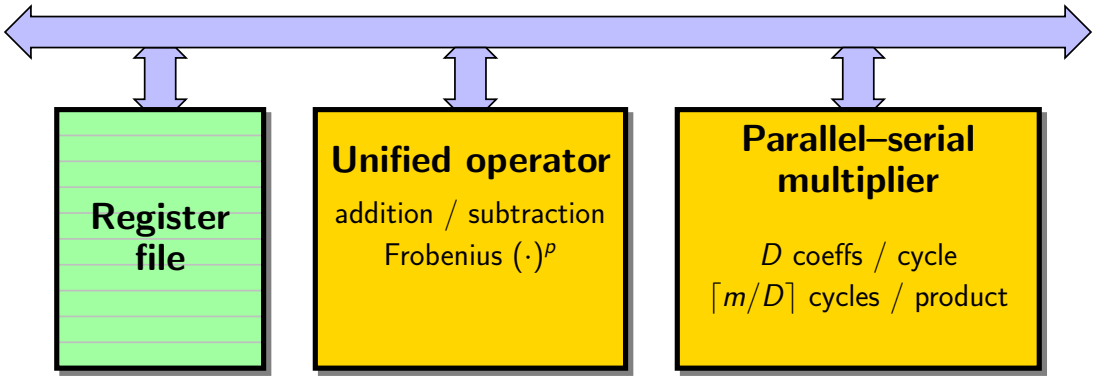
A finite field coprocessor

- General-purpose finite-field arithmetic processor



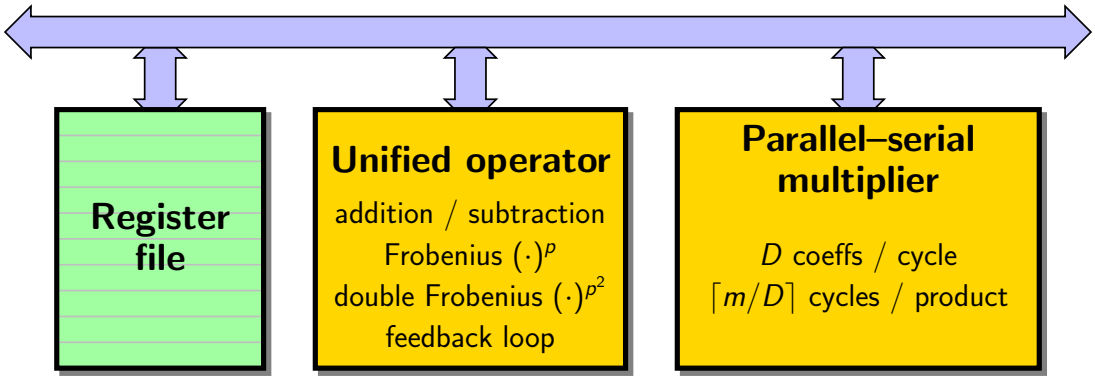
A finite field coprocessor

- General-purpose finite-field arithmetic processor

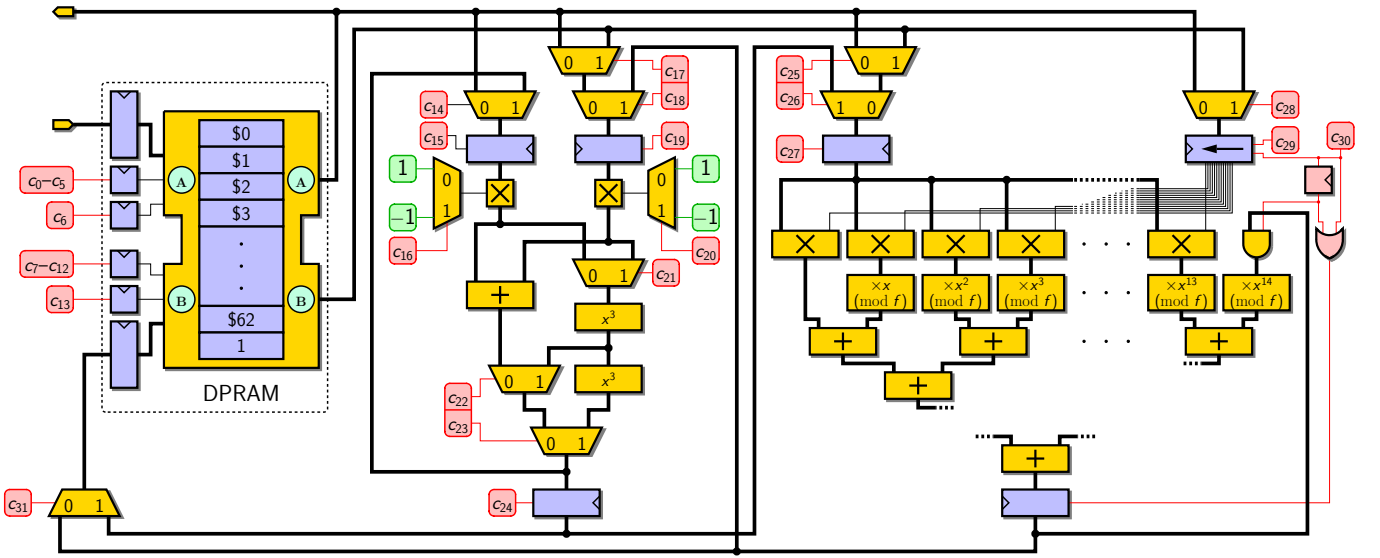


A finite field coprocessor

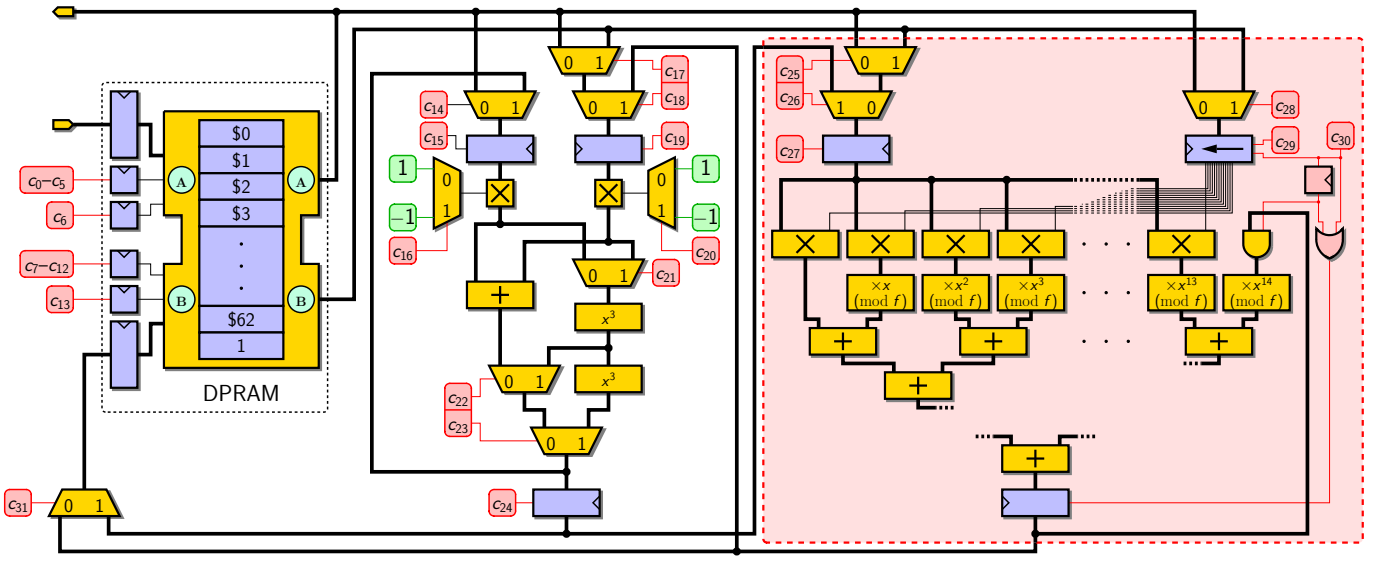
- General-purpose finite-field arithmetic processor



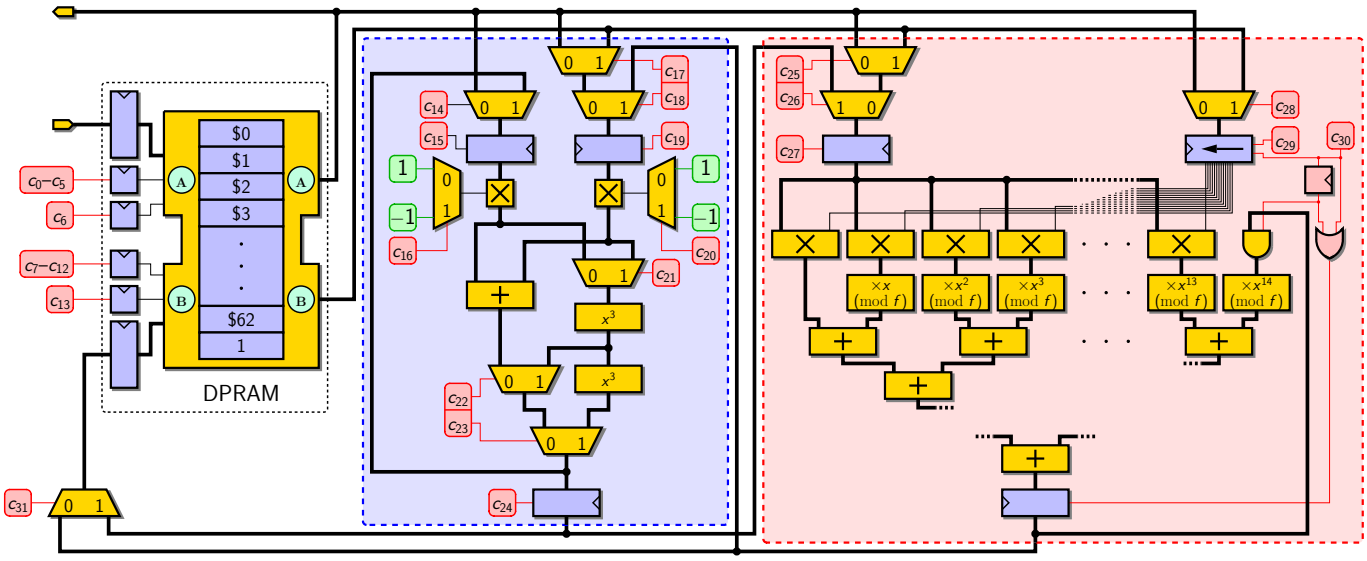
Detailed architecture of the coprocessor



Detailed architecture of the coprocessor



Detailed architecture of the coprocessor



Detailed architecture of the coprocessor