

Journées C2 — April 7, 2011

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

Nicolas Estibals

CAMEL project-team, LORIA, Nancy Université / CNRS / INRIA
Nicolas.Estibals@loria.fr

```
/*
 * CAMEL
 *
 * C.A.
 * M.P.
 * L.E.
 * L.E.
 *
 * 0101000
 * 0101111
 * 01110010
 * 01000010
 * 01000010
 * 01100010
 * 01011011
 * 01101101
 * 01110101
 * 01100101
 * 01000011
 * 0110000111
 * 0100000111
 * 0100000111
 *
 * 1-01100101010
 * 1-001001111011
 * 1-011001111011
 * 1-011001111011
 * 1-011001111011
 * 1-011001111011
 * 1-011001111011
 * 1-011001111011
 * 1-011001111011
 *
 * 0101010000
 * 0110100000
 * 0110100000
 * 0110100000
 * 0110100000
 * 0110100000
 * 0110100000
 * 0110100000
 * 0110100000
 *
 * 0101000000111
 * 0100000000111
 * 0100000000111
 * 0100000000111
 * 0100000000111
 * 0100000000111
 * 0100000000111
 * 0100000000111
 * 0100000000111
 *
 * 0101001111
 * 0100001111
 *
 */
```



LORIA
Laboratoire lorrain de recherche
en informatique et ses applications

Nancy-Université



centre de recherche NANCY - GRAND-EST

Introduction

- ▶ Pairings and cryptology
 - introduced as an **attack** against some elliptic curves
 - used as a **primitive** in many protocols and devices
 - ★ **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

Introduction

- ▶ Pairings and cryptology
 - introduced as an **attack** against some elliptic curves
 - used as a **primitive** in many protocols and devices
 - ★ **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 targets (e.g. smart card)

Introduction

- ▶ Pairings and cryptology
 - introduced as an **attack** against some elliptic curves
 - used as a **primitive** in many protocols and devices
 - ★ **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 targets (e.g. smart card)
- ▶ **Previous work** on FPGA implementations
 - **low-security** pairings
 - most are **performance-oriented** designs
- ▶ Our goal:
 - **AES-128** equivalent security
 - **compact** accelerator

Outline of the talk

- ▶ Context about pairings
- ▶ Pairings-friendly curves with 128 bits of security
- ▶ Implementation & performance results

Pairing is a bilinear map

- ▶ $\mathbb{G}_1 = \langle P \rangle$, $\mathbb{G}_2 = \langle Q \rangle$: additively-written cyclic group of prime order
 $\#\mathbb{G}_1 = \#\mathbb{G}_2 = \ell$
- ▶ \mathbb{G}_T : multiplicatively-written cyclic groups of order $\#\mathbb{G}_T = \#\mathbb{G}_1 = \#\mathbb{G}_2 = \ell$

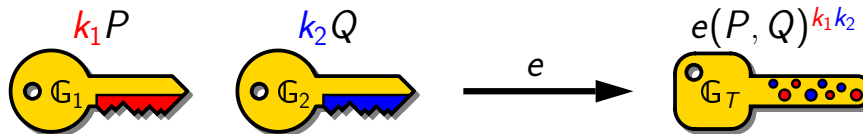
Pairing is a bilinear map

- ▶ $\mathbb{G}_1 = \langle P \rangle$, $\mathbb{G}_2 = \langle Q \rangle$: additively-written cyclic group of prime order
 $\#\mathbb{G}_1 = \#\mathbb{G}_2 = \ell$
- ▶ \mathbb{G}_T : multiplicatively-written cyclic groups of order $\#\mathbb{G}_T = \#\mathbb{G}_1 = \#\mathbb{G}_2 = \ell$
- ▶ $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear pairing iff:
 - non-degeneracy: $e(P, Q) \neq 1_{\mathbb{G}_T}$
 - bilinearity:
 - ★ $e(P_1 + P_2, Q') = e(P_1, Q') \cdot e(P_2, Q')$
 - ★ $e(P', Q_1 + Q_2) = e(P', Q_1) \cdot e(P', Q_2)$
 - computability: e can be efficiently computed

Pairing is a bilinear map

- ▶ $\mathbb{G}_1 = \langle P \rangle$, $\mathbb{G}_2 = \langle Q \rangle$: additively-written cyclic group of prime order
 $\#\mathbb{G}_1 = \#\mathbb{G}_2 = \ell$
- ▶ \mathbb{G}_T : multiplicatively-written cyclic groups of order $\#\mathbb{G}_T = \#\mathbb{G}_1 = \#\mathbb{G}_2 = \ell$
- ▶ $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is a bilinear pairing iff:
 - non-degeneracy: $e(P, Q) \neq 1_{\mathbb{G}_T}$
 - bilinearity:
 - ★ $e(P_1 + P_2, Q') = e(P_1, Q') \cdot e(P_2, Q')$
 - ★ $e(P', Q_1 + Q_2) = e(P', Q_1) \cdot e(P', Q_2)$
 - computability: e can be efficiently computed
- ▶ Important property for cryptographic applications:

$$e(k_1 P, k_2 Q) = e(k_2 P, k_1 Q) = e(P, Q)^{k_1 k_2}$$



Combining secrets without having to reveal them!

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 : $|\#E(\mathbb{F}_q) - (q + 1)| \leq 2\sqrt{q}$
 - 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:

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

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 : $|\#E(\mathbb{F}_q) - (q + 1)| \leq 2\sqrt{q}$
 - 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:

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

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 : $|\#E(\mathbb{F}_q) - (q + 1)| \leq 2\sqrt{q}$
 - 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:

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

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 : $|\#E(\mathbb{F}_q) - (q + 1)| \leq 2\sqrt{q}$
 - 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:
$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \rightarrow \mu_\ell \subset \mathbb{F}_{q^k}^*$$
- ▶ Computed thanks to Miller's iterative algorithm

General attacks

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

- ▶ 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

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

- ▶ 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 $\mathbb{F}_{q^k}^*$
 - FFS or NFS $\rightarrow L_{q^k}[1/3, c]$
 - complexity subexponential in q^k

General attacks

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

- ▶ 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 $\mathbb{F}_{q^k}^*$
 - FFS or NFS $\rightarrow L_{q^k}[1/3, c]$
 - complexity subexponential in q^k
- ▶ k acts as a cursor to balance the complexity of the two attacks
- ▶ $k = 12$: optimal for the 128-bit security level

Outline of the talk

- ▶ Context about pairings
- ▶ Pairings-friendly curves with 128 bits of security
- ▶ Implementation & performance results

Supersingular elliptic curves Vs. Barreto–Naehrig curves

► Definitions:

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

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

► Supersingular curve

⇒ Simpler curve arithmetic (efficient tripling formulae)

► Definition:

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

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

► Ordinary curve

Supersingular elliptic curves Vs. Barreto–Naehrig curves

► Definitions:

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

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

► Supersingular curve

⇒ Simpler **curve arithmetic** (efficient tripling formulae)

► Distortion map, modified pairing:

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

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

⇒ **Symmetric pairing** (BN cannot be used with all protocols)

► Definition:

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

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

► Ordinary curve

► No distortion map

Supersingular elliptic curves Vs. Barreto–Naehrig curves

► Definitions:

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

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

► Supersingular curve

⇒ Simpler **curve arithmetic** (efficient tripling formulae)

► Distortion map, modified pairing:

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

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

⇒ **Symmetric pairing** (BN cannot be used with all protocols)

► Small characteristic field arithmetic

⇒ **No carry**, better suited to **hardware** implementation

► Definition:

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

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

► Ordinary curve

► No distortion map

► Modular arithmetic

Supersingular elliptic curves Vs. Barreto–Naehrig curves

▶ Definitions:

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

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

▶ Supersingular curve

⇒ Simpler **curve arithmetic** (efficient tripling formulae)

▶ Distortion map, modified pairing:

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

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

⇒ **Symmetric pairing** (BN cannot be used with all protocols)

▶ Small characteristic field arithmetic

⇒ **No carry**, better suited to **hardware** implementation

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

⇒ **Larger field** of definition for the same security level. For 128 bits of security:

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

▶ Definition:

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

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

▶ Ordinary curve

▶ No distortion map

▶ Modular arithmetic

▶ **Optimal** embedding degree ($k = 12$)

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

Supersingular elliptic curves

► Definitions:

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

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

► Supersingular curve

⇒ Simpler **curve arithmetic** (efficient tripling formulae)

► Distortion map, modified pairing:

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

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

⇒ **Symmetric pairing**

► Small characteristic field arithmetic

⇒ **No carry**, better suited to **hardware** implementation

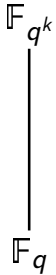
► Small embedding degree ($k = 6$ or 4)

⇒ **Larger field** of definition for the same security level.

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

Which field of definition?

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



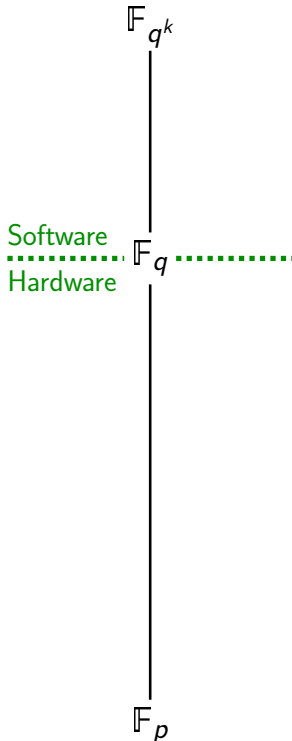
► Arithmetic of \mathbb{F}_{q^k} over \mathbb{F}_q :

- tower field fixed by pairing construction
- already optimized by previous works
- Critical operation: products in \mathbb{F}_q

Which field of definition?

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

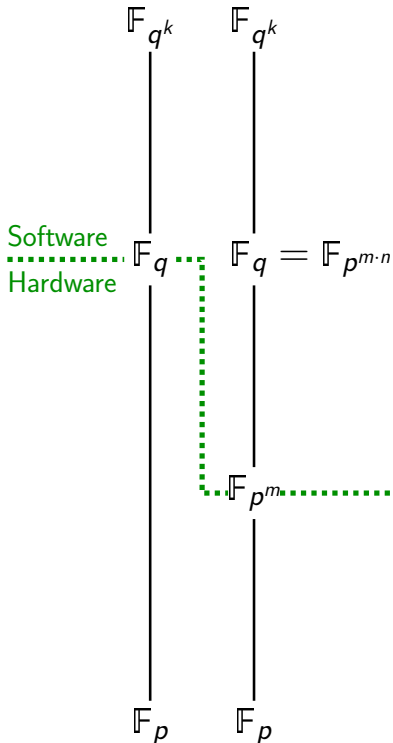
- ▶ Arithmetic of \mathbb{F}_{q^k} over \mathbb{F}_q :
 - tower field **fixed by pairing construction**
 - already optimized by **previous works**
 - **Critical** operation: **products** in \mathbb{F}_q
- ▶ Arithmetic of \mathbb{F}_q
 - traditionally implemented in **hardware**
 - **does not scale** to the 128-bit security level



Which field of definition?

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

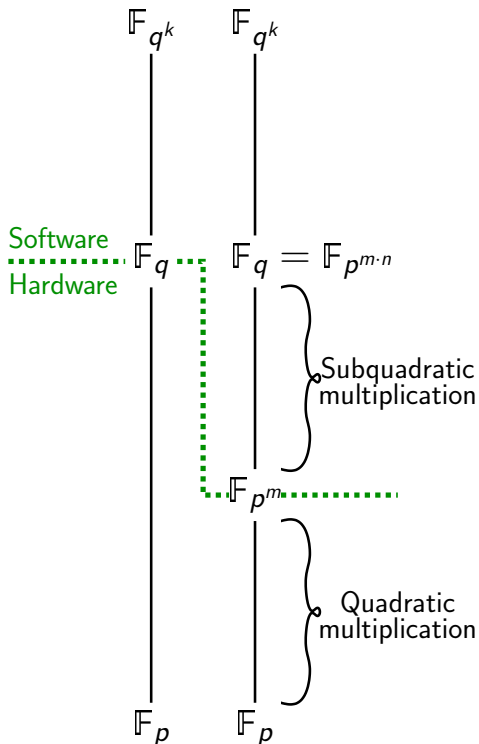
- ▶ Arithmetic of \mathbb{F}_{q^k} over \mathbb{F}_q :
 - tower field fixed by pairing construction
 - already optimized by previous works
 - Critical operation: products in \mathbb{F}_q
- ▶ Arithmetic of \mathbb{F}_q
 - traditionally implemented in hardware
 - does not scale to the 128-bit security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



Which field of definition?

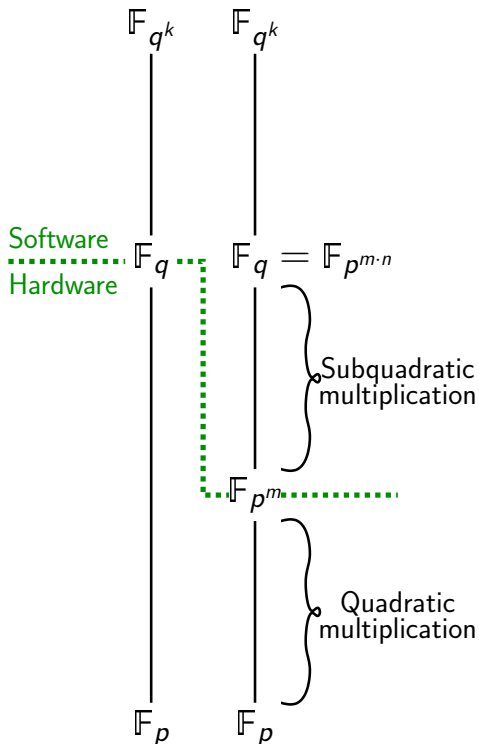
$$e : E(\mathbb{F}_q)[\ell] \times E(\mathbb{F}_{q^k})[\ell] \rightarrow \mu_\ell \subset \mathbb{F}_{q^k}^*$$

- ▶ Arithmetic of \mathbb{F}_{q^k} over \mathbb{F}_q :
 - tower field fixed by pairing construction
 - already optimized by previous works
 - Critical operation: products in \mathbb{F}_q
- ▶ Arithmetic of \mathbb{F}_q
 - traditionally implemented in hardware
 - does not scale to the 128-bit security level
- ▶ 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?

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



- ▶ Arithmetic of \mathbb{F}_{q^k} over \mathbb{F}_q :
 - tower field fixed by pairing construction
 - already optimized by previous works
 - Critical operation: products in \mathbb{F}_q
- ▶ Arithmetic of \mathbb{F}_q
 - traditionally implemented in hardware
 - does not scale to the 128-bit security level
- ▶ 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 additional attacks

Weil Descent-based attacks

- ▶ 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:

$$E(\mathbb{F}_{p^{m \cdot n}}) \cong W_E(\mathbb{F}_{p^m})$$

Weil Descent-based attacks

- ▶ 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:

$$E(\mathbb{F}_{p^{m \cdot n}}) \cong W_E(\mathbb{F}_{p^m})$$

- ▶ Gaudry–Hess–Smart attack:

- $W_E(\mathbb{F}_{p^m})$ might map to $\text{Jac}(\mathcal{C})$, 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-based attacks

- ▶ 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:

$$E(\mathbb{F}_{p^{m \cdot n}}) \cong W_E(\mathbb{F}_{p^m})$$

- ▶ Gaudry–Hess–Smart attack:

- $W_E(\mathbb{F}_{p^m})$ might map to $\text{Jac}(\mathcal{C})$, with \mathcal{C} a curve of genus at least n
- index calculus algorithm: solve DLP in $\tilde{O}((p^m)^{2-\frac{2}{n}})$

- ▶ Static Diffie–Hellman problem

- leakage when reusing private key (e.g. ElGamal encryption)
- Granger’s attack: complexity in $\tilde{O}((p^m)^{1-\frac{1}{n+1}})$
- revoke key after a certain amount of use is an effective workaround

Suitable curves for 128-bit security level

p^m	n	$\log_2 \ell$	Cost of the attacks (bits)			
			Pollard's ρ	FFS		
3^{503}	1	697	342	132		
3^{97}	5	338	163	130		
3^{67}	7	612	300	129		
3^{53}	11	672	330	140		
3^{43}	13	764	376	138		

Suitable curves for 128-bit security level

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

Suitable curves for 128-bit security level

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

Suitable curves for 128-bit security level

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

Suitable curves for 128-bit security level

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

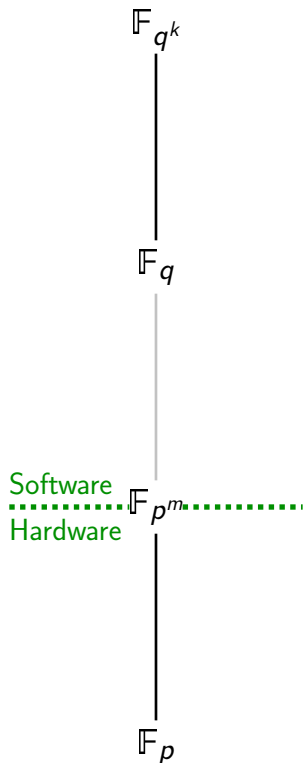
Outline of the talk

- ▶ Context about pairings
- ▶ Pairings-friendly curves with 128 bits of security
- ▶ **Implementation & performance results**

Arithmetic of the extension field

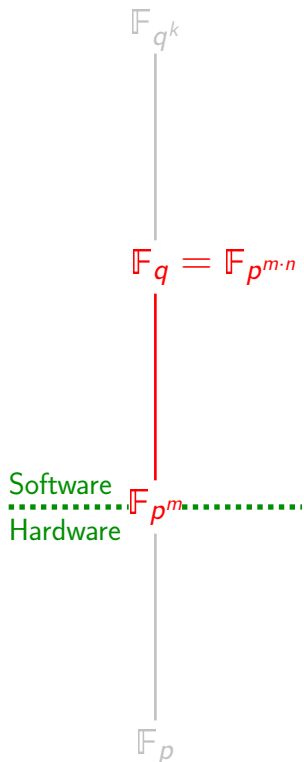
► We already have:

- pairing algorithm expressed as operations in \mathbb{F}_q
- finite field coprocessor for \mathbb{F}_{p^m}

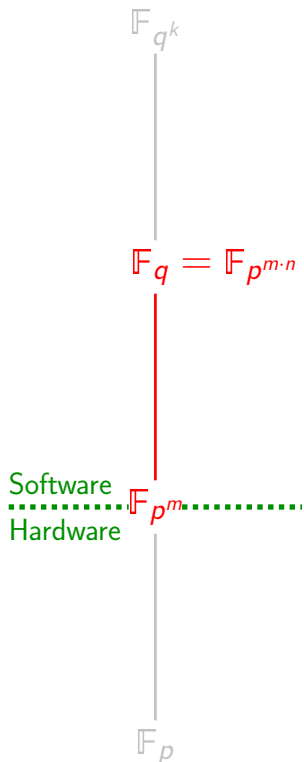


Arithmetic of the extension field

- ▶ We already have:
 - pairing algorithm expressed as operations in \mathbb{F}_q
 - finite field coprocessor for \mathbb{F}_{p^m}
- ▶ **Polynomial representation:** $\mathbb{F}_{p^{m \cdot n}} \cong \mathbb{F}_{p^m}[X]/(f(X))$
 - f irreducible polynomial of degree n
 - addition, Frobenius automorphism (squaring or cubing):
easy to compute

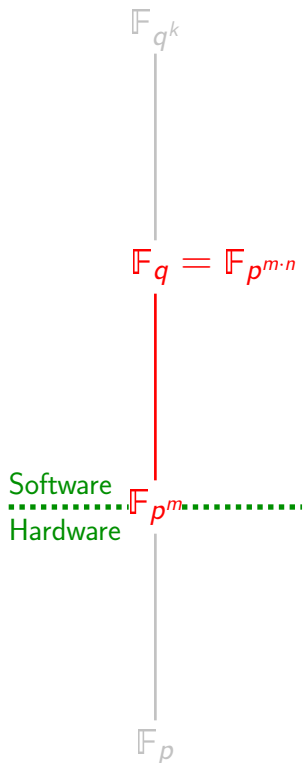


Arithmetic of the extension field



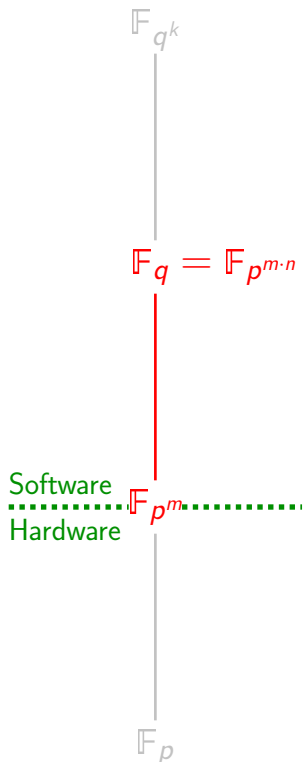
- ▶ We already have:
 - pairing algorithm expressed as operations in \mathbb{F}_q
 - finite field coprocessor for \mathbb{F}_{p^m}
- ▶ **Polynomial representation:** $\mathbb{F}_{p^{m \cdot n}} \cong \mathbb{F}_{p^m}[X]/(f(X))$
 - f irreducible polynomial of degree n
 - addition, Frobenius automorphism (squaring or cubing):
easy to compute
- ▶ **Multiplication :** Bottleneck of pairing computation
- ▶ Our test cases:
 - multiplication in $\mathbb{F}_{3^{97 \cdot 5}}$
 - ★ using "schoolbook" algorithm
 - ★ 25 products in $\mathbb{F}_{3^{97}}$
 - ★ 24 additions in $\mathbb{F}_{3^{97}}$
 - multiplication in $\mathbb{F}_{2^{163 \cdot 7}}$
 - ★ using "schoolbook" algorithm
 - ★ 49 products in $\mathbb{F}_{2^{163}}$
 - ★ 48 additions in $\mathbb{F}_{2^{163}}$

Arithmetic of the extension field



- ▶ We already have:
 - pairing algorithm expressed as operations in \mathbb{F}_q
 - finite field coprocessor for \mathbb{F}_{p^m}
- ▶ **Polynomial representation:** $\mathbb{F}_{p^{m \cdot n}} \cong \mathbb{F}_{p^m}[X]/(f(X))$
 - f irreducible polynomial of degree n
 - addition, Frobenius automorphism (squaring or cubing):
easy to compute
- ▶ **Multiplication :** Bottleneck of pairing computation
- ▶ Our test cases:
 - multiplication in $\mathbb{F}_{3^{97 \cdot 5}}$
 - ★ using CRT-based algorithm
 - ★ 12 products in $\mathbb{F}_{3^{97}}$
 - ★ 53 additions in $\mathbb{F}_{3^{97}}$
 - multiplication in $\mathbb{F}_{2^{163 \cdot 7}}$
 - ★ using "schoolbook" algorithm
 - ★ 49 products in $\mathbb{F}_{2^{163}}$
 - ★ 48 additions in $\mathbb{F}_{2^{163}}$

Arithmetic of the extension field



- ▶ We already have:
 - pairing algorithm expressed as operations in \mathbb{F}_q
 - finite field coprocessor for \mathbb{F}_{p^m}
- ▶ **Polynomial representation:** $\mathbb{F}_{p^{m.n}} \cong \mathbb{F}_{p^m}[X]/(f(X))$
 - f irreducible polynomial of degree n
 - addition, Frobenius automorphism (squaring or cubing):
easy to compute
- ▶ **Multiplication :** Bottleneck of pairing computation
- ▶ Our test cases:
 - multiplication in $\mathbb{F}_{397.5}$
 - ★ using CRT-based algorithm
 - ★ 12 products in \mathbb{F}_{397}
 - ★ 53 additions in \mathbb{F}_{397}
 - multiplication in $\mathbb{F}_{2^{163}.7}$
 - ★ using Montgomery's formulae
 - ★ 22 products in $\mathbb{F}_{2^{163}}$
 - ★ 84 additions in $\mathbb{F}_{2^{163}}$

Experimental setup

► Count of operations

- Full pairing computation over $E(\mathbb{F}_{397.5})$

	×	+	(.) ³
\mathbb{F}_{397}	37289	253314	21099

- Full pairing computation over $E(\mathbb{F}_{2^{163}.7})$

	×	+	(.) ²
$\mathbb{F}_{2^{163}}$	88509	448361	52782

Experimental setup

▶ Count of operations

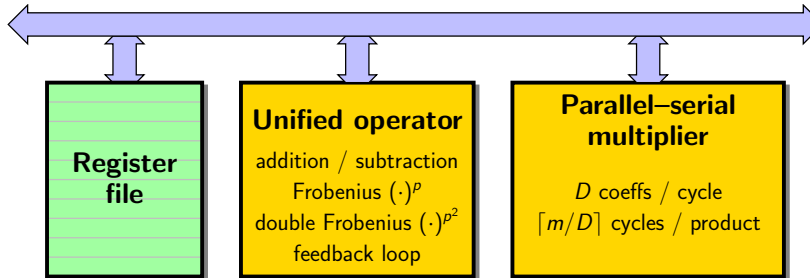
- Full pairing computation over $E(\mathbb{F}_{397.5})$

	×	+	(.) ³
\mathbb{F}_{397}	37289	253314	21099

- Full pairing computation over $E(\mathbb{F}_{2163.7})$

	×	+	(.) ²
\mathbb{F}_{2163}	88509	448361	52782

▶ Finite field coprocessors



- Prototyped on [Xilinx Virtex-4 LX FPGAs](#)
- Post-place-and-route [timing](#) and [area](#) estimations

Experimental setup

▶ Count of operations

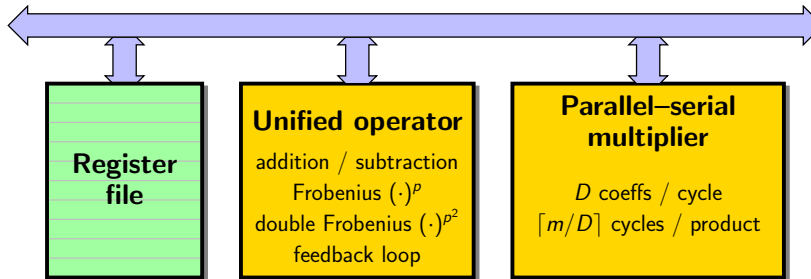
- Full pairing computation over $E(\mathbb{F}_{397.5})$

	×	+	(.) ³
\mathbb{F}_{397}	37289	253314	21099

- Full pairing computation over $E(\mathbb{F}_{2^{163.7}})$

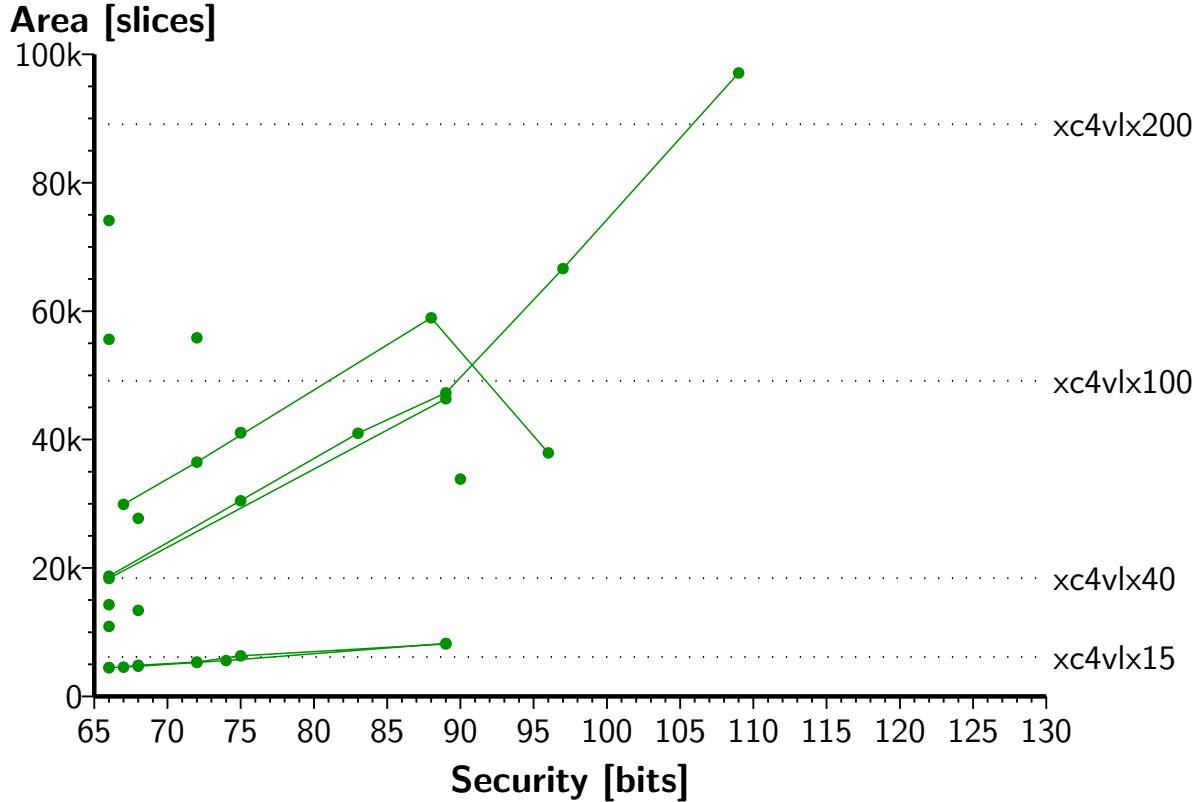
	×	+	(.) ²
$\mathbb{F}_{2^{163}}$	88509	448361	52782

▶ Finite field coprocessors

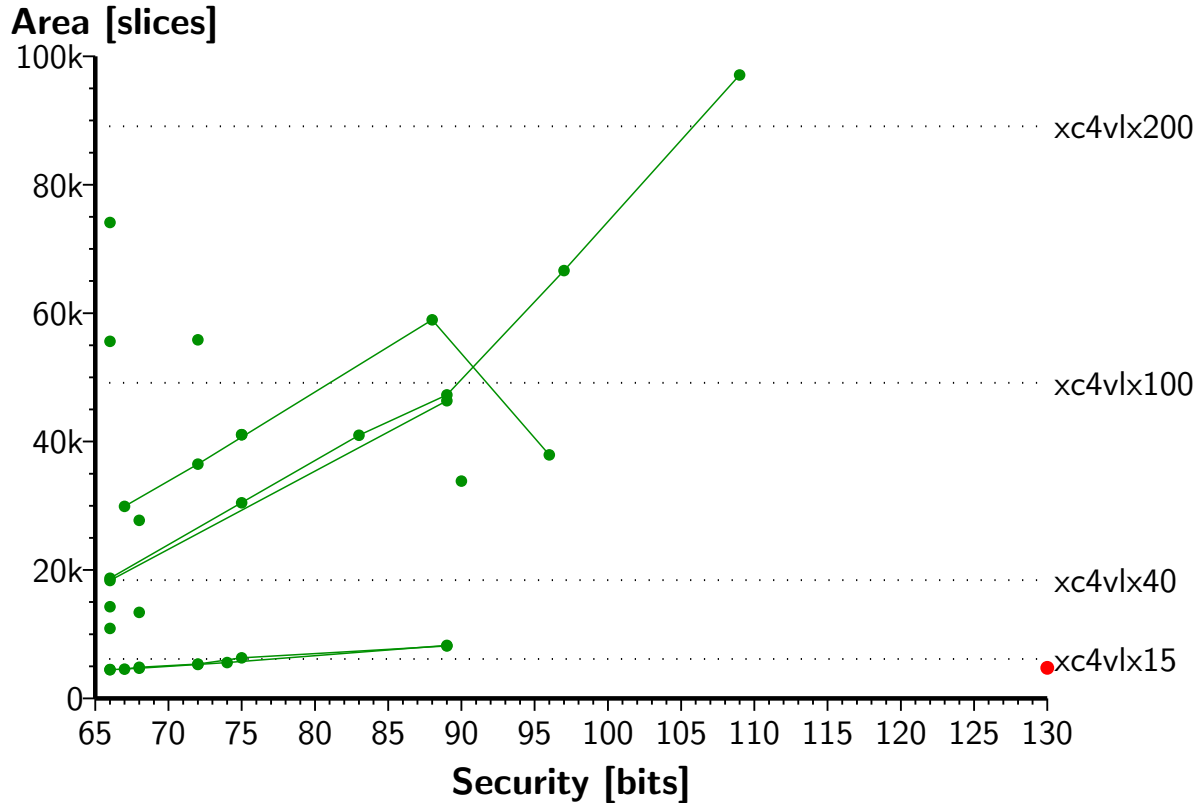


- Prototyped on [Xilinx Virtex-4 LX FPGAs](#)
- Post-place-and-route [timing](#) and [area](#) estimations
- ▶ [Scheduled](#) operations for full pairing computation
 - on $E(\mathbb{F}_{397.5})$, [428,854 cycles](#)
 - on $E(\mathbb{F}_{2^{163.7}})$, [1,147,131 cycles](#)

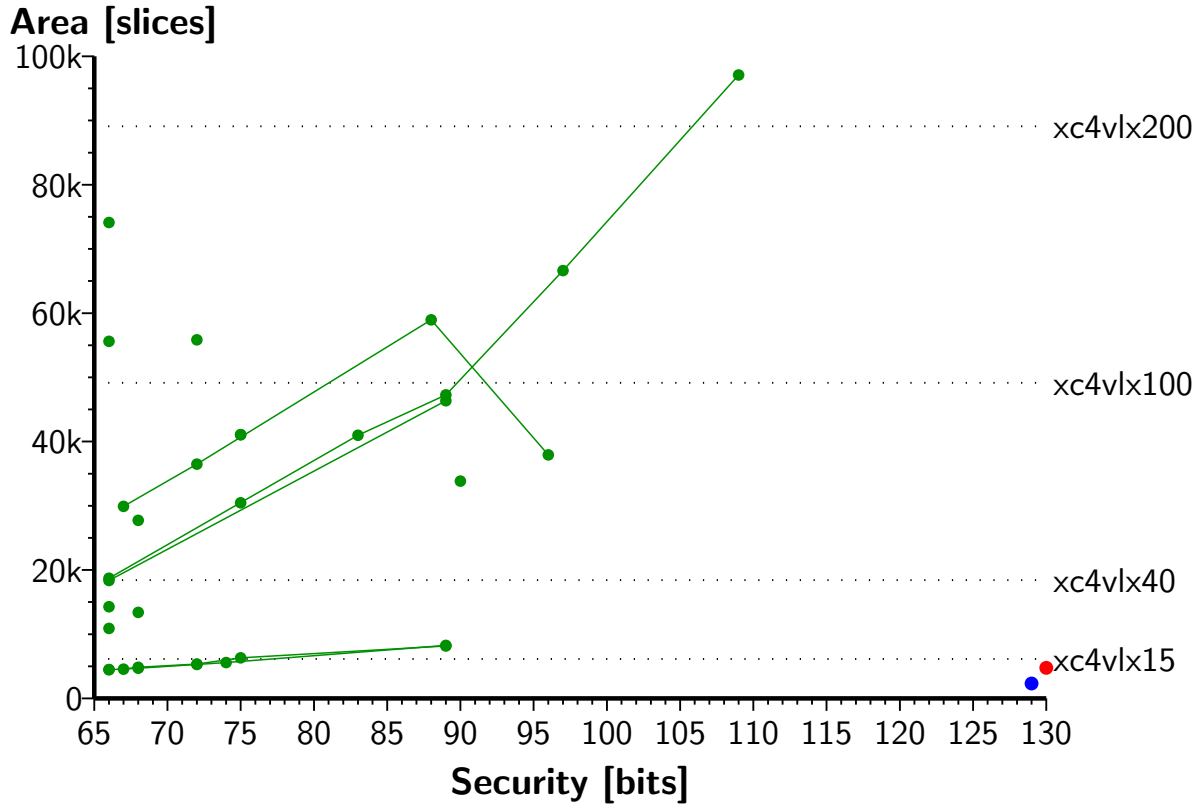
Area



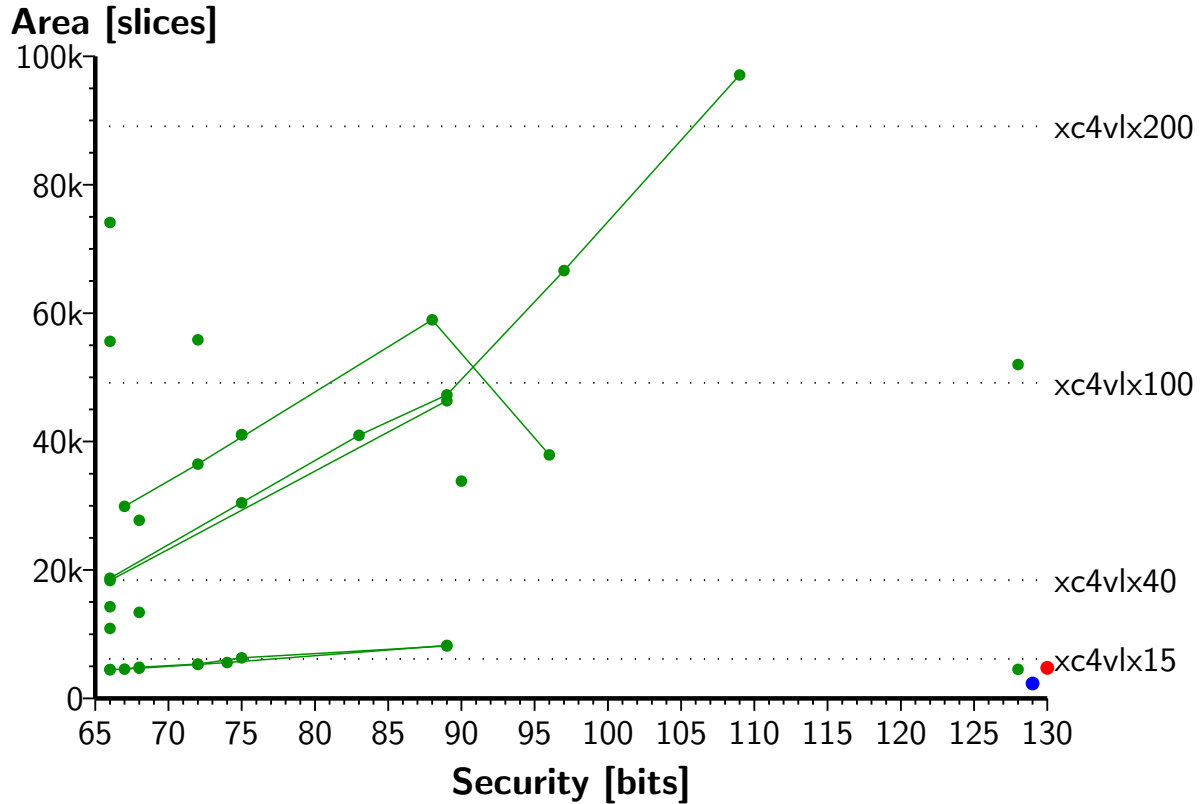
Area



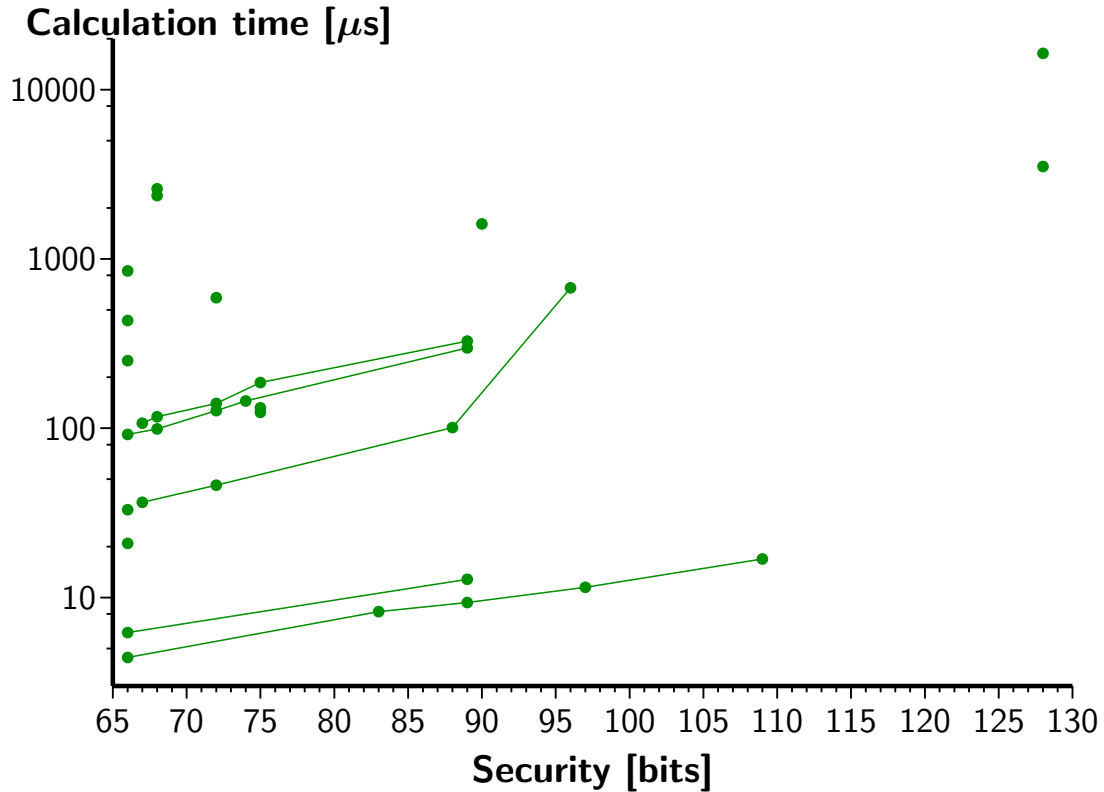
Area



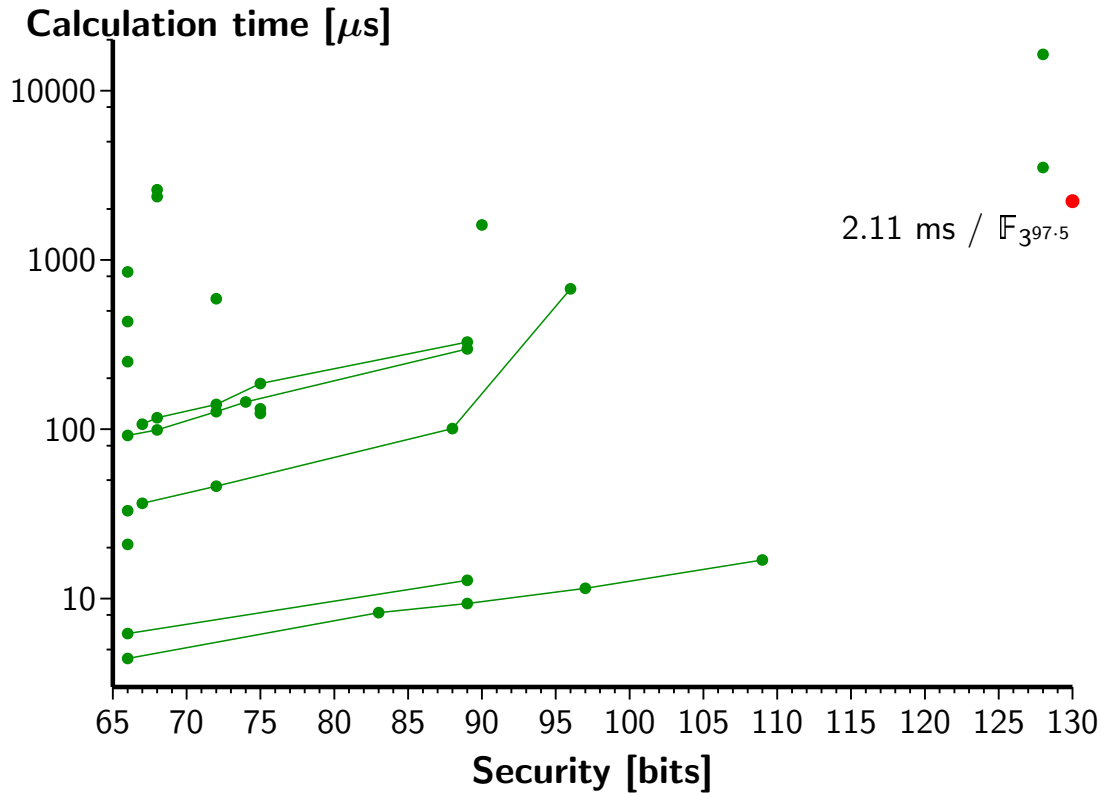
Area



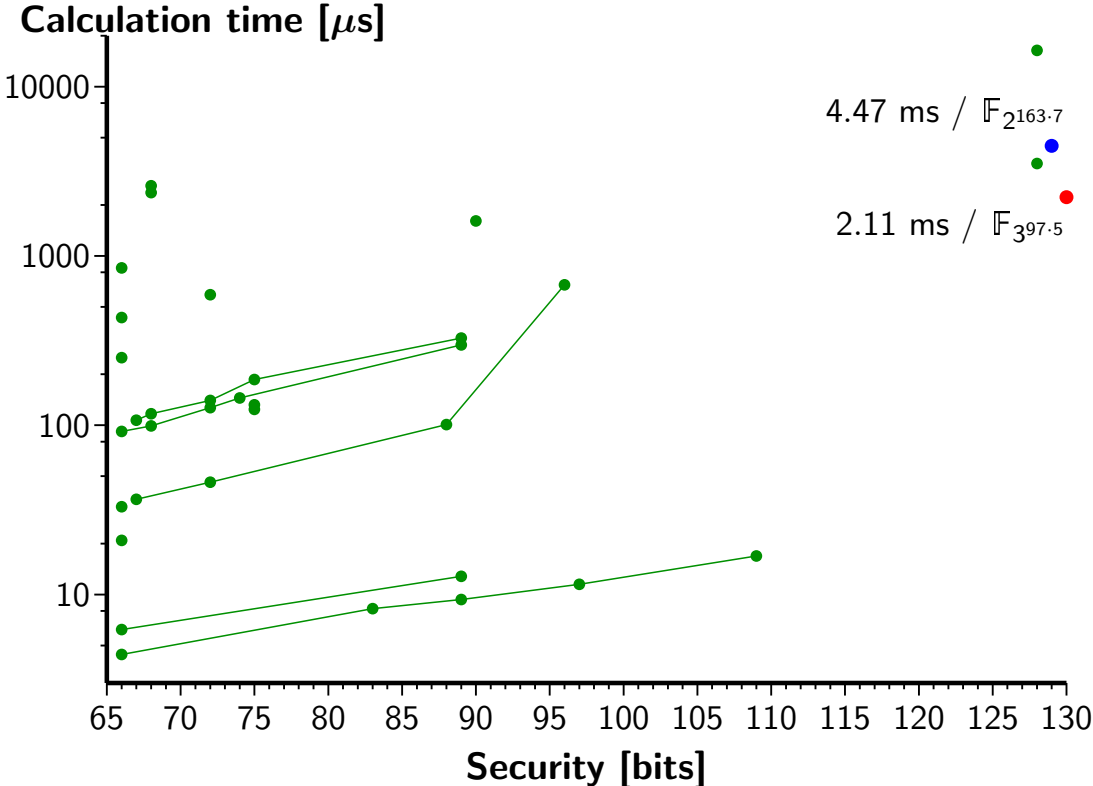
Calculation time



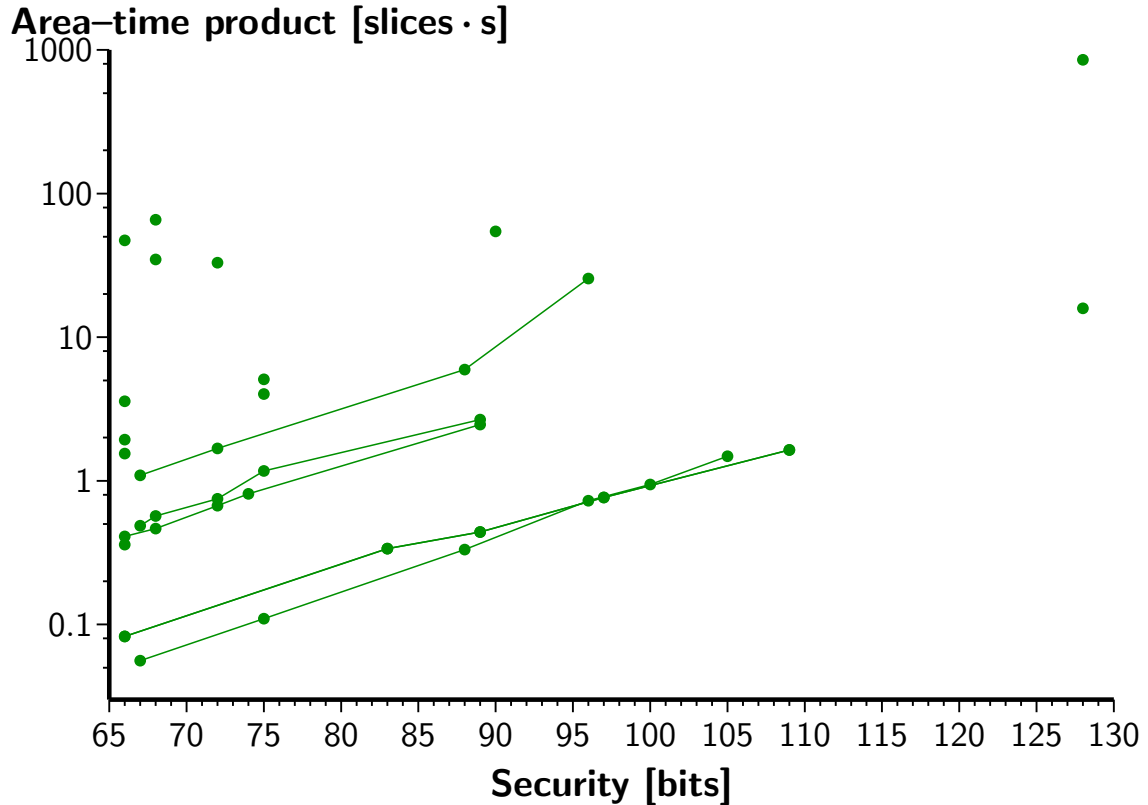
Calculation time



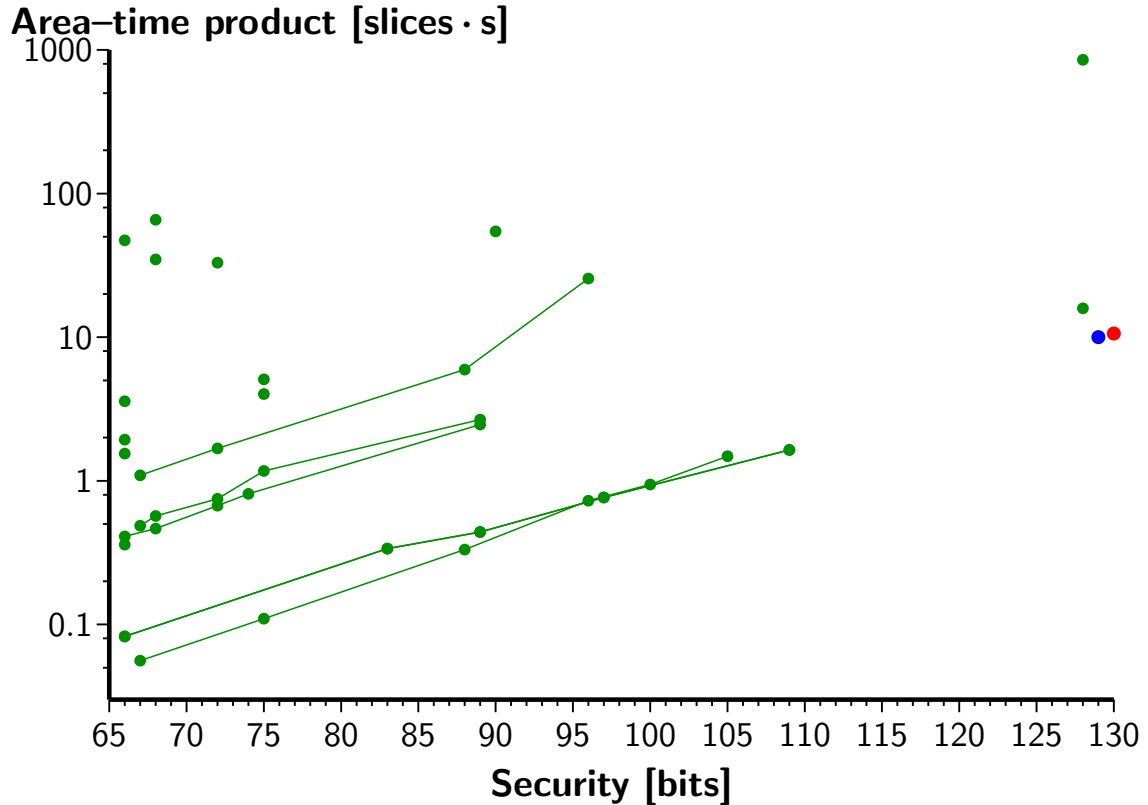
Calculation time



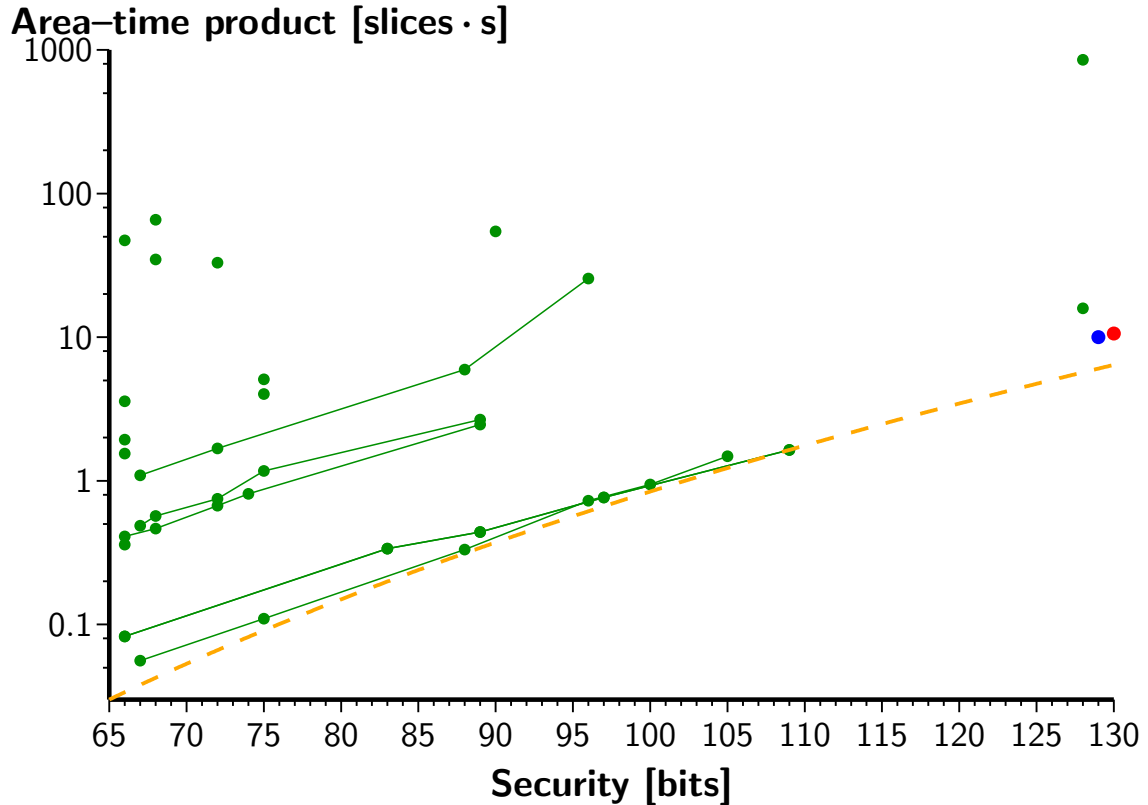
Area-Time product



Area-Time product



Area-Time product



Comparison with ASIC and software

	Supersingular curves	BN-curves
FPGA	2.11 ms (This Work)	52 ms (Ghosh <i>et al.</i> , 2010)
ASIC	–	2.91 ms (Fan <i>et al.</i> , 2009)
Software (2.4 GHz Intel Core2)	7.59 ms (Beuchat <i>et al.</i> , 2009)	0.92 ms (Aranha <i>et al.</i> , 2010)

Conclusion

- ▶ Compact, yet reasonably fast, accelerator for pairings with 128 bits of security
 - supersingular elliptic curve
 - low characteristic
 - take advantage of the sub-optimal k to implement efficient field arithmetic

Conclusion

- ▶ Compact, yet reasonably fast, accelerator for pairings with 128 bits of security
 - supersingular elliptic curve
 - low characteristic
 - take advantage of the sub-optimal k to implement efficient field arithmetic
 - mathematical choice/algorithm/hardware codesign

Conclusion

- ▶ Compact, yet reasonably fast, accelerator for pairings with 128 bits of security
 - supersingular elliptic curve
 - low characteristic
 - take advantage of the sub-optimal k to implement efficient field arithmetic
 - mathematical choice/algorithm/hardware codesign
- ▶ Implement this pairing on more curves:
 - better understanding of the software/hardware frontier
 - hopefully improve performance
 - try higher security level
 - study genus-2 supersingular curves

Thank you for you attention!



Questions?