PhD defense — October 30th, 2013

Algorithms and arithmetic for the implementation of cryptographic pairings

Nicolas Estibals

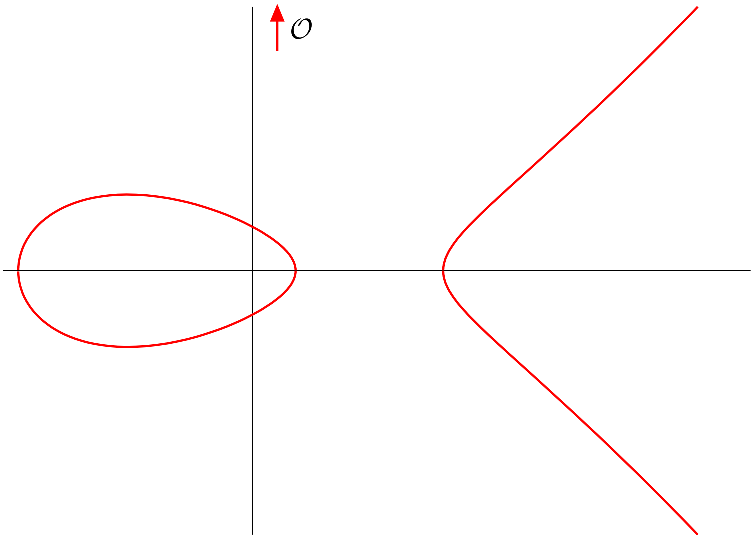
CARAMEL project-team, LORIA,

Université de Lorraine / CNRS / INRIA, France

Nicolas.Estibals@loria.fr

What is an elliptic curve?

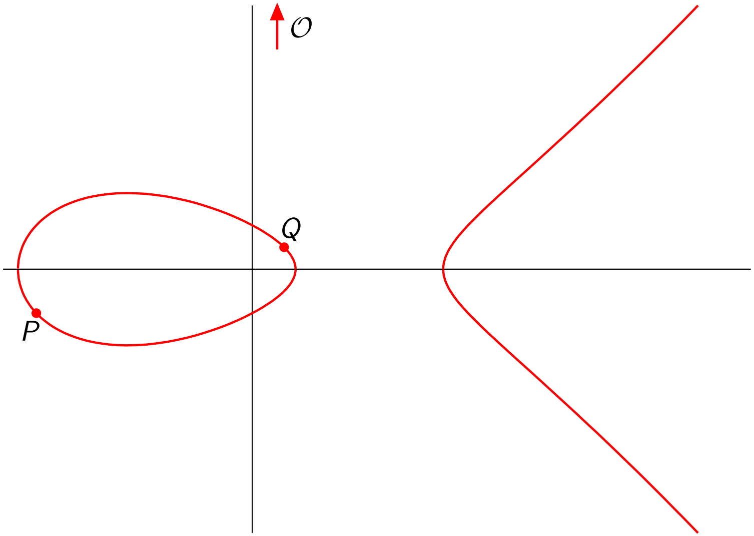
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group

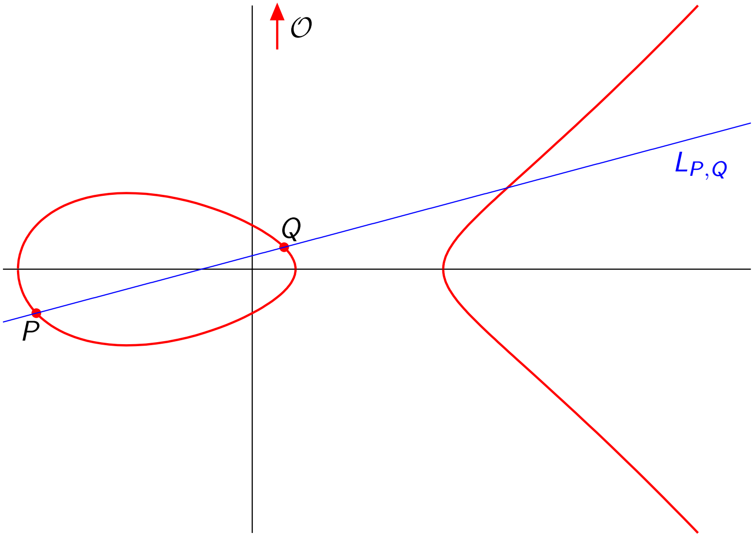
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group

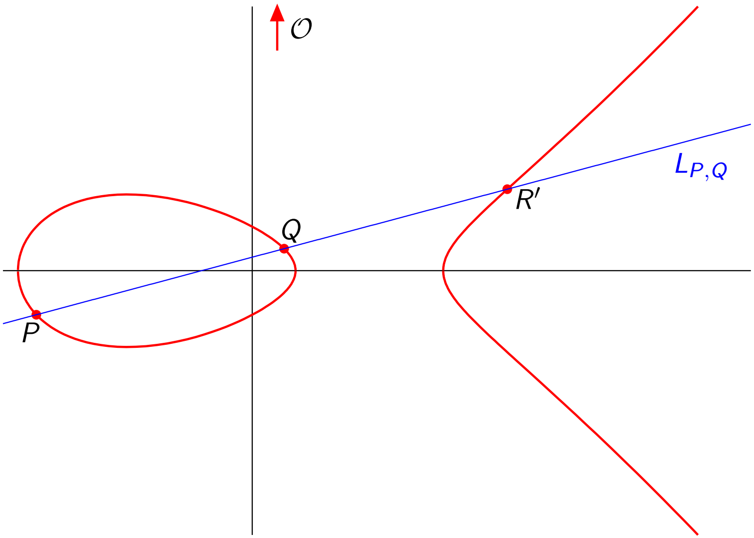
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group

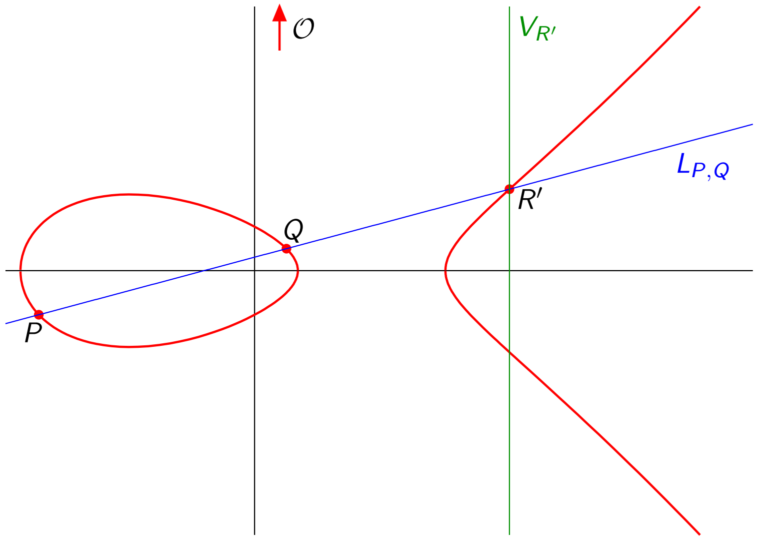
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group

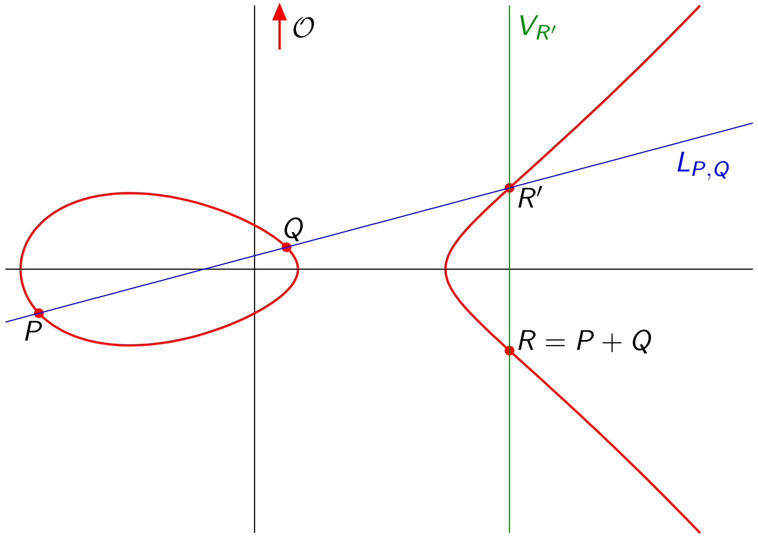
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group

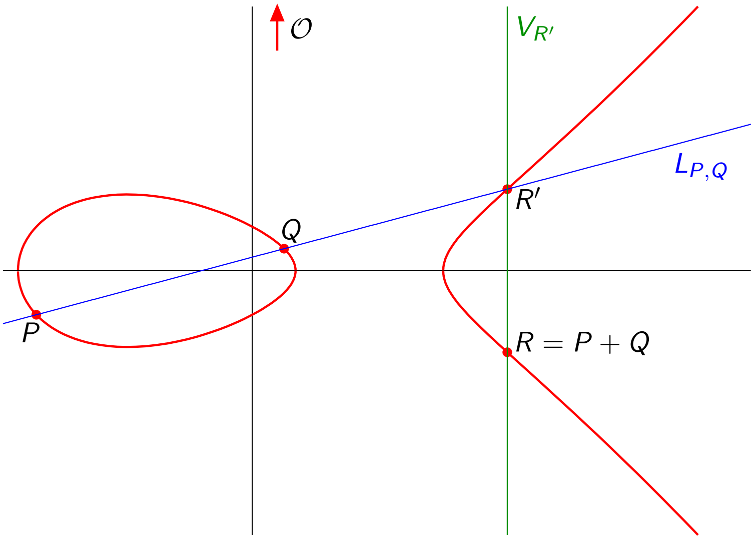
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group
- In practice: K is a finite field $\mathbb{F}_q$
- $E(\mathbb{F}_q)$ is a finite group

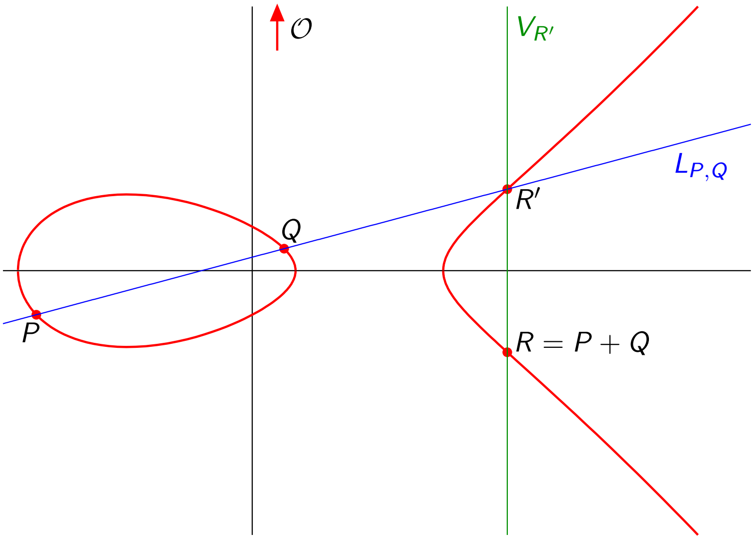
$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



What is an elliptic curve?

- Set of points $E(K)$ is a group
- In practice: K is a finite field $\mathbb{F}_q$
- $E(\mathbb{F}_q)$ is a finite group
- $[n]P = \underbrace{P + \cdots + P}_{n \text{ times}}$

$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$

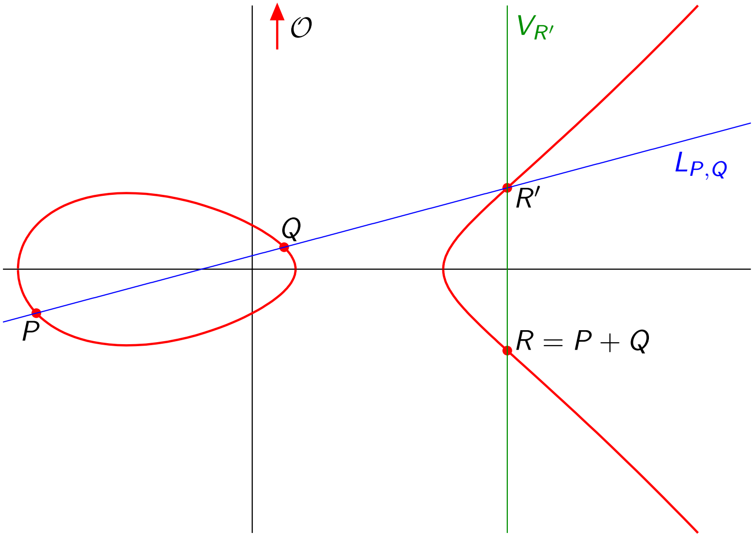


What is an elliptic curve?

- Set of points $E(K)$ is a group
- In practice: K is a finite field $\mathbb{F}_q$
- $E(\mathbb{F}_q)$ is a finite group
- $[n]P = \underbrace{P + \cdots + P}_{n \text{ times}}$
- ℓ: a large prime dividing $\#E(\mathbb{F}_q)$
- Use a cyclic subgroup of

$$E[\ell] = \{P \mid [\ell]P = \mathcal{O}\}$$

$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$

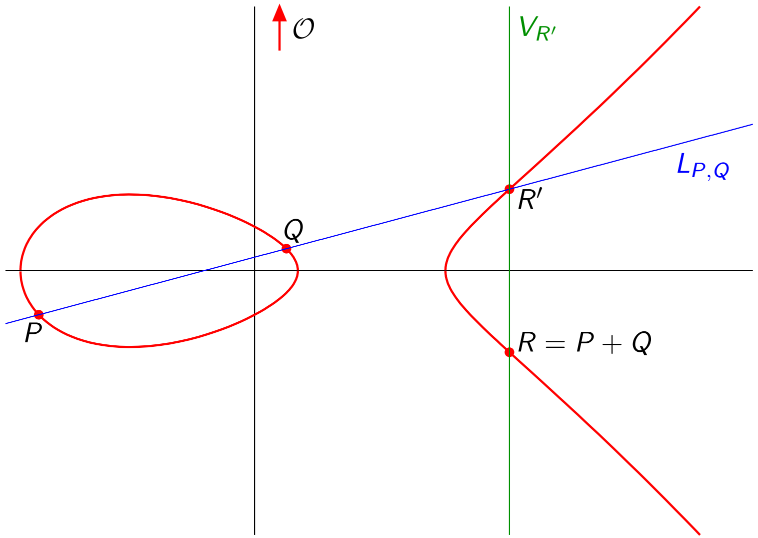


What is an elliptic curve?

- Set of points $E(K)$ is a group
- In practice: K is a finite field $\mathbb{F}_q$
- $E(\mathbb{F}_q)$ is a finite group
- $[n]P = \underbrace{P + \cdots + P}_{n \text{ times}}$
- ℓ: a large prime dividing $\#E(\mathbb{F}_q)$
- Use a cyclic subgroup of

$$E[\ell] = \{P \mid [\ell]P = \mathcal{O}\}$$

$$E/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 1$ and $\deg f = 3$



- Our favorite curves: $E_3 : y^2 = x^3 - x \pm 1$
  - characteristic 3
  - supersingular

Elliptic Curve Cryptography

Discrete Logarithm Problem (DLP)

Let $\mathbb{G}$ be a cyclic group, P a generator, given $Q \in \mathbb{G}$, it is supposed to be hard to compute a such that

$$Q = [a]P$$

Elliptic Curve Cryptography

Discrete Logarithm Problem (DLP)

Let $\mathbb{G}$ be a cyclic group, P a generator, given $Q \in \mathbb{G}$, it is supposed to be hard to compute a such that

$$Q = [a]P$$

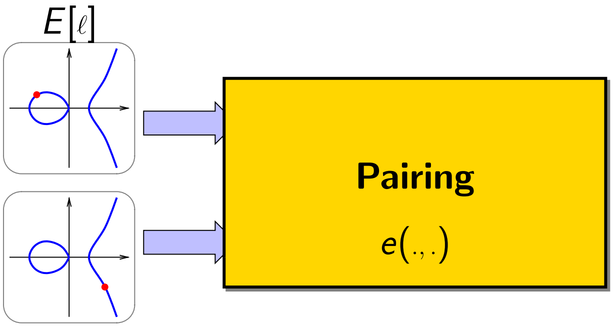
- Use this hard problem to design cryptographic protocols
- Diffie–Hellman key exchange:

  - Alice generates a secret integer a
  - Alice sends $[a]P$ to Bob
  - Alice computes $[a][b]P$

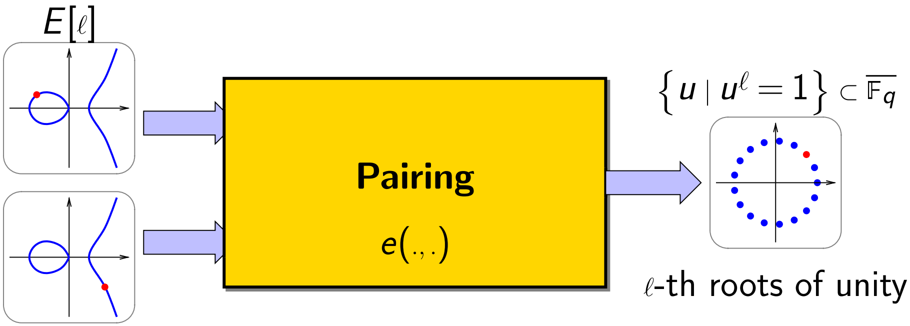
  - Bob generates a secret integer b
  - Bob sends $[b]P$ to Alice
  - Bob computes $[b][a]P$

They both share the same secret: $[ab]P$

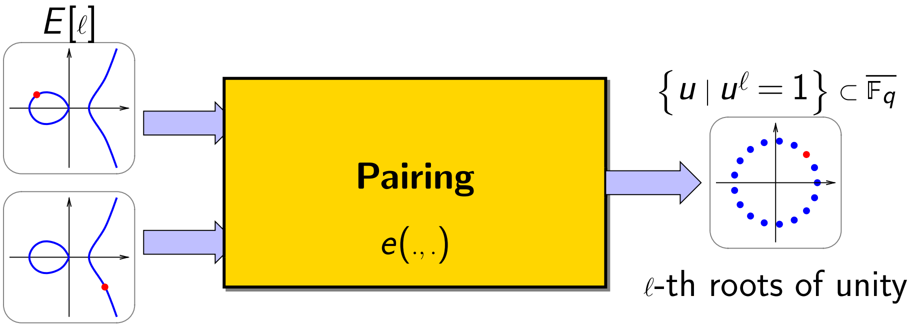
What is a pairing?



What is a pairing?



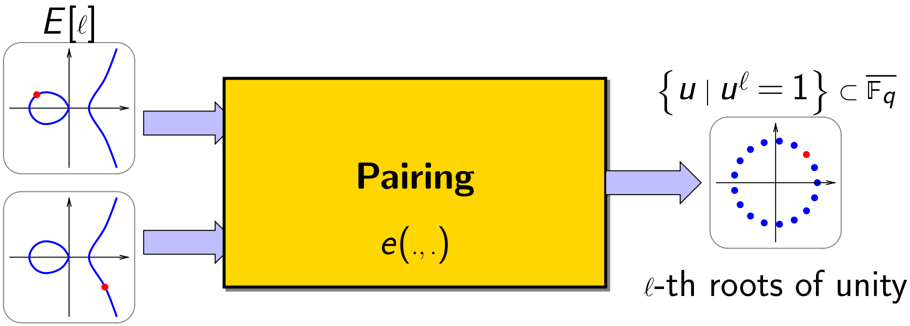
What is a pairing?



- Bilinear map:

$$e(P + P', Q) = e(P, Q) \cdot e(P', Q)$$
$$e(P, Q + Q') = e(P, Q) \cdot e(P, Q')$$

What is a pairing?



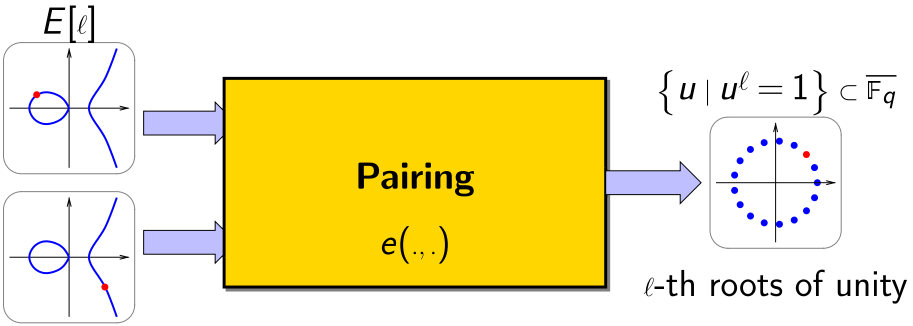
- Bilinear map:

$$e(P+P', Q) = e(P,Q) \cdot e(P', Q)$$
$$e(P, Q+Q') = e(P,Q) \cdot e(P, Q')$$

- Cryptographic interest: *Mixing two secrets without having to know them*

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

What is a pairing?



- Bilinear map:

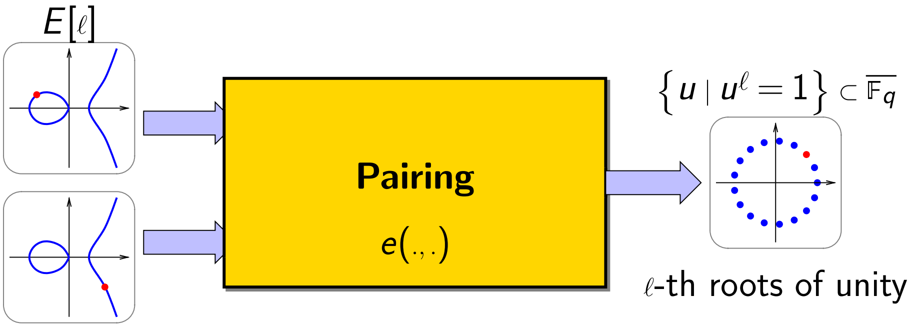
$$e(P+P', Q) = e(P,Q) \cdot e(P',Q)$$
$$e(P, Q+Q') = e(P,Q) \cdot e(P,Q')$$

- Cryptographic interest: *Mixing two secrets without having to know them*

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

- Useful for advanced protocols
  - short signature
  - electronic voting
  - electronic money
  - ...

What is a pairing?



- Bilinear map:

$$e(P + P', Q) = e(P, Q) \cdot e(P', Q)$$
$$e(P, Q + Q') = e(P, Q) \cdot e(P, Q')$$

- Cryptographic interest: *Mixing two secrets without having to know them*

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

- Useful for advanced protocols
  - short signature
  - electronic voting
  - electronic money
  - ...
- DLP should be hard on all the groups involved

Security considerations

- Security measurement
  - number of operations to break a cryptosystem
  - today's recommendation: 128-bit security

$$2^{128} \text{ operations}$$

Security considerations

- Security measurement
  - number of operations to break a cryptosystem
  - today's recommendation: 128-bit security

$$2^{128} \text{ operations}$$

- Difficulty of the DLP on the curve
  - depends on the order ℓ
  - roughly $\sqrt{\ell}$ operations

Security considerations

- Security measurement
  - number of operations to break a cryptosystem
  - today's recommendation: 128-bit security

$$2^{128} \text{ operations}$$

For our favorite curve
E_3 over $\mathbb{F}_{3^{509}}$

- Difficulty of the DLP on the curve
  - depends on the order ℓ — $\ell \approx 2^{697}$
  - roughly $\sqrt{\ell}$ operations — $\approx 2^{349}$ operations

Security considerations

- Security measurement
  - number of operations to break a cryptosystem
  - today's recommendation: 128-bit security

$$2^{128} \text{ operations}$$

- Difficulty of the DLP on the curve
  - depends on the order ℓ
  - roughly $\sqrt{\ell}$ operations
- Difficulty of the DLP on the roots of unity
  - embedding degree: k such that all roots lie in $\mathbb{F}_{q^k}$

For our favorite curve
E_3 over $\mathbb{F}_{3^{509}}$

$\ell \approx 2^{697}$

$\approx 2^{349}$ operations

$k = 6$, so DLP in $(\mathbb{F}_{3^{6\cdot 509}})^*$

Security considerations

For our favorite curve E_3 over $\mathbb{F}_{3^{509}}$

- Security measurement
  - number of operations to break a cryptosystem
  - today's recommendation: 128-bit security

$$2^{128} \text{ operations}$$

- Difficulty of the DLP on the curve
  - depends on the order ℓ — $\ell \approx 2^{697}$
  - roughly $\sqrt{\ell}$ operations — $\approx 2^{349}$ operations

- Difficulty of the DLP on the roots of unity
  - embedding degree: k such that all roots lie in $\mathbb{F}_{q^k}$ — $k = 6$, so DLP in $(\mathbb{F}_{3^{6 \cdot 509}})^*$
  - Subexponential algorithms exist
    - ⋆ function field sieve — $\approx 2^{132}$ operations

Security considerations

For our favorite curve
E_3 over $\mathbb{F}_{3^{509}}$

- Security measurement
  - number of operations to break a cryptosystem
  - today's recommendation: 128-bit security

2^{128} operations

- Difficulty of the DLP on the curve
  - depends on the order ℓ — $\ell \approx 2^{697}$
  - roughly $\sqrt{\ell}$ operations — $\approx 2^{349}$ operations
- Difficulty of the DLP on the roots of unity
  - embedding degree: k such that all roots lie in $\mathbb{F}_{q^k}$ — $k = 6$, so DLP in $(\mathbb{F}_{3^{6\cdot 509}})^*$
  - Subexponential algorithms exist
    - ⋆ function field sieve — $\approx 2^{132}$ operations
    - ⋆ very recent results (2013)

      Records by Joux and Göloğlu et al. records

      Joux

      Barbulescu, Gaudry, Joux, Thomé

      Adj, Menezes, Oliveira, Rodríguez-Henríquez — $\lessapprox 2^{75}$ operations

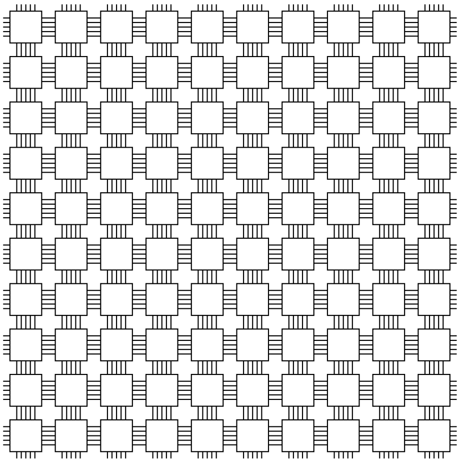
Why cryptography and hardware implementations?

- Growth of numeric exchanges
  - many applications
    - bank services
    - secure firmware updates
    - personal communications
    - ...
  - many targets
    - embedded electronics
    - smart cards
    - smartphones
    - computers, servers
- Security implies non-trivial computations
- Need for hardware implementations
  - CPUs may be inadequate
  - limited resources



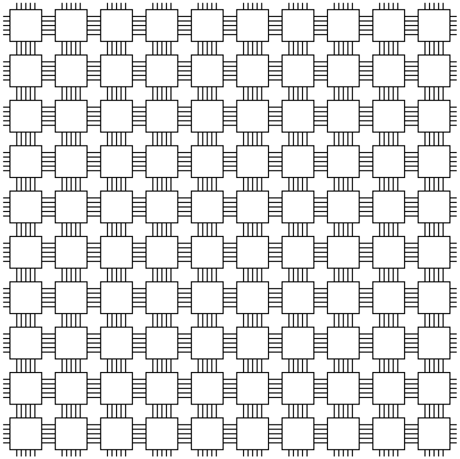


Hardware implementation



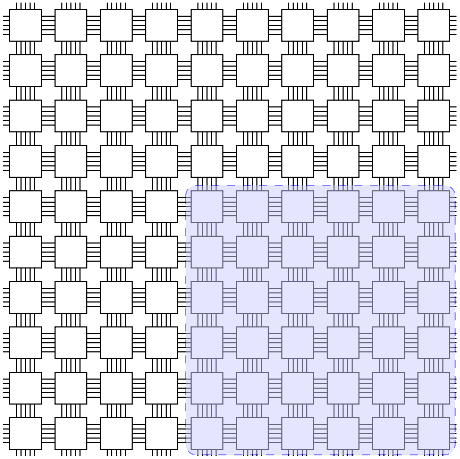
- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection

Hardware implementation



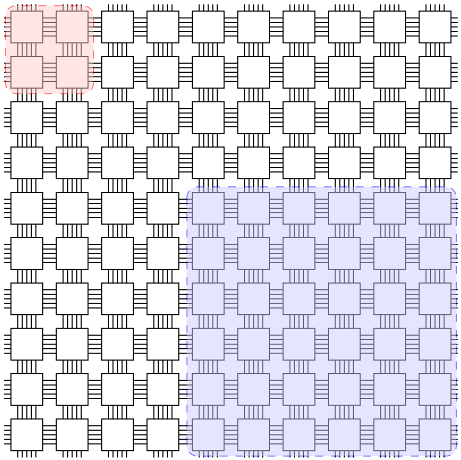
- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)

Hardware implementation

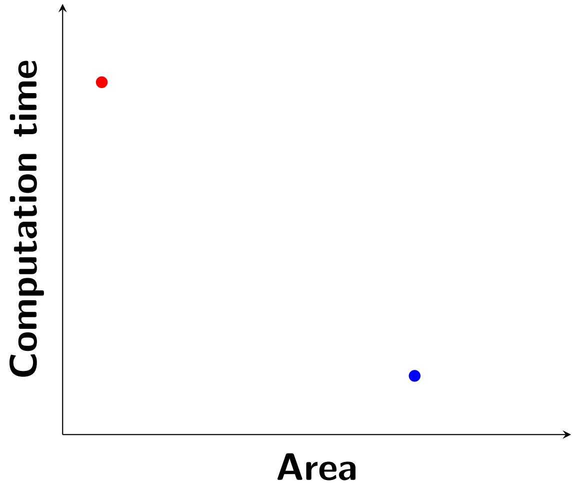


- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)
  - area (slices)

Hardware implementation

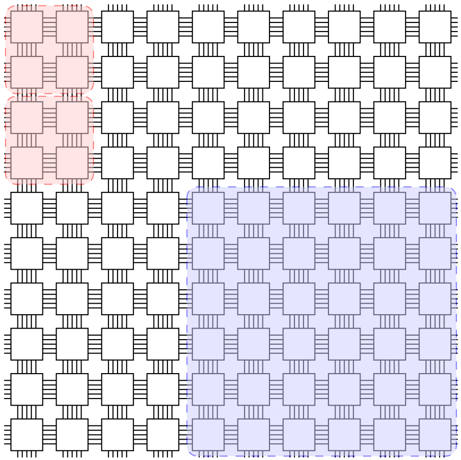


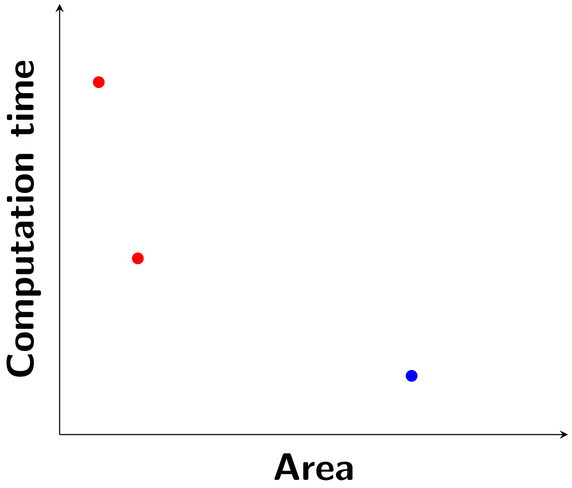
- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)
  - area (slices)
- Different designs for the same computation
  - optimized for latency
  - optimized for compactness



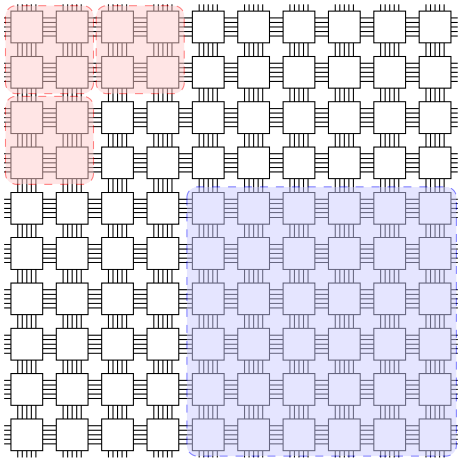
Hardware implementation

- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)
  - area (slices)
- Different designs for the same computation
  - optimized for latency
  - optimized for compactness

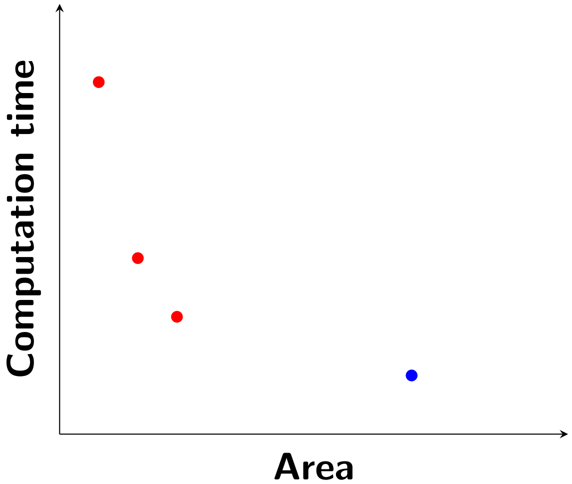




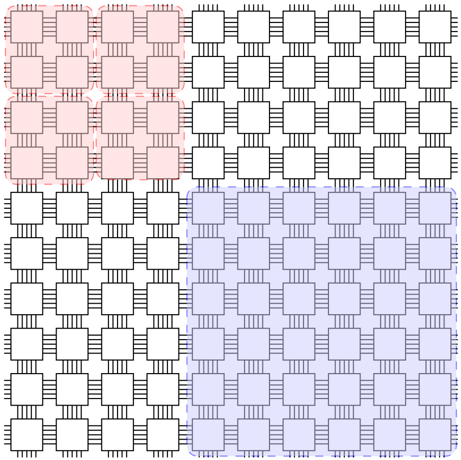
Hardware implementation



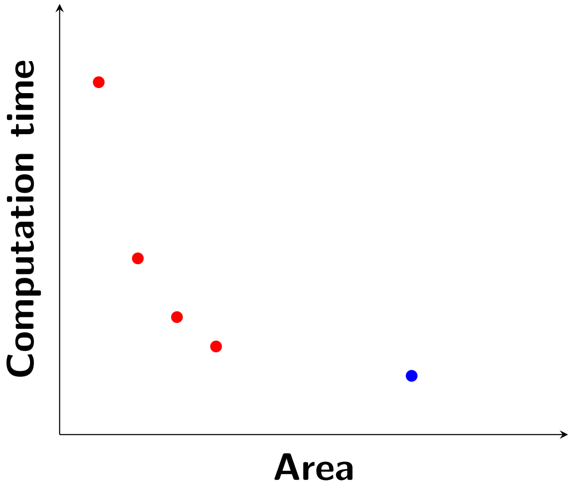
- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)
  - area (slices)
- Different designs for the same computation
  - optimized for latency
  - optimized for compactness



Hardware implementation

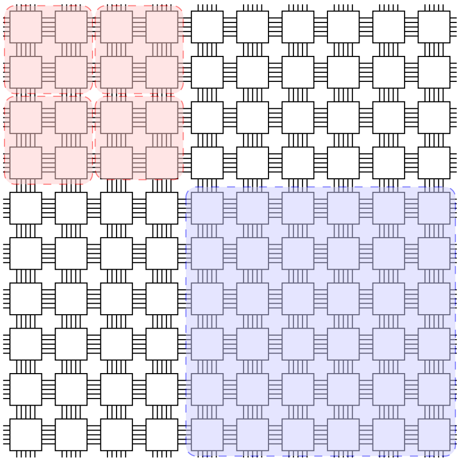


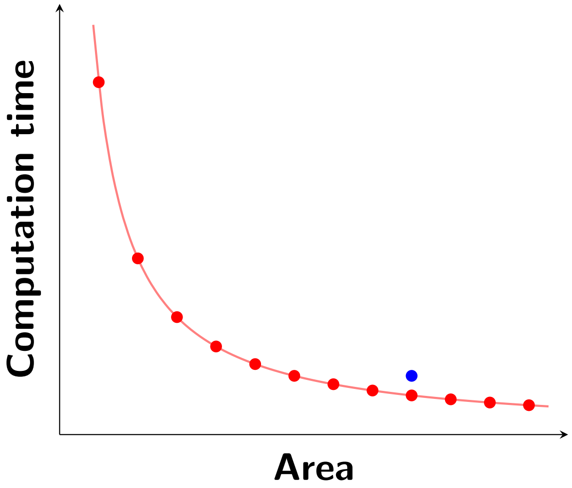
- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)
  - area (slices)
- Different designs for the same computation
  - optimized for latency
  - optimized for compactness



Hardware implementation

- Our target: Field Programmable Gate Array (FPGA)
  - integrated circuit
  - matrix of simple configurable logic cells
  - programmable interconnection
- Performance metric
  - time (ms)
  - area (slices)
  - time–area product
- Different designs for the same computation
  - optimized for latency
  - optimized for compactness
  - optimized for throughput





Contributions

- Fast accelerator for pairings [CHES 2009, IEEE TC 2011]
  Joint work with Beuchat, Detrey, Okamoto and Rodríguez-Henríquez
  - parallel architecture
  - pipelined subquadratic multiplier
- Compact design for pairings reaching 128-bit security
  - composite extension fields [Paring 2010]
  - hyperelliptic curves [CT-RSA 2012]
    Joint work with Aranha, Beuchat and Detrey
- Formulae for sub-quadratic multiplication [WAIFI 2012]
  Joint work with Barbulescu, Detrey and Zimmermann
  - exhaustive search
  - improved formulae for $\mathbb{F}_{3^{5m}}$

Contributions

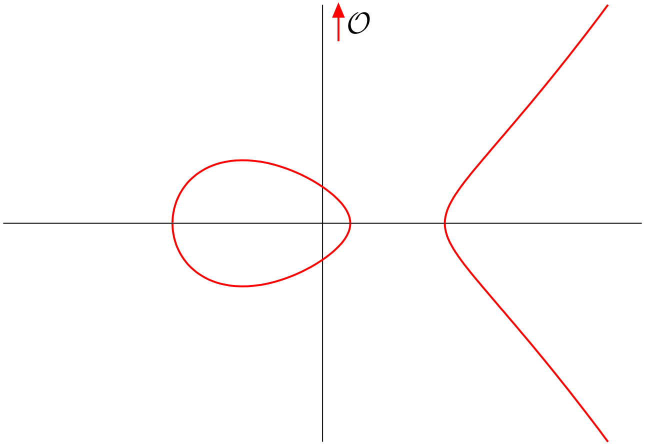
- Fast accelerator for pairings [CHES 2009, IEEE TC 2011]
  Joint work with Beuchat, Detrey, Okamoto and Rodríguez-Henríquez
  - parallel architecture
  - pipelined subquadratic multiplier
- Compact design for pairings reaching 128-bit security
  - composite extension fields [Paring 2010]
  - hyperelliptic curves [CT-RSA 2012]
    Joint work with Aranha, Beuchat and Detrey
- Formulae for sub-quadratic multiplication [WAIFI 2012]
  Joint work with Barbulescu, Detrey and Zimmermann
  - exhaustive search
  - improved formulae for $\mathbb{F}_{3^{5m}}$

Outline of the talk

- Compact design through composite extension fields
- Pairing on genus-2 hyperelliptic curves
- Searching for efficient multiplication algorithms
- Conclusion and Perspectives

Computing the pairing: Miller's algorithm

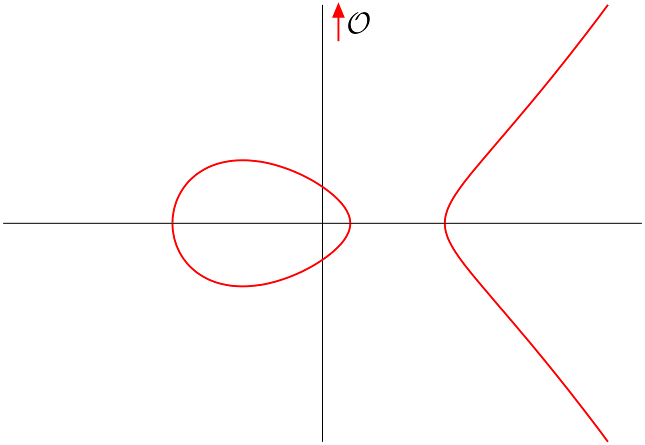
- Computation of the pairing relies on Miller functions: $f_{n,P}$



Computing the pairing: Miller's algorithm

- Computation of the pairing relies on Miller functions: $f_{n,P}$
  - an inductive identity defined by

$$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$

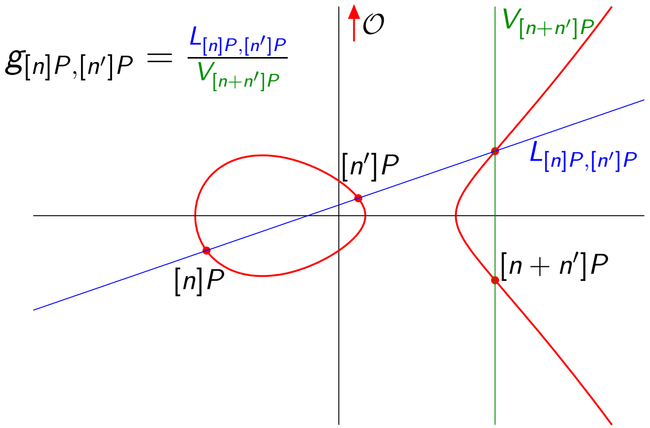


Computing the pairing: Miller's algorithm

- Computation of the pairing relies on Miller functions: $f_{n,P}$
  - an inductive identity defined by

$$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$

  - $g_{[n]P,[n']P}$ derived from the addition of $[n]P$ and $[n']P$

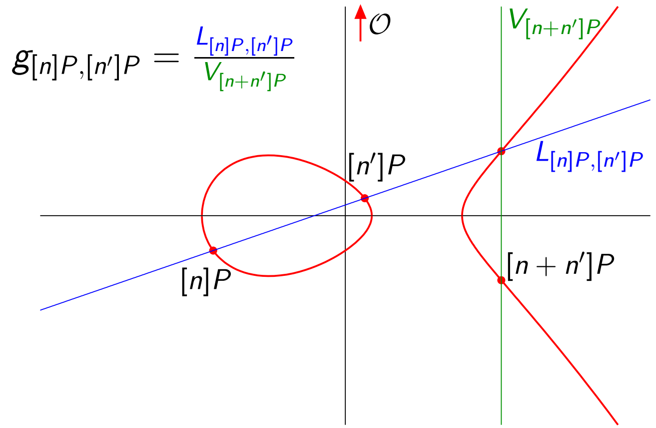


Computing the pairing: Miller's algorithm

- Computation of the pairing relies on Miller functions: $f_{n,P}$
  - an inductive identity defined by

$$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$

  - $g_{[n]P,[n']P}$ derived from the addition of $[n]P$ and $[n']P$



- Tate pairing: $f_{\#E(\mathbb{F}_q),P}$
  - use an addition chain
  - in practice: double-and-add

$\log_2 \#E(\mathbb{F}_q)$ iterations

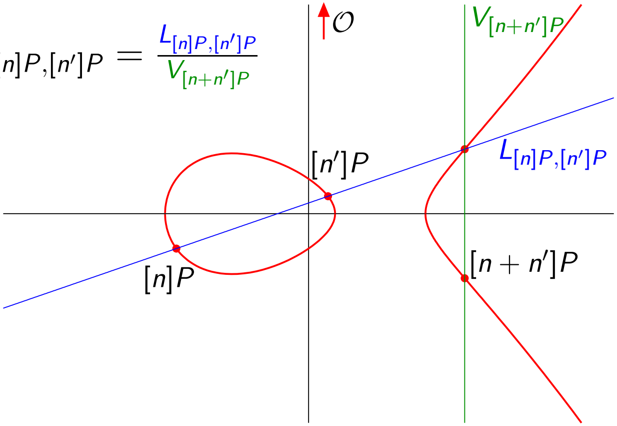
Computing the pairing: Miller's algorithm

- Computation of the pairing relies on Miller functions: $f_{n,P}$
  - an inductive identity defined by

$$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$

  - $g_{[n]P,[n']P}$ derived from the addition of $[n]P$ and $[n']P$

$$g_{[n]P,[n']P} = \frac{L_{[n]P,[n']P}}{V_{[n+n']P}}$$



- Tate pairing: $f_{\#E(\mathbb{F}_q),P}$
  - use an addition chain
  - in practice: double-and-add

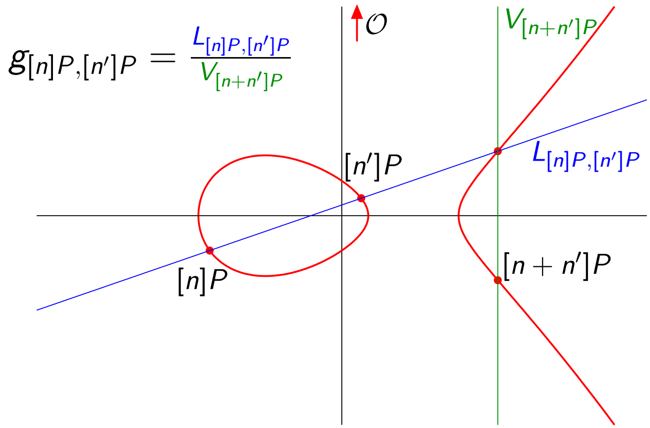
$\log_2 \#E(\mathbb{F}_q)$ iterations

For $E_3(\mathbb{F}_{3^{509}})$	Tate pairing
# iterations	509
$\times$	10330
$+$	45170
$(.)^3$	8136
$(.)^{-1}$	2

- $\#E_3(\mathbb{F}_{3^{509}}) = 3^{509} + 3^{255} + 1$
  - triple-and-add algorithm

Computing the pairing: Miller's algorithm

- Computation of the pairing relies on Miller functions: $f_{n,P}$
  - an inductive identity defined by
    $$\begin{aligned} f_{1,P} &= 1 \\ f_{n+n',P} &= f_{n,P} \cdot f_{n',P} \cdot g_{[n]P,[n']P} \end{aligned}$$
  - $g_{[n]P,[n']P}$ derived from the addition of $[n]P$ and $[n']P$

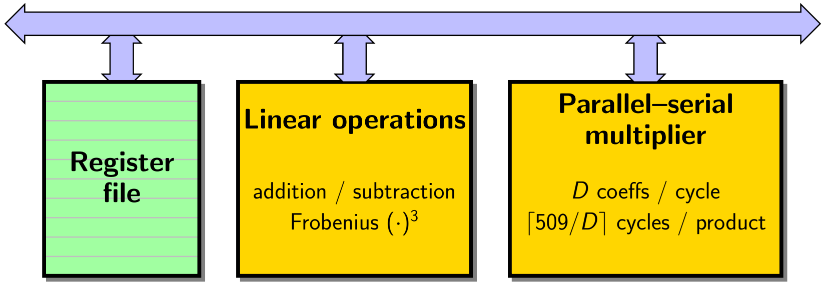


- Tate pairing: $f_{\#E(\mathbb{F}_q),P}$
  - use an addition chain
  - in practice: double-and-add
    $\log_2 \#E(\mathbb{F}_q)$ iterations

For $E_3(\mathbb{F}_{3^{509}})$	Tate pairing	Eta T
# iterations	509	254
$\times$	10330	3638
$+$	45170	17240
$(.)^3$	8136	4068
$(.)^{-1}$	2	1

- $\#E_3(\mathbb{F}_{3^{509}}) = 3^{509} + 3^{255} + 1$
  - triple-and-add algorithm
- Many improvements
  - vertical elimination
  - use of some curve endomorphisms
    - ⋆ Frobenius: Ate
    - ⋆ Verschiebung: Eta, Eta T

An arithmetic coprocessor

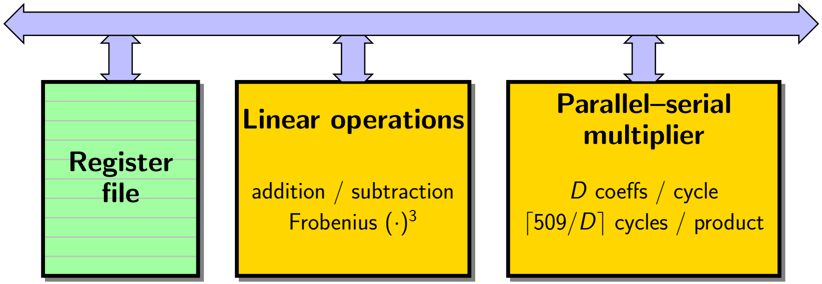


- Only need arithmetic operations in $\mathbb{F}_{3^{509}}$
  - implement a specialized processor
- Multiplication is critical
  - separate linear operations and multiplications
  - careful scheduling to keep multiplier busy

Operation count

$\times$	3638
$+$	17240
$(.)^3$	4068
$(.)^{-1}$	1

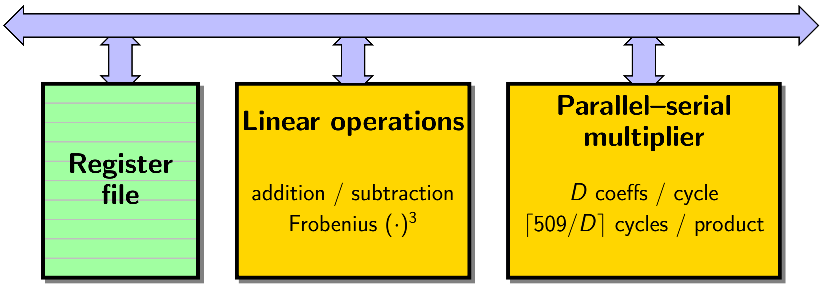
An arithmetic coprocessor



- Only need arithmetic operations in $\mathbb{F}_{3^{509}}$
  - implement a specialized processor
- Multiplication is critical
  - separate linear operations and multiplications
  - careful scheduling to keep multiplier busy
- Inverse is only needed once: Itoh–Tsujii algorithm
  - no need for hardware support

Operation count	
$\times$	3638
$+$	17240
$(.)^3$	4068
$(.)^{-1}$	1

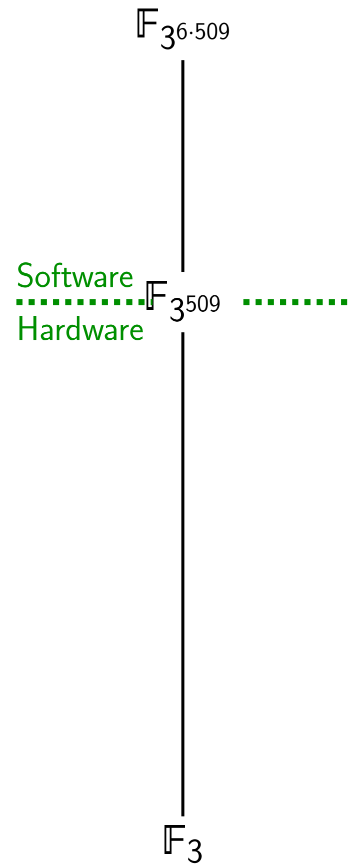
An arithmetic coprocessor



- Only need arithmetic operations in $\mathbb{F}_{3^{509}}$
  - implement a specialized processor
- Multiplication is critical
  - separate linear operations and multiplications
  - careful scheduling to keep multiplier busy
- Inverse is only needed once: Itoh–Tsujii algorithm
  - no need for hardware support
- Synthesis results for $\mathbb{F}_{3^{509}}$: 9625 slices
  - almost fully occupy a Virtex 6 LX 75 T (82%)
  - computation time: ≈ 4 ms

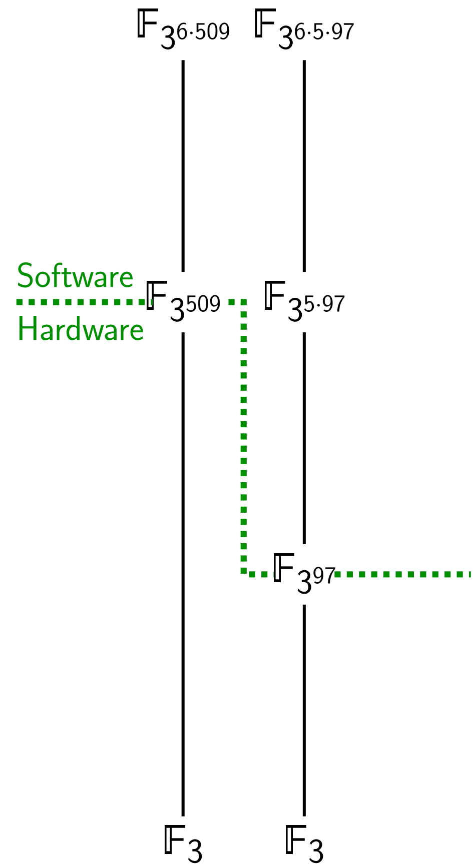
Operation count	
$\times$	3638
$+$	17240
$(.)^3$	4068
$(.)^{-1}$	1

Field of composite extension degree



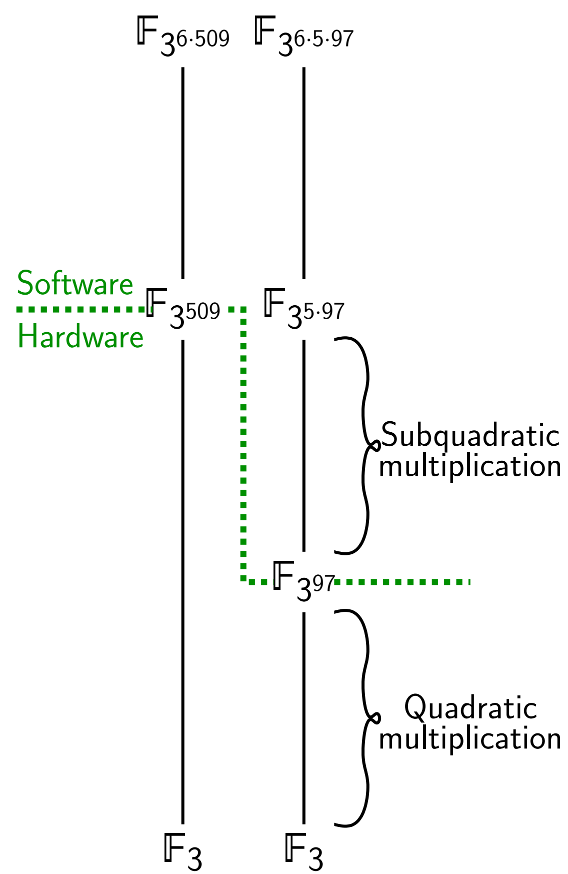
Field of composite extension degree

- Provides some arithmetic advantages
  - smaller datapath



Field of composite extension degree

- Provides some arithmetic advantages
  - smaller datapath
  - efficient multiplication algorithm



Field of composite extension degree

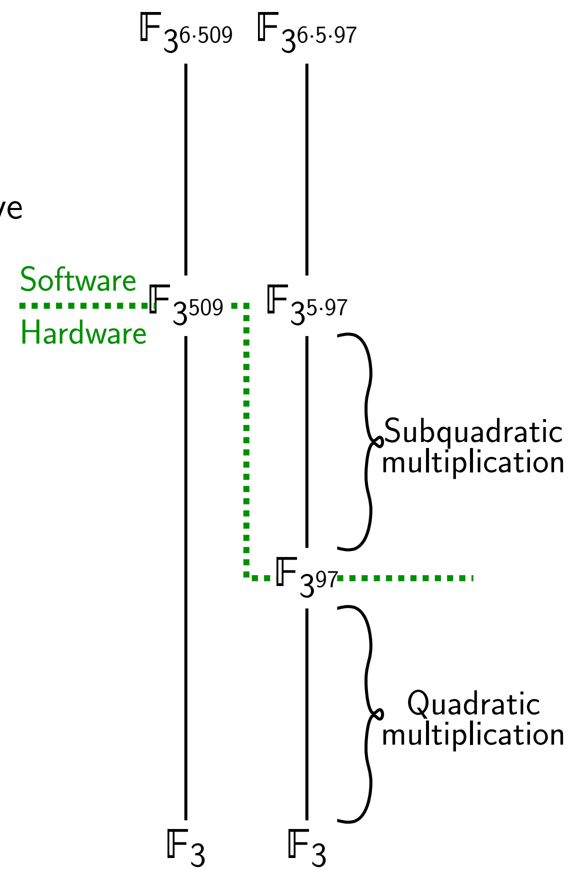
- Provides some arithmetic advantages
  - smaller datapath
  - efficient multiplication algorithm
- Allows Weil Descent based attacks on the curve
  - GHS: using the composite extension degree

    $$\approx 2^{279} \text{ operations}$$

  - SDHP: Granger's algorithm

    $$\approx 2^{142} \text{ operations}$$

  - limited effect on security



Field of composite extension degree

- Provides some arithmetic advantages
  - smaller datapath
  - efficient multiplication algorithm

- Allows Weil Descent based attacks on the curve
  - GHS: using the composite extension degree

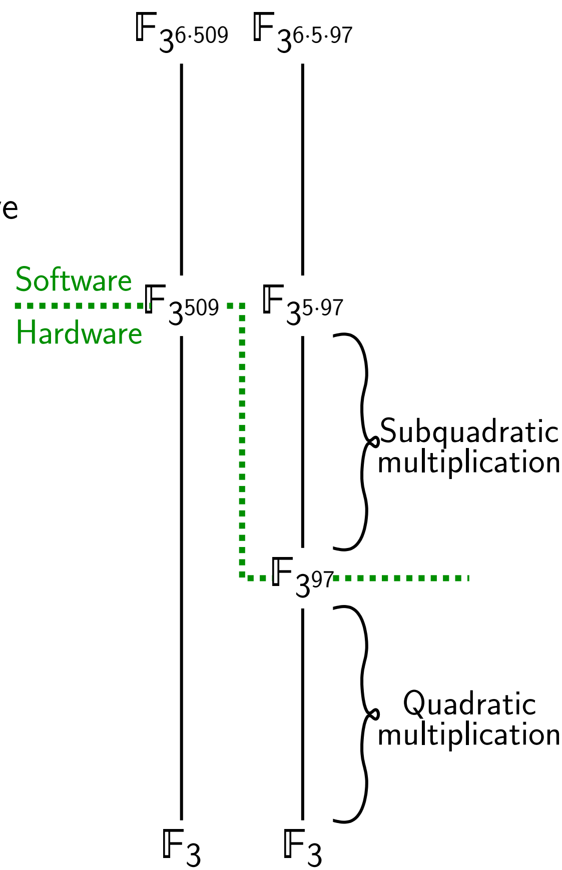
    $$\approx 2^{279} \text{ operations}$$

  - SDHP: Granger's algorithm

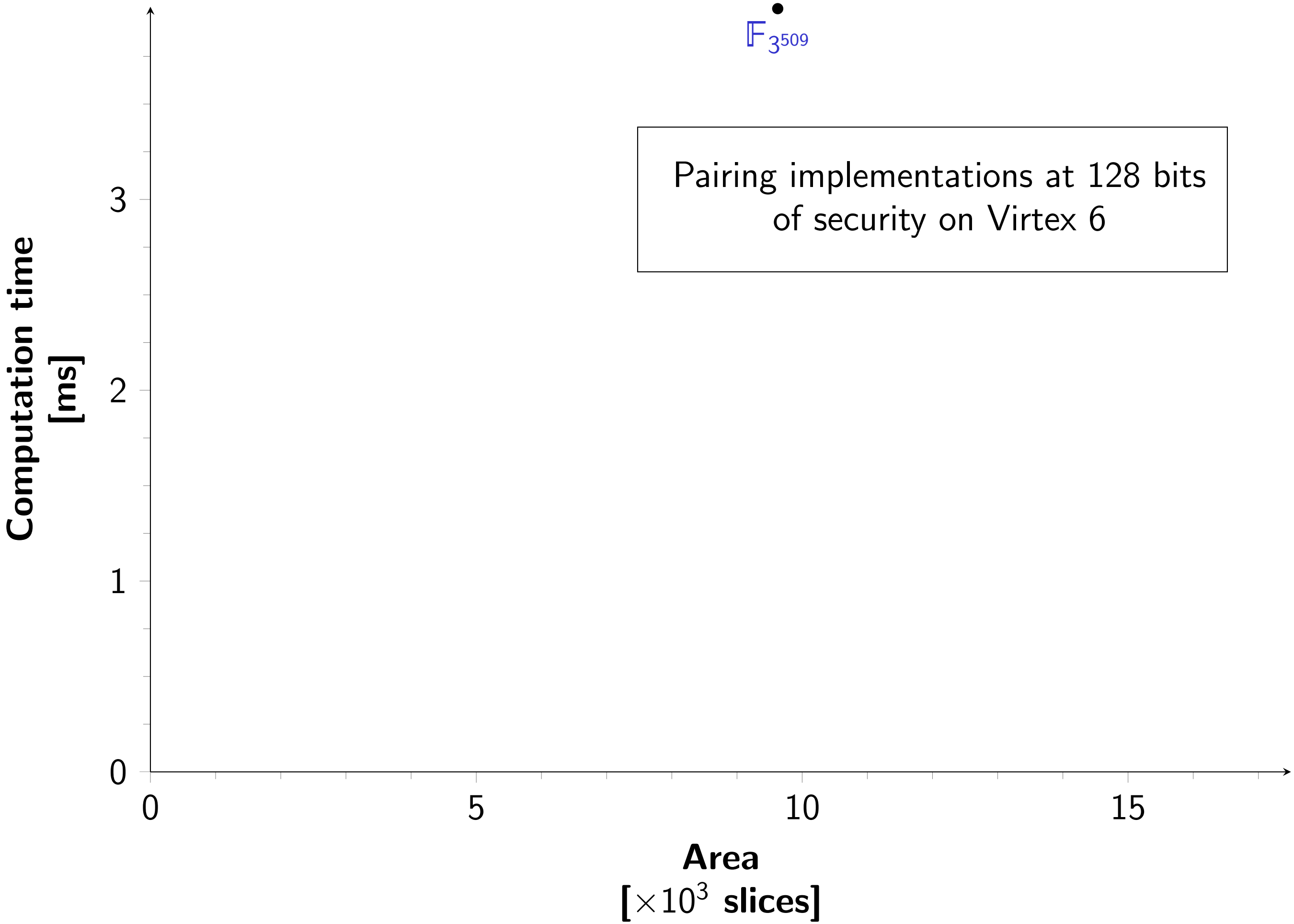
    $$\approx 2^{142} \text{ operations}$$

  - limited effect on security

- Results
  - 1848 slices of the same Virtex 6 LX (15%)
    5.2 times smaller
  - compute a pairing in 1.6 ms
    2.5 times faster

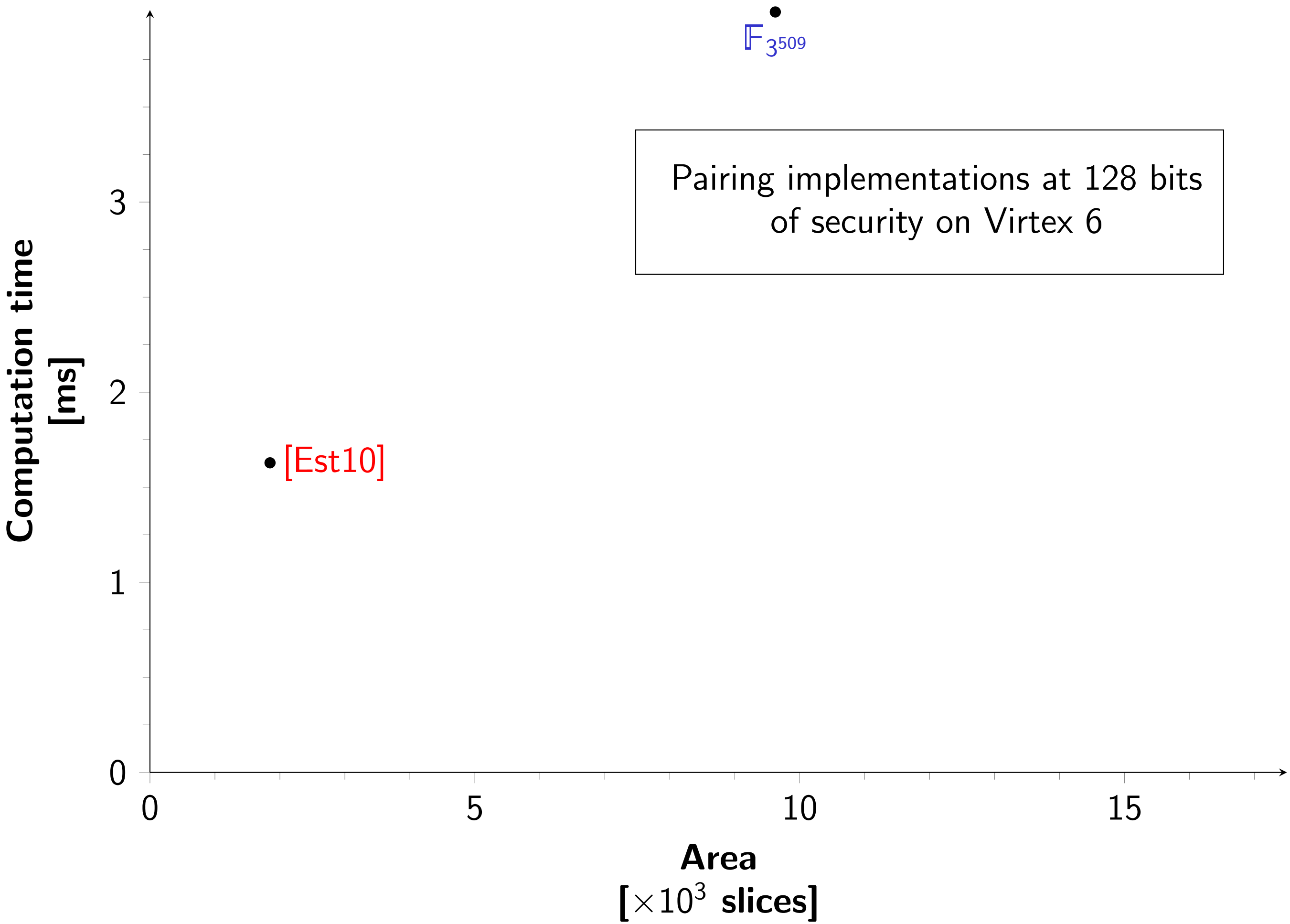


Benchmarks



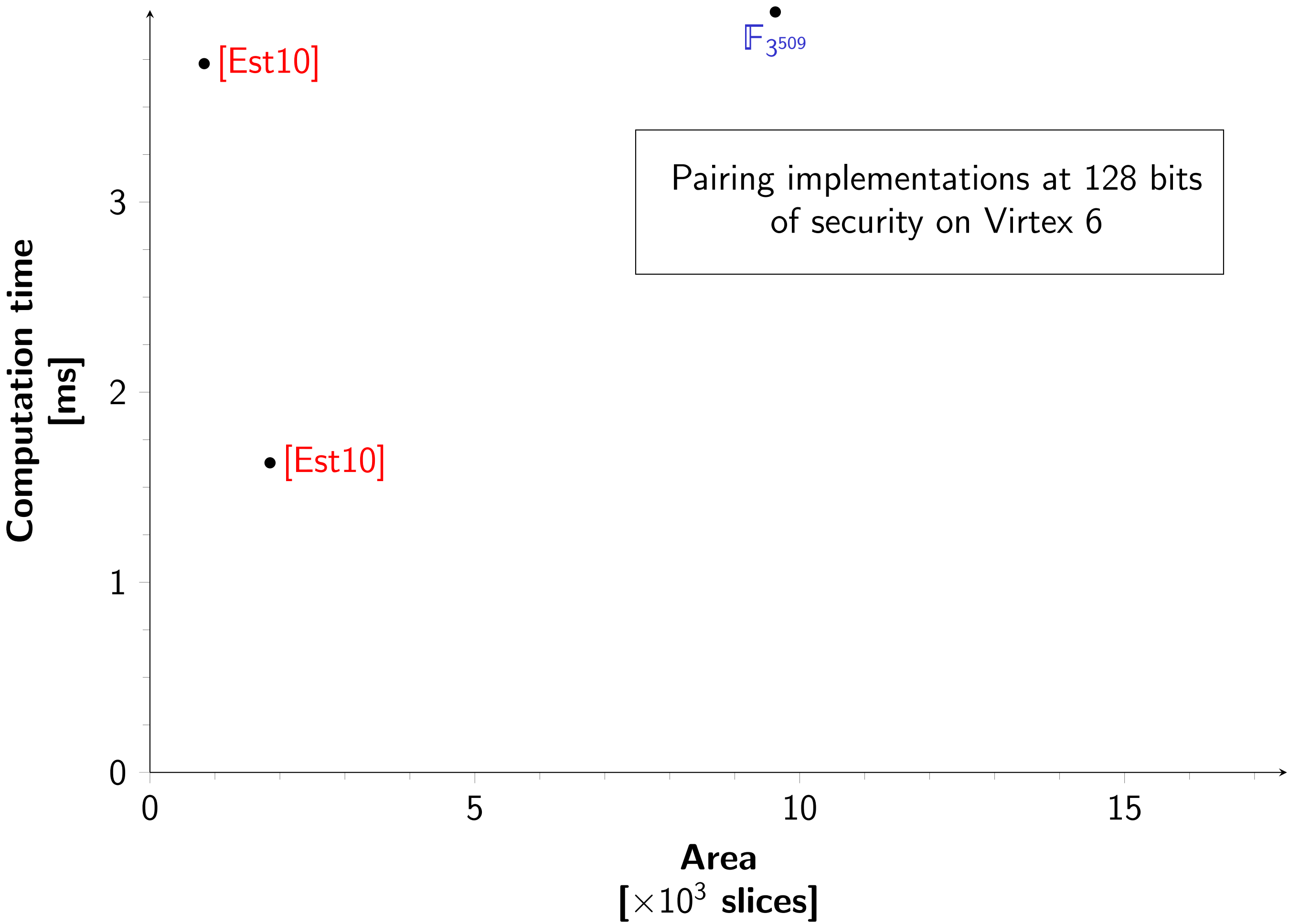
Computation time [ms]
0
1
2
3
$\mathbb{F}_{3^{509}}$
Pairing implementations at 128 bits of security on Virtex 6
0
5
10
15
Area [$\times 10^3$ slices]

Benchmarks

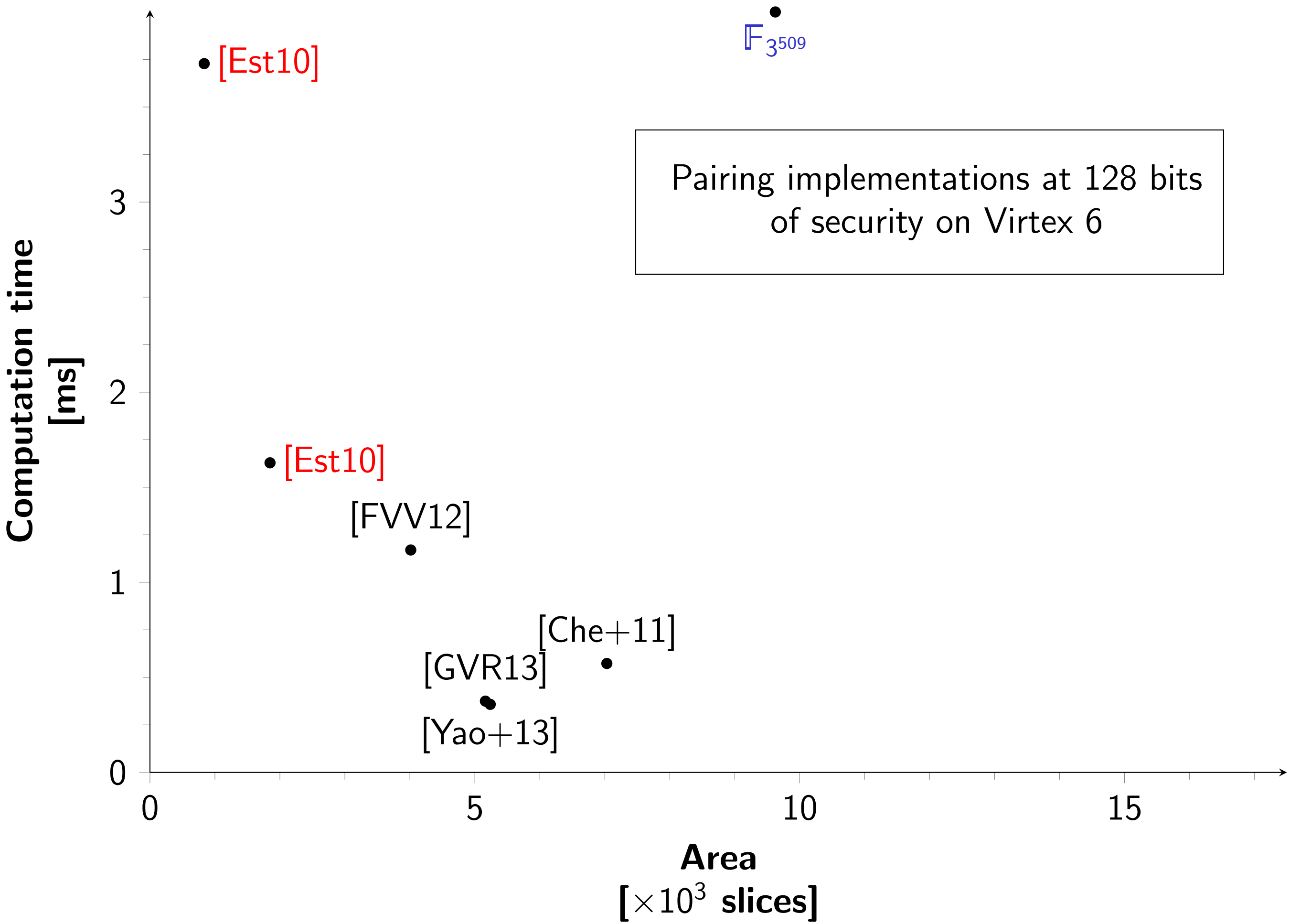


Pairing implementations at 128 bits of security on Virtex 6
$\mathbb{F}_{3^{509}}$
[Est10]
Computation time [ms]
0
1
2
3
0
5
10
15
Area [$\times 10^3$ slices]

Benchmarks

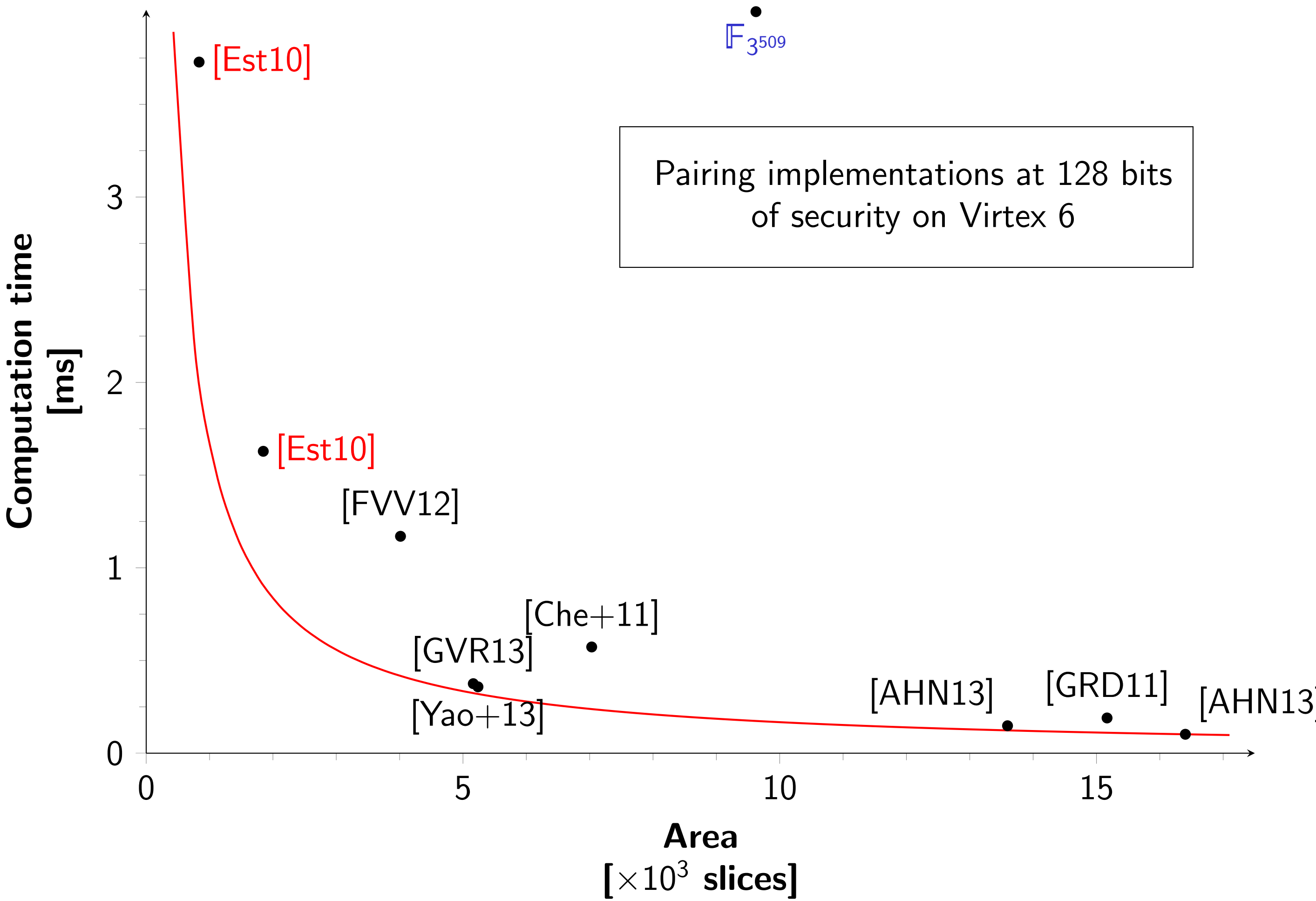

[Est10]
$\mathbb{F}_{3^{509}}$
[Est10]
Pairing implementations at 128 bits of security on Virtex 6
Computation time [ms]
0
1
2
3
0
5
10
15
Area [$\times 10^3$ slices]

Benchmarks



Computation time
[ms]
[Est10]
[Est10]
[FVV12]
[Che+11]
[GVR13]
[Yao+13]
$\mathbb{F}_{3^{509}}$
Pairing implementations at 128 bits of security on Virtex 6
0
1
2
3
0
5
10
15
Area
[$\times 10^3$ slices]

Benchmarks

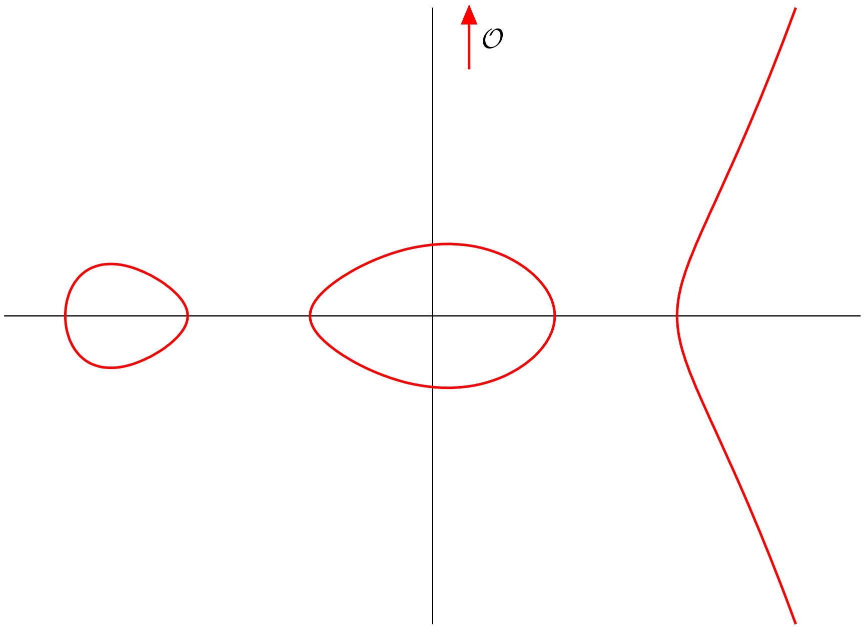

Pairing implementations at 128 bits of security on Virtex 6
Computation time [ms]
Area [$\times 10^3$ slices]
0
1
2
3
0
5
10
15
[Est10]
[Est10]
[FVV12]
[Che+11]
[GVR13]
[Yao+13]
[AHN13]
[GRD11]
[AHN13]
$\mathbb{F}_{3^{509}}$

Outline of the talk

- Compact design through composite extension fields
- Pairing on genus-2 hyperelliptic curves
- Searching for efficient multiplication algorithms
- Conclusion and Perspectives

Genus-2 hyperelliptic curves

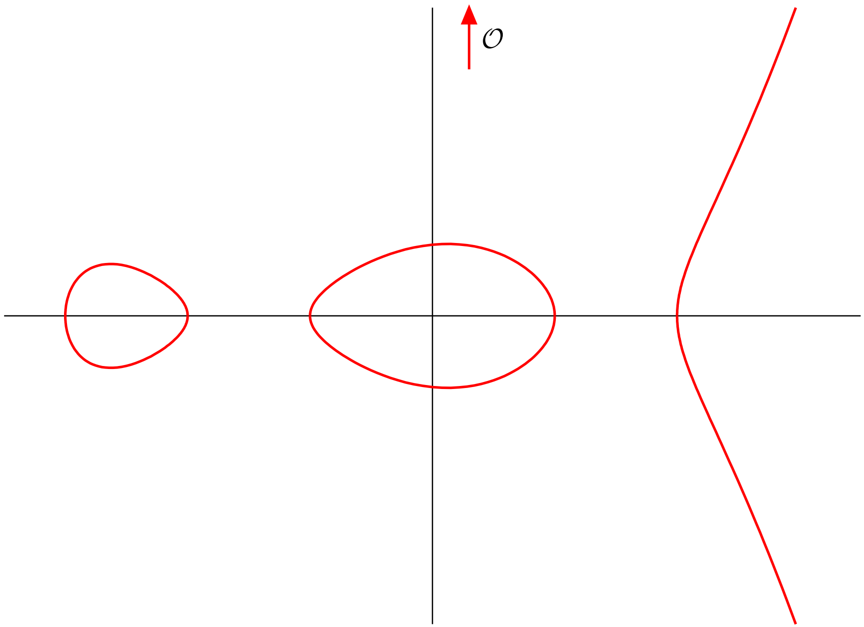
$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

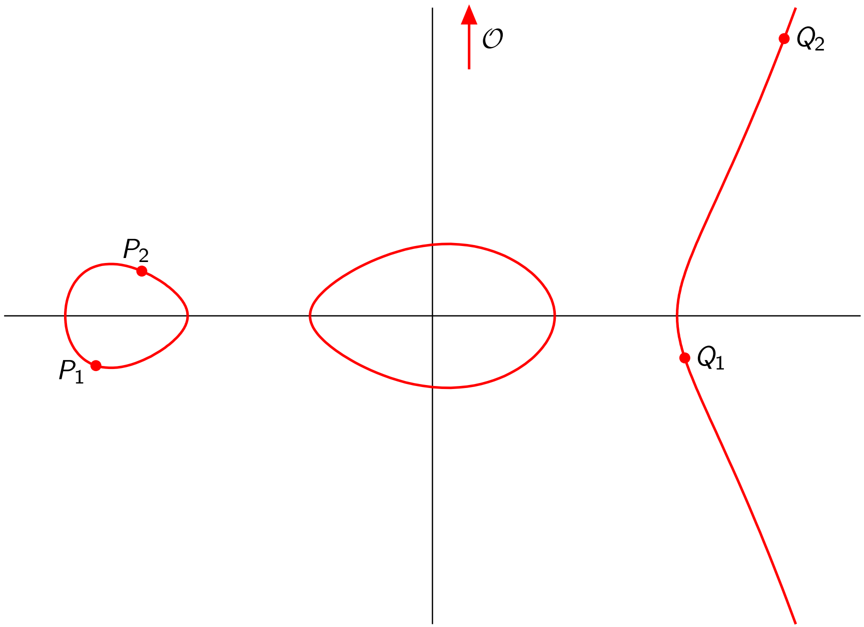
- $C(K)$ not a group!



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

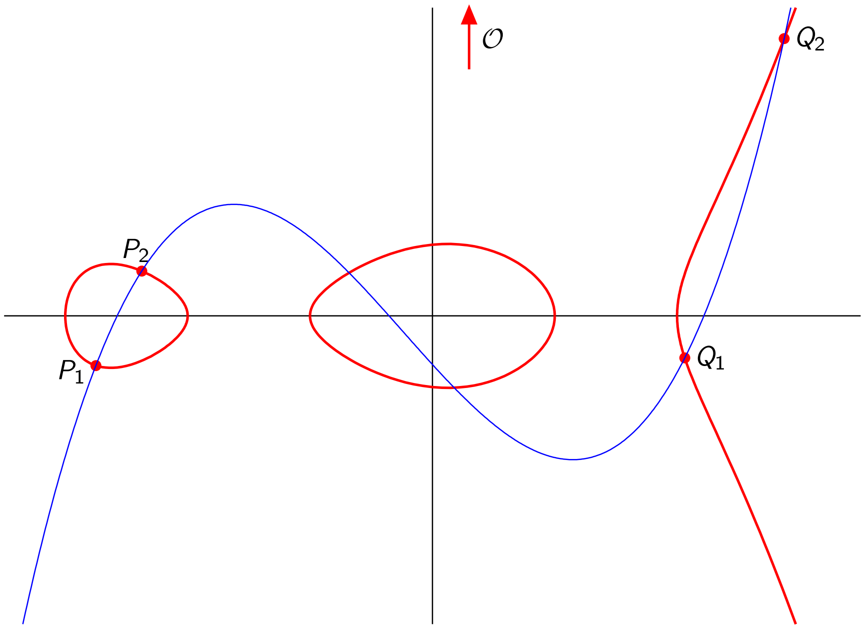
- $C(K)$ not a group!
- But pairs of points
  $\{P_1, P_2\}$



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

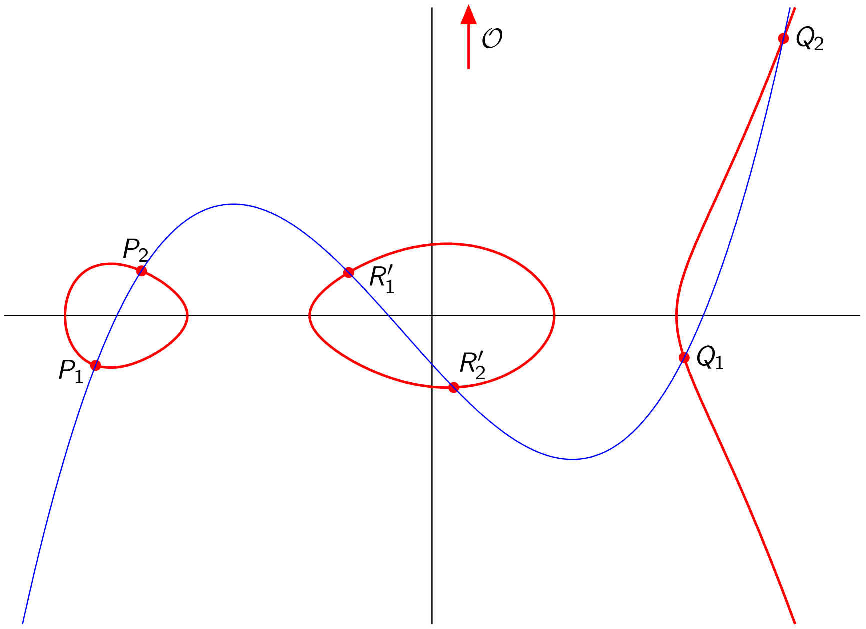
- $C(K)$ not a group!
- But pairs of points $\{P_1, P_2\}$



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

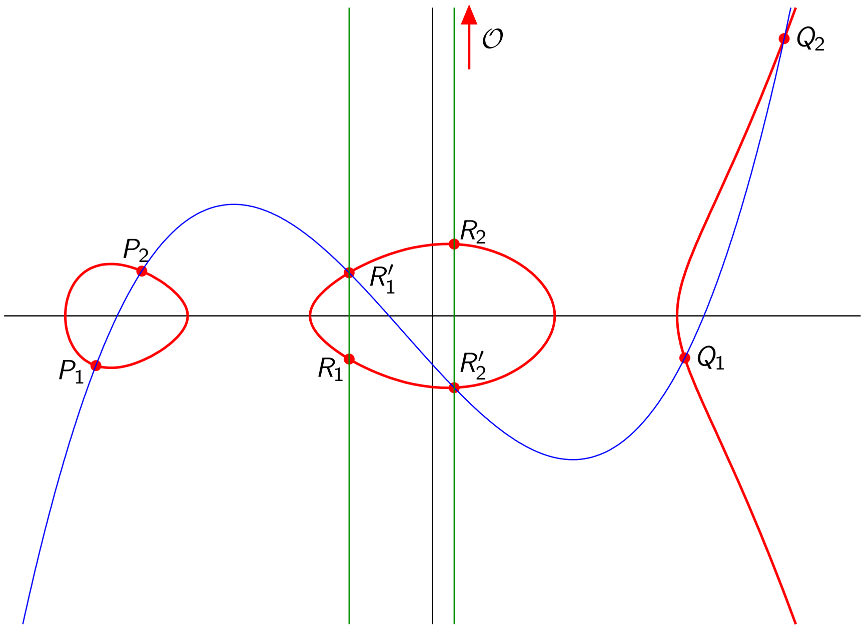
- $C(K)$ not a group!
- But pairs of points
  $\{P_1, P_2\}$



Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$

- $C(K)$ not a group!
- But pairs of points
  $\{P_1, P_2\}$



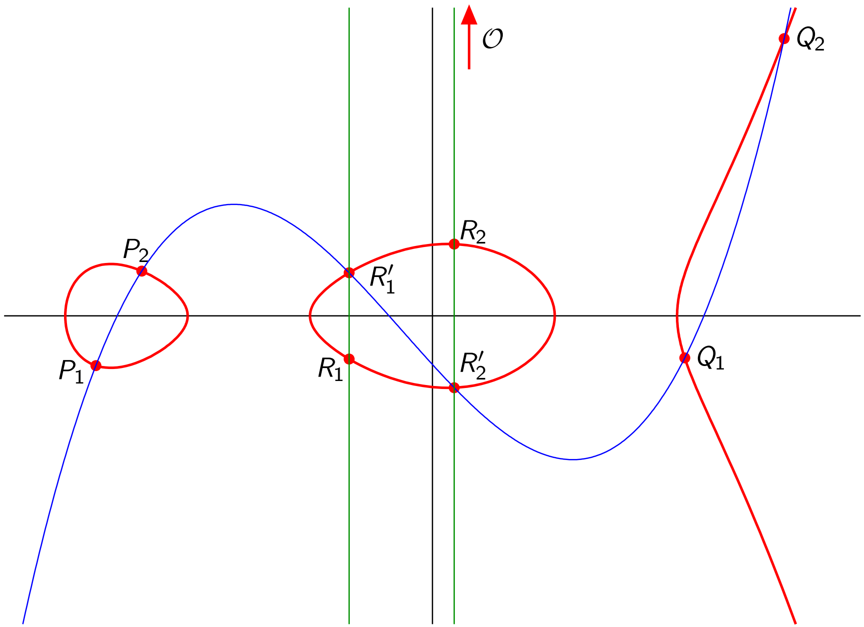
$$\{P_1, P_2\} + \{Q_1, Q_2\} = \{R_1, R_2\}$$

Genus-2 hyperelliptic curves

$$C/K : y^2 + h(x)y = f(x)$$

with $\deg h \leq 2$ and $\deg f = 5$

- $C(K)$ not a group!
- But pairs of points
  $$\{P_1, P_2\}$$
- More formally
  - Jacobian of the curve
    $$\mathrm{Jac}_C$$
  - is a group

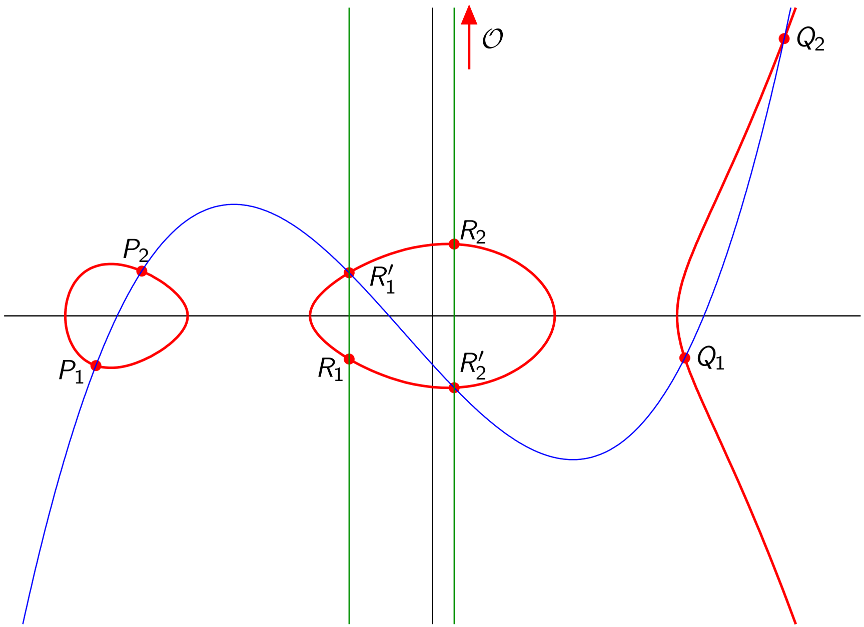


$$\{P_1, P_2\} + \{Q_1, Q_2\} = \{R_1, R_2\}$$

Genus-2 hyperelliptic curves

- $C(K)$ not a group!
- But pairs of points
  $$\{P_1, P_2\}$$
- More formally
  - Jacobian of the curve
    $$\mathrm{Jac}_C$$
  - is a group
- Chosen curves
  $$H_2 : y^2 + y = x^5 + x^3 + d, \text{ with } d \in \{0, 1\}$$
  - characteristic 2
  - supersingular

$$C/K : y^2 + h(x)y = f(x)$$
with $\deg h \leq 2$ and $\deg f = 5$



$$\{P_1, P_2\} + \{Q_1, Q_2\} = \{R_1, R_2\}$$

Optimal Eta

- Parameters for 128-bit security
  - Embedding degree $k = 12$
  - Field: $\mathbb{F}_{2^{367}}$

Optimal Eta

- Parameters for 128-bit security
  - Embedding degree $k = 12$
  - Field: $\mathbb{F}_{2^{367}}$
  - $\#\,\mathrm{Jac}_C(\mathbb{F}_{2^{367}}) = 2^{734} - 2^{551} - 2^{367} + 2^{184} + 1$
- Our pairing algorithm

Algorithm	Tate (double-and-add)
# iterations	734

Optimal Eta

- Parameters for 128-bit security
  - Embedding degree $k = 12$
  - Field: $\mathbb{F}_{2^{367}}$
  - $\#\,\mathrm{Jac}_C(\mathbb{F}_{2^{367}}) = 4 \cdot 8^{244} - 4 \cdot 2^{183} - 2 \cdot 8^{122} + 1$
- Our pairing algorithm
  - Efficient octupling formula: octuple-and-add

Algorithm	Tate (double-and-add)	Tate (octuple-and-add)
# iterations	734	245

Optimal Eta

- Parameters for 128-bit security
  - Embedding degree $k = 12$
  - Field: $\mathbb{F}_{2^{367}}$
  - $\#\,\mathrm{Jac}_C(\mathbb{F}_{2^{367}}) = 4 \cdot 8^{244} - 4 \cdot 2^{183} - 2 \cdot 8^{122} + 1$
- Our pairing algorithm
  - Efficient octupling formula: octuple-and-add
  - adapted Verschiebung: Eta T

Algorithm	Tate (double-and-add)	Tate (octuple-and-add)	Eta T
# iterations	734	245	184

Optimal Eta

- Parameters for 128-bit security
  - Embedding degree $k = 12$
  - Field: $\mathbb{F}_{2^{367}}$
  - $\# \mathrm{Jac}_C(\mathbb{F}_{2^{367}}) = 4 \cdot 8^{244} - 4 \cdot 2^{183} - 2 \cdot 8^{122} + 1$
- Our pairing algorithm
  - Efficient octupling formula: octuple-and-add
  - adapted Verschiebung: Eta T
  - Vercauteren's optimal technique: optimal Eta

Algorithm	Tate (double-and-add)	Tate (octuple-and-add)	Eta T	Optimal Eta
# iterations	734	245	184	123

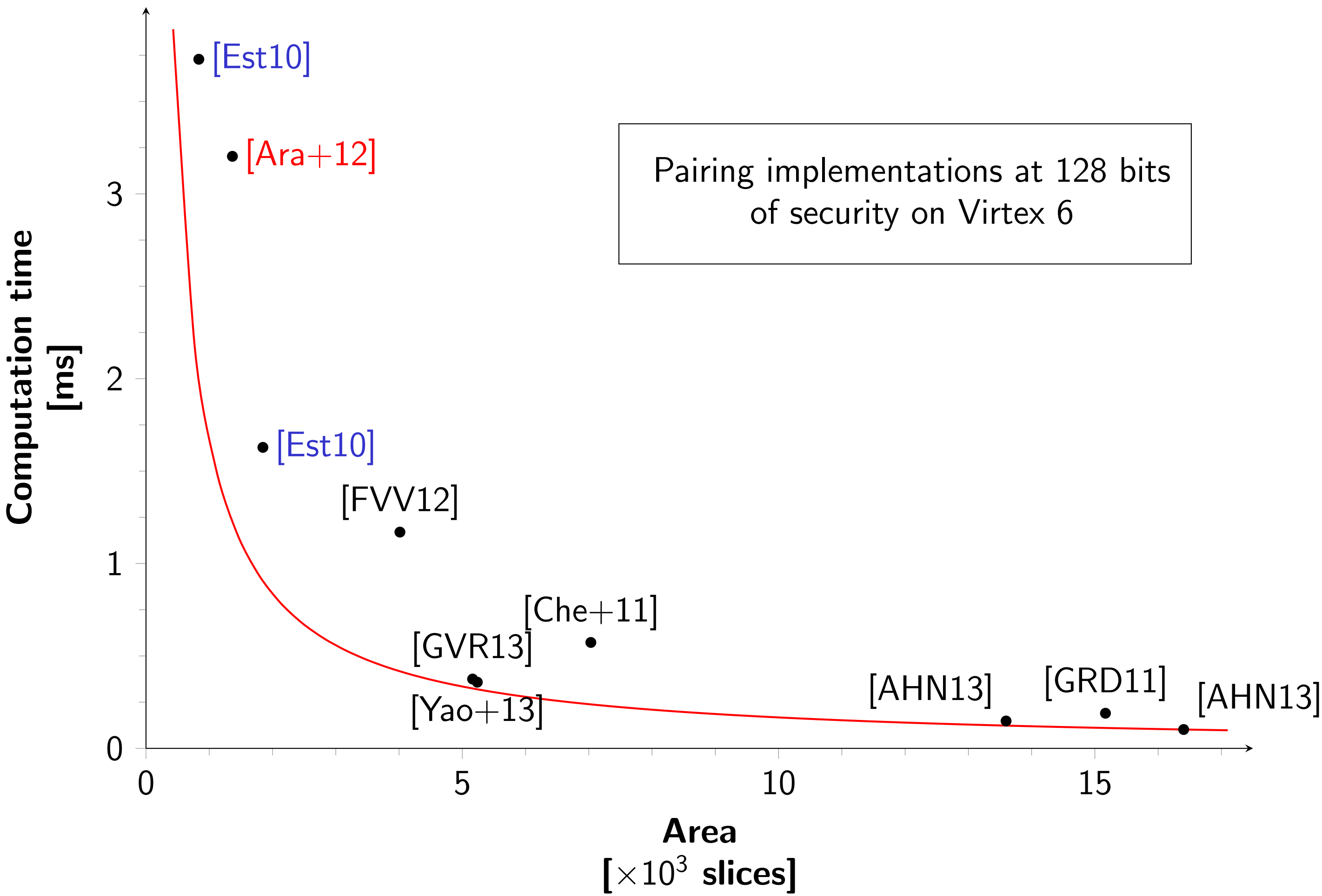
Optimal Eta

- Parameters for 128-bit security
  - Embedding degree $k = 12$
  - Field: $\mathbb{F}_{2^{367}}$
  - $\# \mathrm{Jac}_C(\mathbb{F}_{2^{367}}) = 4 \cdot 8^{244} - 4 \cdot 2^{183} - 2 \cdot 8^{122} + 1$
- Our pairing algorithm
  - Efficient octupling formula: octuple-and-add
  - adapted Verschiebung: Eta T
  - Vercauteren's optimal technique: optimal Eta

Algorithm	Tate (double-and-add)	Tate (octuple-and-add)	Eta T	Optimal Eta
# iterations	734	245	184	123

- Implementation on the previous coprocessor adapted for $\mathbb{F}_{2^{367}}$
  - 1366 slices on the same Virtex 6 LX (12%)
  - 3.2 ms
  - comparable performances with the elliptic case

Benchmarks



Pairing implementations at 128 bits of security on Virtex 6
Computation time [ms]
Area [$\times 10^3$ slices]
0
1
2
3
0
5
10
15
[Est10]
[Ara+12]
[Est10]
[FVV12]
[Che+11]
[GVR13]
[Yao+13]
[AHN13]
[GRD11]
[AHN13]

Outline of the talk

- Compact design through composite extension fields
- Pairing on genus-2 hyperelliptic curves
- Searching for efficient multiplication algorithms
- Conclusion and Perspectives

Origin of the problem

- Polynomial multiplication is an expensive arithmetic operation
- Schoolbook algorithm: quadratic cost

Origin of the problem

- Polynomial multiplication is an expensive arithmetic operation
- Schoolbook algorithm: quadratic cost
- Karatsuba (1962): first subquadratic multiplication algorithm

$$(a_0 + a_1 X)(b_0 + b_1 X) = a_0 b_0 + (a_0 b_1 + a_1 b_0)X + a_1 b_1 X^2$$

Origin of the problem

- Polynomial multiplication is an expensive arithmetic operation
- Schoolbook algorithm: quadratic cost
- Karatsuba (1962): first subquadratic multiplication algorithm

$$
\begin{aligned}
(a_0 + a_1 X)(b_0 + b_1 X) &= a_0 b_0 + (a_0 b_1 + a_1 b_0) X + a_1 b_1 X^2 \\
&= a_0 b_0 + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1) X + a_1 b_1 X^2
\end{aligned}
$$

Origin of the problem

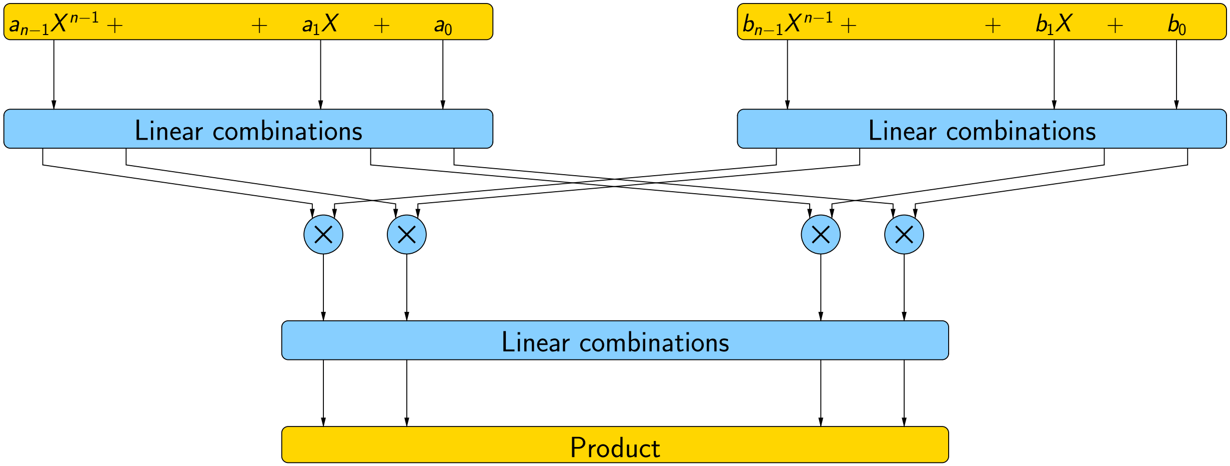
- Polynomial multiplication is an expensive arithmetic operation
- Schoolbook algorithm: quadratic cost
- Karatsuba (1962): first subquadratic multiplication algorithm

$$\begin{aligned}(a_0 + a_1 X)(b_0 + b_1 X) &= a_0 b_0 + (a_0 b_1 + a_1 b_0)X + a_1 b_1 X^2 \\ &= a_0 b_0 + ((a_0 + a_1)(b_0 + b_1) - a_0 b_0 - a_1 b_1)X + a_1 b_1 X^2\end{aligned}$$

- Well-studied problem
  - asymptotic complexity
  - theoretical bilinear complexity
  - small and "cryptographic" size
- *Five, six, and seven-term Karatsuba-like formulae*, P. Montgomery (2005)
  - ad-hoc formulae
  - exhaustive search for five-term multiplication
  - non-exhaustive search for six and seven-term multiplications
- Our approach: improve the search algorithm

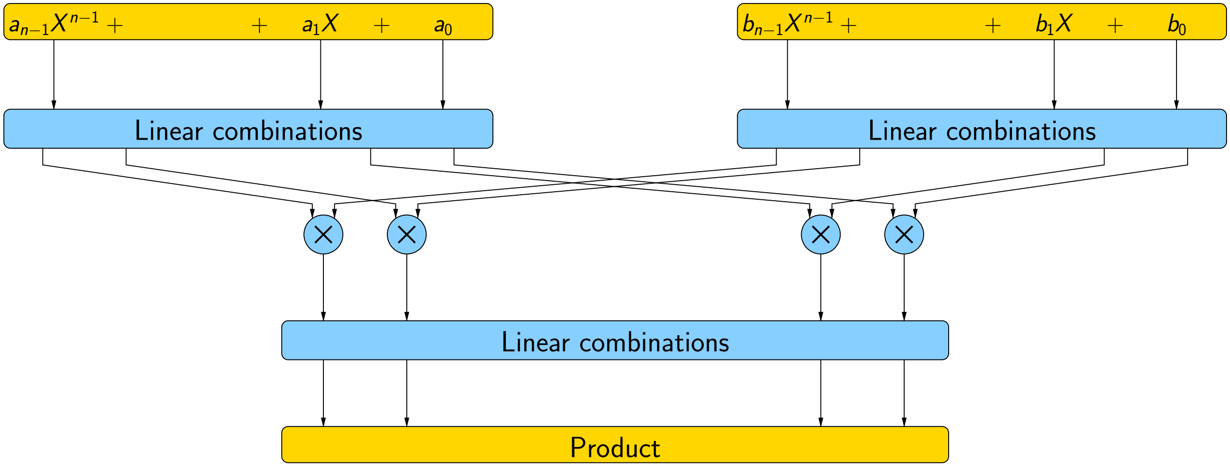
Generalization of the problem

- Model of a multiplication algorithm



Generalization of the problem

- Model of a multiplication algorithm



- Also true for any bilinear application
  - multiplication in extension fields
  - sparse products
  - matrix multiplications
  - ...

Formal framework

Formulation in terms of vector space for an $n \times m$ multiplication over a given field K

- Represent the coefficients of the result and the products as elements of

  $$V \text{ the } nm\text{-dimensional } K\text{-vector space generated by } \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$$

  where the $a_i b_j$'s are seen as formal elements

Formal framework

Formulation in terms of vector space for an $n \times m$ multiplication over a given field K

- Represent the coefficients of the result and the products as elements of

  $$V \text{ the } nm\text{-dimensional } K\text{-vector space generated by } \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$$

  where the $a_i b_j$'s are seen as formal elements

- Our target: the coefficients of the result is a family $\mathcal{T} \subset V$ that spans the target subspace $T = \operatorname{Span} \mathcal{T}$ of V

Formal framework

Formulation in terms of vector space for an $n \times m$ multiplication over a given field K

- Represent the coefficients of the result and the products as elements of

  $$V \text{ the } nm\text{-dimensional } K\text{-vector space generated by } \{a_i b_j\}_{0 \leq i < n, 0 \leq j < m}$$

  where the $a_i b_j$'s are seen as formal elements

- Our target: the coefficients of the result is a family $\mathcal{T} \subset V$ that spans the target subspace $T = \operatorname{Span} \mathcal{T}$ of V

- The set $\mathcal{G}$ of the potential products to use in a formula: the generators

Formal framework

Formulation in terms of vector space for an $n \times m$ multiplication over a given field K

- Represent the coefficients of the result and the products as elements of

  $$V \text{ the } nm\text{-dimensional } K\text{-vector space generated by } \{a_i b_j\}_{0 \le i < n, 0 \le j < m}$$

  where the $a_i b_j$'s are seen as formal elements
- Our target: the coefficients of the result is a family $\mathcal{T} \subset V$ that spans the target subspace $T = \operatorname{Span} \mathcal{T}$ of V
- The set $\mathcal{G}$ of the potential products to use in a formula: the generators
- Goal: find the optimal formulae (i.e. with a minimum number of products)
  - for increasing k until a solution is found
  - find each subset $\mathcal{W} \subset \mathcal{G}$ of exactly k products
  - which gives a valid formula (i.e. that lineary generates the coefficients of the result)

    $$\mathcal{T} \subset \operatorname{Span} \mathcal{W}$$

Resolution

- Naive approach: test each subset of k potential products

```
expand_family(∅, 𝒢)

procedure expand_family(𝒲, ℋ)
   if #𝒲 = k then
      if 𝒯 ⊂ Span 𝒲 then
         𝒲 is a solution
   else

      while ℋ ≠ ∅ do
         Pick a h in ℋ
         ℋ ← ℋ \ {h}
         expand_family(𝒲 ∪ {h}, ℋ)
end procedure
```

- Complexity depends on

$$\binom{\#\mathcal{G}}{k}$$

Resolution

- Naive approach: test each subset of k potential products
- Better approach: test each vector space of dimension k generated by potential products

```
expand_subspace({0}, 𝒢)

procedure expand_subspace(W, ℋ)
    if dim W = k then
        if 𝒯 ⊂ W then
            W is a solution
    else
        ℋ ← ℋ \ W
        while ℋ ≠ ∅ do
            Pick a h in ℋ
            ℋ ← ℋ \ {h}
            expand_subspace(W ⊕ Span(h), ℋ)
end procedure
```

- Complexity still depends on

$$\binom{\#\mathcal{G}}{k}$$

Resolution

- Naive approach: test each subset of k potential products
- Better approach: test each vector space of dimension k generated by potential products
- Even better approach: part of the solution is already known, use incomplete basis theorem

$\texttt{expand\_subspace}(\mathcal{T}, \mathcal{G})$

procedure $\texttt{expand\_subspace}(W, \mathcal{H})$
  if $\dim W = k$ **then**
    if $\operatorname{rank}(W \cap \mathcal{G}) = k$ **then**
      W is a solution
  else
    $\mathcal{H} \leftarrow \mathcal{H} \setminus W$
    while $\mathcal{H} \neq \emptyset$ **do**
      Pick a h in $\mathcal{H}$
      $\mathcal{H} \leftarrow \mathcal{H} \setminus \{h\}$
      $\texttt{expand\_subspace}(W \oplus \operatorname{Span}(h), \mathcal{H})$
end procedure

- Complexity now depends on

$$\binom{\#\mathcal{G}}{k - \operatorname{rank}\mathcal{T}}$$

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	**# of tests**	**# of solutions**	**# of formulae**	**Computation time (1 core)**
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

$$
\begin{aligned}
\mathcal{G} = \{ & a_0 \cdot b_0, & a_1 \cdot b_0, && (a_0 + a_1) \cdot b_0, && a_2 \cdot b_0, && (a_0 + a_2) \cdot b_0, && \ldots \\
& a_0 \cdot b_1, & a_1 \cdot b_1, && (a_0 + a_1) \cdot b_1, && a_2 \cdot b_1, && (a_0 + a_2) \cdot b_1, && \ldots \\
& a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_1), && (a_0 + a_1) \cdot (b_0 + b_1), && a_2 \cdot (b_0 + b_1), && (a_0 + a_2) \cdot (b_0 + b_1), && \ldots \\
& a_0 \cdot b_2, & a_1 \cdot b_2, && (a_0 + a_1) \cdot b_2, && a_2 \cdot b_2, && (a_0 + a_2) \cdot b_2, && \ldots \\
& a_0 \cdot (b_0 + b_2), & a_1 \cdot (b_0 + b_2), && (a_0 + a_1) \cdot (b_0 + b_2), && a_2 \cdot (b_0 + b_2), && (a_0 + a_2) \cdot (b_0 + b_2), && \ldots \\
& \ldots \}
\end{aligned}
$$

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

$$
\begin{array}{llllll}
\mathcal{G} = \{a_0 \cdot b_0, & a_1 \cdot b_0, & (a_0 + a_1) \cdot b_0, & a_2 \cdot b_0, & (a_0 + a_2) \cdot b_0, & \ldots \\
a_0 \cdot b_1, & a_1 \cdot b_1, & (a_0 + a_1) \cdot b_1, & a_2 \cdot b_1, & (a_0 + a_2) \cdot b_1, & \ldots \\
a_0 \cdot (b_0 + b_1), & a_1 \cdot (b_0 + b_1), & (a_0 + a_1) \cdot (b_0 + b_1), & a_2 \cdot (b_0 + b_1), & (a_0 + a_2) \cdot (b_0 + b_1), & \ldots \\
a_0 \cdot b_2, & a_1 \cdot b_2, & (a_0 + a_1) \cdot b_2, & a_2 \cdot b_2, & (a_0 + a_2) \cdot b_2, & \ldots \\
a_0 \cdot (b_0 + b_2), & a_1 \cdot (b_0 + b_2), & (a_0 + a_1) \cdot (b_0 + b_2), & a_2 \cdot (b_0 + b_2), & (a_0 + a_2) \cdot (b_0 + b_2), & \ldots \\
\ldots\} & & & & &
\end{array}
$$

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

- Optimal formulae for sparse multiplication useful in pairing computation
  - in the genus-2 pairing, from 11 to 9 subproducts

Some results

- Multiplication of $n \times m$ term binary polynomials

Ring	$n \times m$	$\#\mathcal{G}$	k	# of tests	# of solutions	# of formulae	Computation time (1 core)
$\mathbb{F}_2[X]$	2×2	9	3	1	1	1	0
	3×3	49	6	9	3	9	0
	4×4	225	9	$6.60 \cdot 10^3$	4	4	30 ms
	5×5	961	13	$9.65 \cdot 10^9$	27	27	2 d 15 h
	6×6	3 969	14	$4.37 \cdot 10^9$	—	—	7 d
	6×6	(Sym.) 63	17	$8.08 \cdot 10^6$	6	54	18 s
	7×7	(Sym.) 127	22	$3.38 \cdot 10^{12}$	2 618	19 550	184 d

- Optimal formulae for sparse multiplication useful in pairing computation
  - in the genus-2 pairing, from 11 to 9 subproducts
- Optimal multiplication for the extensions $\mathbb{F}_{3^{5m}}$
  - 11 subproducts instead of 12 previously
  - yields a 5% improvement for the pairing on E_3

Outline of the talk

- Compact design through composite extension fields
- Pairing on genus-2 hyperelliptic curves
- Searching for efficient multiplication algorithms
- Conclusion and Perspectives

Conclusion

- Hardware implementations of pairing
- An algorithm to search for multiplication formulae

Conclusion

- Hardware implementations of pairing
- An algorithm to search for multiplication formulae
- Unified framework for constructing pairing algorithms
  - lot of literature on pairing algorithms
  - generally concepts and results only for specific cases
  - covers both elliptic and hyperelliptic cases
  - covers the different variants of the Tate pairing:
    - Ate, Eta, Eta T, optimal Ate, ...

Conclusion

- Hardware implementations of pairing
- An algorithm to search for multiplication formulae
- Unified framework for constructing pairing algorithms
  - lot of literature on pairing algorithms
  - generally concepts and results only for specific cases
  - covers both elliptic and hyperelliptic cases
  - covers the different variants of the Tate pairing:
    - Ate, Eta, Eta T, optimal Ate, ...
- General method for cryptographic implementations
  - study mathematical structures
  - fix parameters thanks to cryptanalysis
  - algorithmic optimizations
  - choose the right arithmetic representation
  - implement different hardware accelerators

Perspectives

- Lower-level architecture
  - FPGA is a good prototyping platform
  - but with limited uses in real-life devices
  - develop skills in ASIC designs
  - power consumption awareness
- Integrate side-channel counter-measures
  - side-channel attacks are very effective threats
  - embedded systems need to be protected
- Use this method on different cryptographic primitives
  - scalar multiplication on hyperelliptic curves
  - lattice-based cryptography