RAIM'09 — October 27, 2009

Hardware Accelerator for the Tate Pairing based on Karatsuba Multipliers

Jérémie Detrey

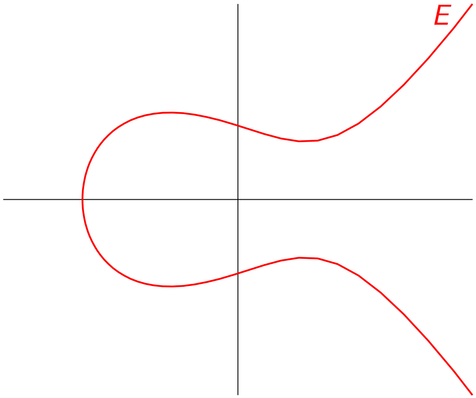
CACAO project-team, LORIA
INRIA Nancy – Grand-Est
Jeremie.Detrey@loria.fr

Joint work with:

Jean-Luc Beuchat	LCIS, University of Tsukuba, Japan
Nicolas Estibals	CACAO, LORIA, Nancy, France
Eiji Okamoto	LCIS, University of Tsukuba, Japan
Francisco Rodríguez-Henríquez	CINVESTAV-IPN, Mexico City, Mexico

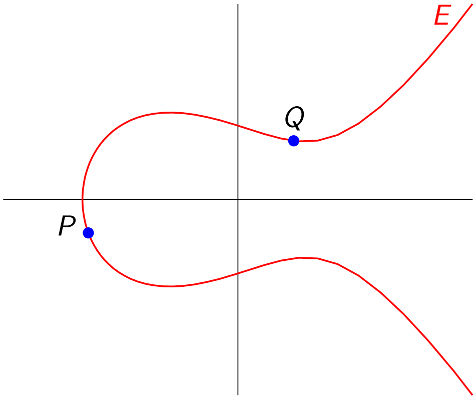
Elliptic curves

- E defined by a Weierstraß equation of the form $y^2 = x^3 + Ax + B$



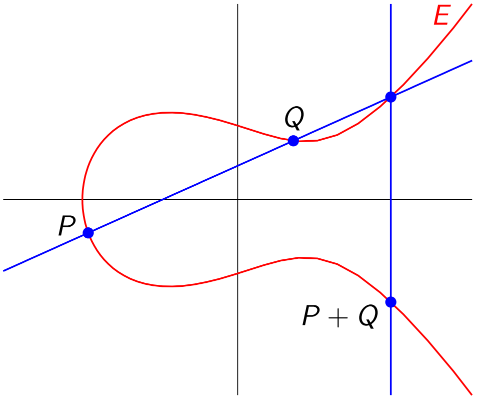
Elliptic curves

- E defined by a Weierstraß equation of the form $y^2 = x^3 + Ax + B$
- $E(K)$ set of rational points over a field K



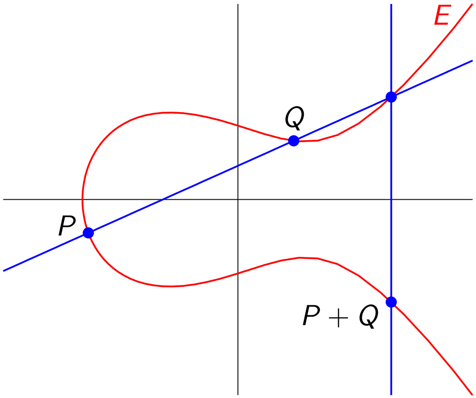
Elliptic curves

- E defined by a Weierstraß equation of the form $y^2 = x^3 + Ax + B$
- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$



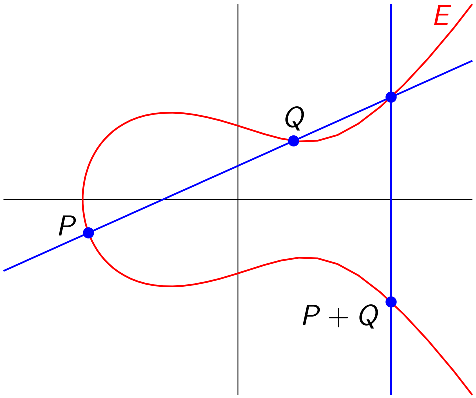
Elliptic curves

- E defined by a Weierstraß equation of the form $y^2 = x^3 + Ax + B$
- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$
- Many applications in cryptography since 1985
  - EC-based Diffie-Hellman key exchange
  - EC-based Digital Signature Algorithm
  - ...
- Interest: smaller keys than usual cryptosystems (RSA, DSA, ElGamal, ...)



Elliptic curves

- E defined by a Weierstraß equation of the form $y^2 = x^3 + Ax + B$
- $E(K)$ set of rational points over a field K
- Additive group law over $E(K)$
- Many applications in cryptography since 1985
  - EC-based Diffie-Hellman key exchange
  - EC-based Digital Signature Algorithm
  - ...
- Interest: smaller keys than usual cryptosystems (RSA, DSA, ElGamal, ...)
- But there's more: bilinear pairings



Outline of the talk

- Pairing-based cryptography
- Hardware accelerator for the Tate pairing
- Implementation results
- Concluding thoughts

Outline of the talk

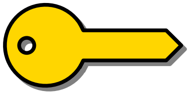
- Pairing-based cryptography
- Hardware accelerator for the Tate pairing
- Implementation results
- Concluding thoughts

Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$

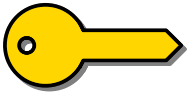
Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



Group cryptography

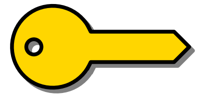
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

Group cryptography

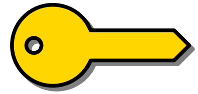
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



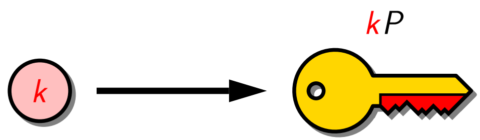
- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$

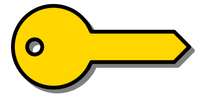


- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

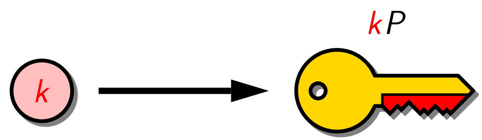


Group cryptography

- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



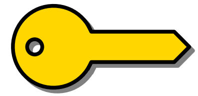
- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$



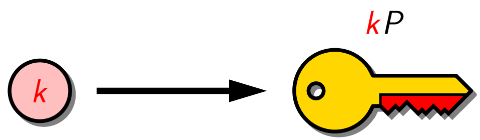
- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

Group cryptography

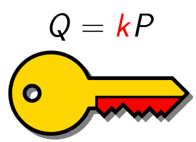
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

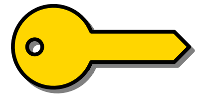


- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

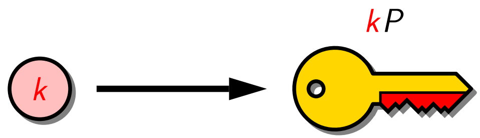


Group cryptography

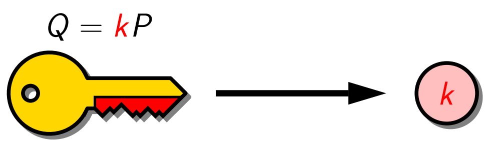
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$

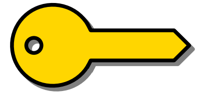


- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$

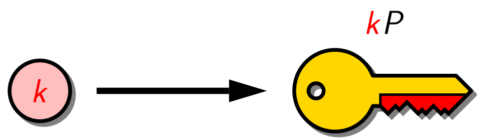


Group cryptography

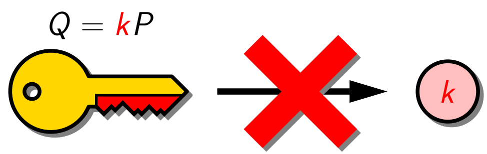
- $(\mathbb{G}_1, +)$, an additively-written cyclic group of prime order $\#\mathbb{G}_1 = \ell$
- P, a generator of the group: $\mathbb{G}_1 = \langle P \rangle$



- Scalar multiplication: for any integer k, we have $kP = \underbrace{P + P + \cdots + P}_{k \text{ times}}$



- Discrete logarithm: given $Q \in \mathbb{G}_1$, compute k such that $Q = kP$



- We assume that the discrete logarithm problem (DLP) in $\mathbb{G}_1$ is hard

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

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

  that satisfies the following conditions:
  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:

$$\hat{e}(Q_1 + Q_2, R) = \hat{e}(Q_1, R) \cdot \hat{e}(Q_2, R) \qquad \hat{e}(Q, R_1 + R_2) = \hat{e}(Q, R_1) \cdot \hat{e}(Q, R_2)$$

  - computability: $\hat{e}$ can be efficiently computed

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map
$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$
that satisfies the following conditions:
  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:
$$\hat{e}(Q_1 + Q_2, R) = \hat{e}(Q_1, R) \cdot \hat{e}(Q_2, R) \qquad \hat{e}(Q, R_1 + R_2) = \hat{e}(Q, R_1) \cdot \hat{e}(Q, R_2)$$
  - computability: $\hat{e}$ can be efficiently computed
- Immediate property: for any two integers k_1 and k_2
$$\hat{e}(k_1 Q, k_2 R) = \hat{e}(Q, R)^{k_1 k_2}$$

Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

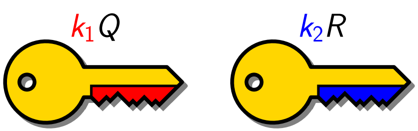
$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

  that satisfies the following conditions:
  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:

$$\hat{e}(Q_1 + Q_2, R) = \hat{e}(Q_1, R) \cdot \hat{e}(Q_2, R) \qquad \hat{e}(Q, R_1 + R_2) = \hat{e}(Q, R_1) \cdot \hat{e}(Q, R_2)$$

  - computability: $\hat{e}$ can be efficiently computed
- Immediate property: for any two integers k_1 and k_2

$$\hat{e}(k_1 Q, k_2 R) = \hat{e}(Q, R)^{k_1 k_2}$$



Bilinear pairings

- $(\mathbb{G}_2, \times)$, a multiplicatively-written cyclic group of order $\#\mathbb{G}_2 = \#\mathbb{G}_1 = \ell$
- A bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ is a map

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

  that satisfies the following conditions:

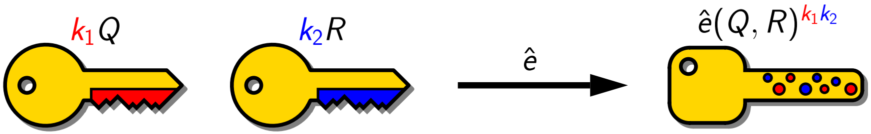
  - non-degeneracy: $\hat{e}(P, P) \neq 1_{\mathbb{G}_2}$ (equivalently $\hat{e}(P, P)$ generates $\mathbb{G}_2$)
  - bilinearity:

$$\hat{e}(Q_1 + Q_2, R) = \hat{e}(Q_1, R) \cdot \hat{e}(Q_2, R) \qquad \hat{e}(Q, R_1 + R_2) = \hat{e}(Q, R_1) \cdot \hat{e}(Q, R_2)$$

  - computability: $\hat{e}$ can be efficiently computed

- Immediate property: for any two integers k_1 and k_2

$$\hat{e}(k_1 Q, k_2 R) = \hat{e}(Q, R)^{k_1 k_2}$$



Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes–Okamoto–Vanstone and Frey–Rück attacks, 1993 and 1994

$$\mathrm{DLP}_{\mathbb{G}_1}$$

$$kP$$

Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes–Okamoto–Vanstone and Frey–Rück attacks, 1993 and 1994

$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes–Okamoto–Vanstone and Frey–Rück attacks, 1993 and 1994

$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

  - for cryptographic applications, we will also require the DLP in $\mathbb{G}_2$ to be hard

Pairings in cryptography

- At first, used to attack supersingular elliptic curves
  - Menezes–Okamoto–Vanstone and Frey–Rück attacks, 1993 and 1994

$$\begin{array}{ccc} \mathrm{DLP}_{\mathbb{G}_1} & <_{\mathrm{P}} & \mathrm{DLP}_{\mathbb{G}_2} \\ kP & \longrightarrow & \hat{e}(kP, P) = \hat{e}(P, P)^k \end{array}$$

  - for cryptographic applications, we will also require the DLP in $\mathbb{G}_2$ to be hard

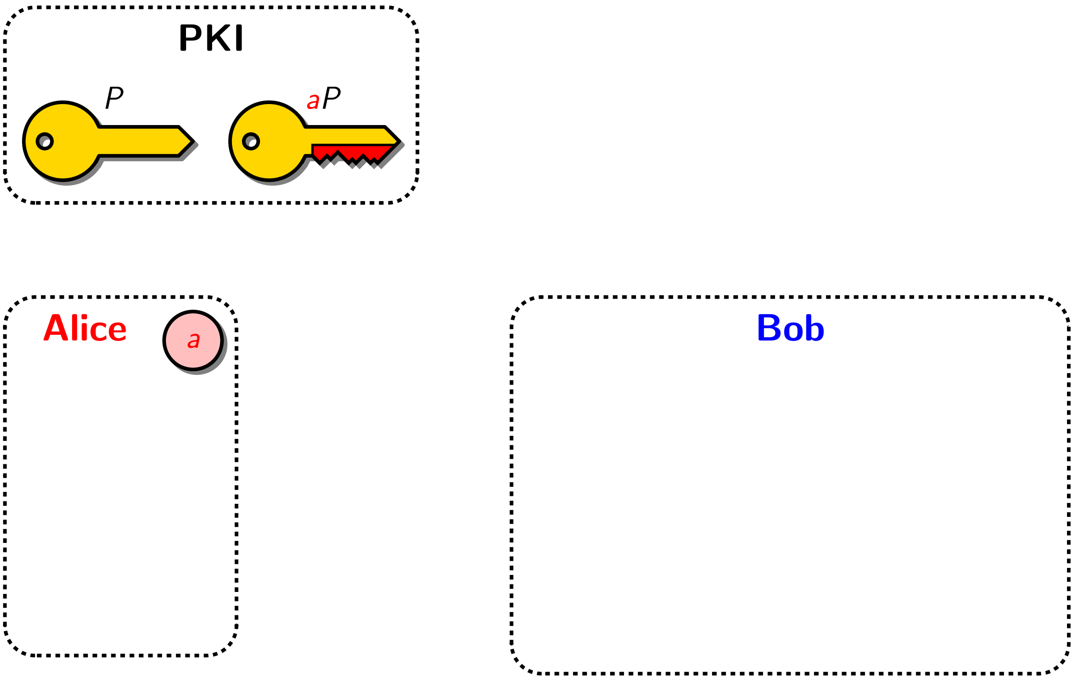
- One-round three-party key agreement (Joux, 2000)

- Identity-based encryption
  - Boneh–Franklin, 2001
  - Sakai–Kasahara, 2001

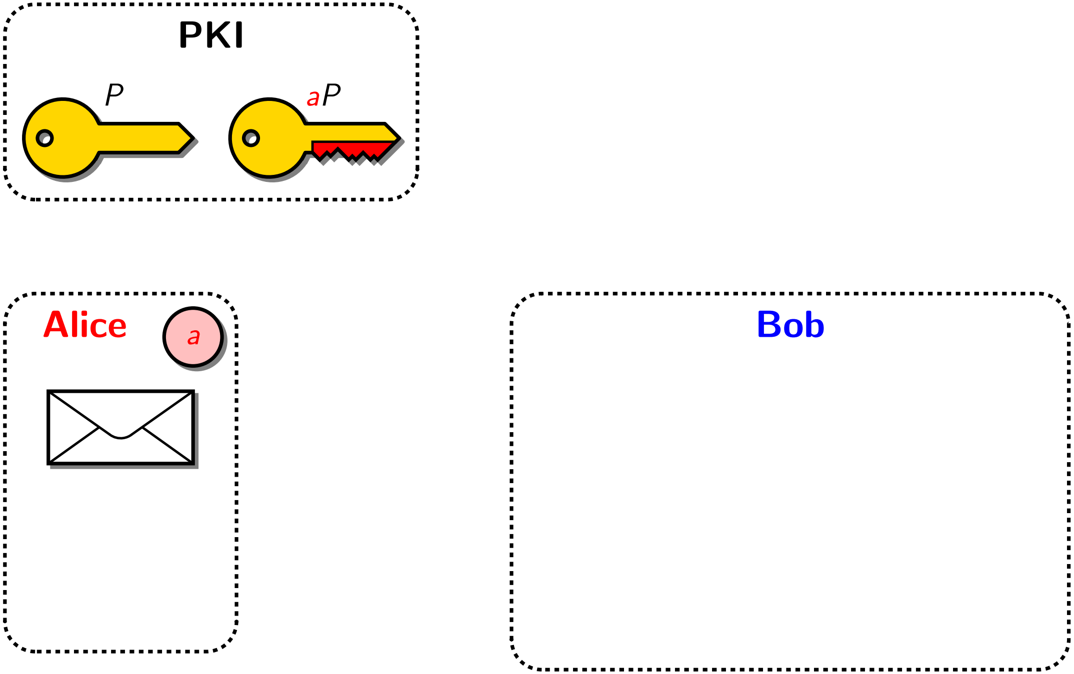
- Short digital signatures
  - Boneh–Lynn–Shacham, 2001
  - Zang–Safavi-Naini–Susilo, 2004

- ...

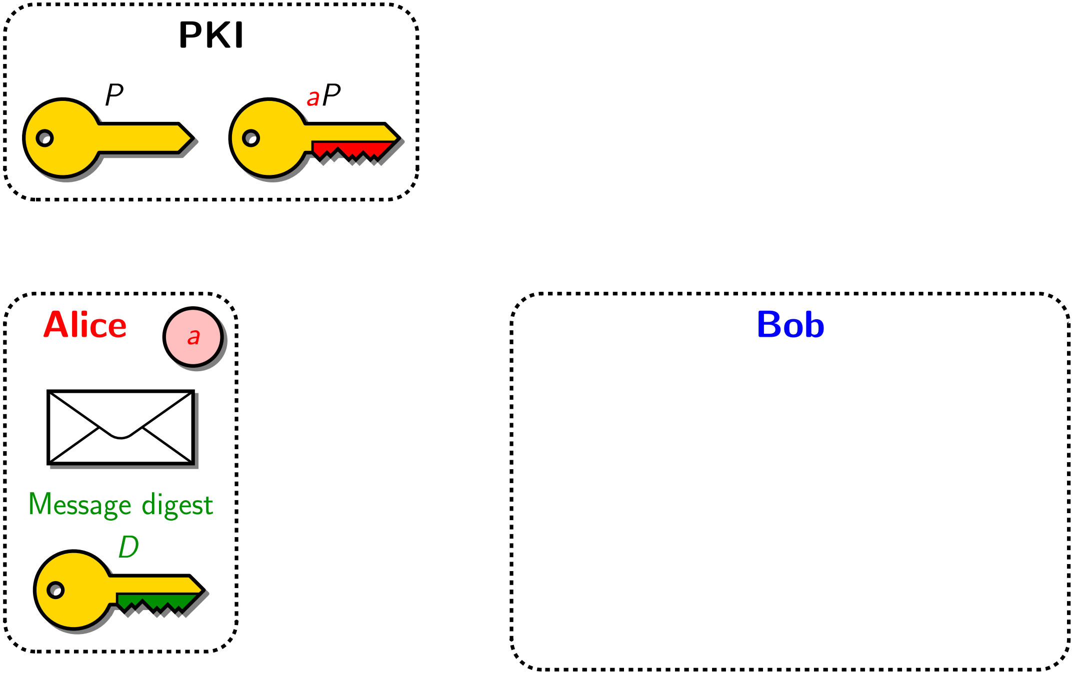
Short signature (Boneh, Lynn & Shacham, 2001)



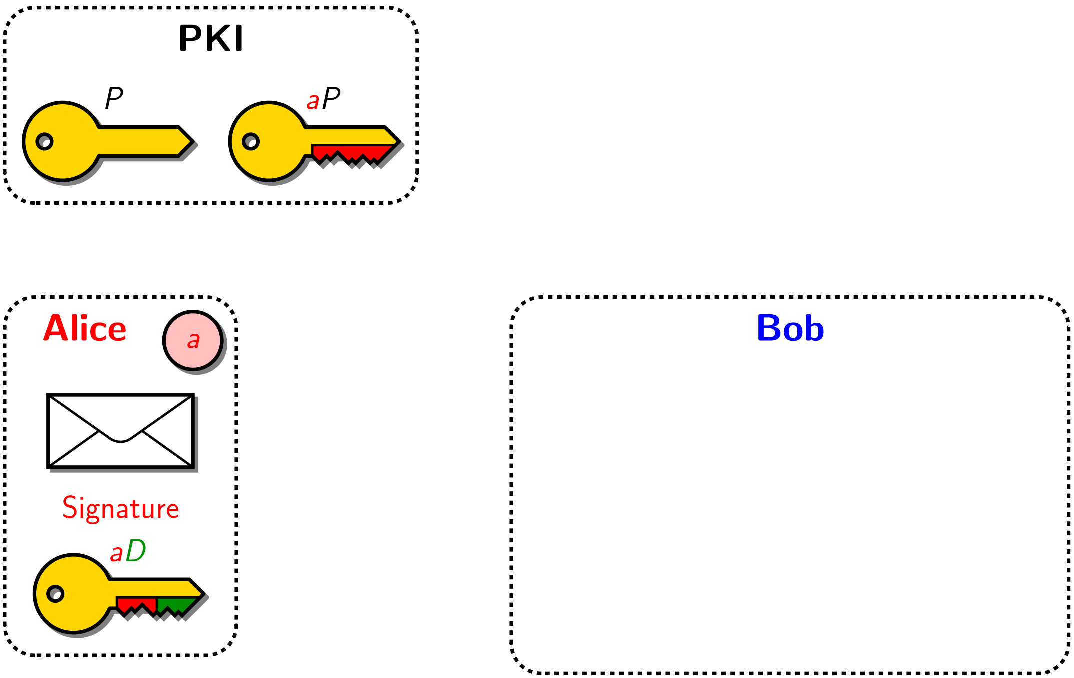
Short signature (Boneh, Lynn & Shacham, 2001)



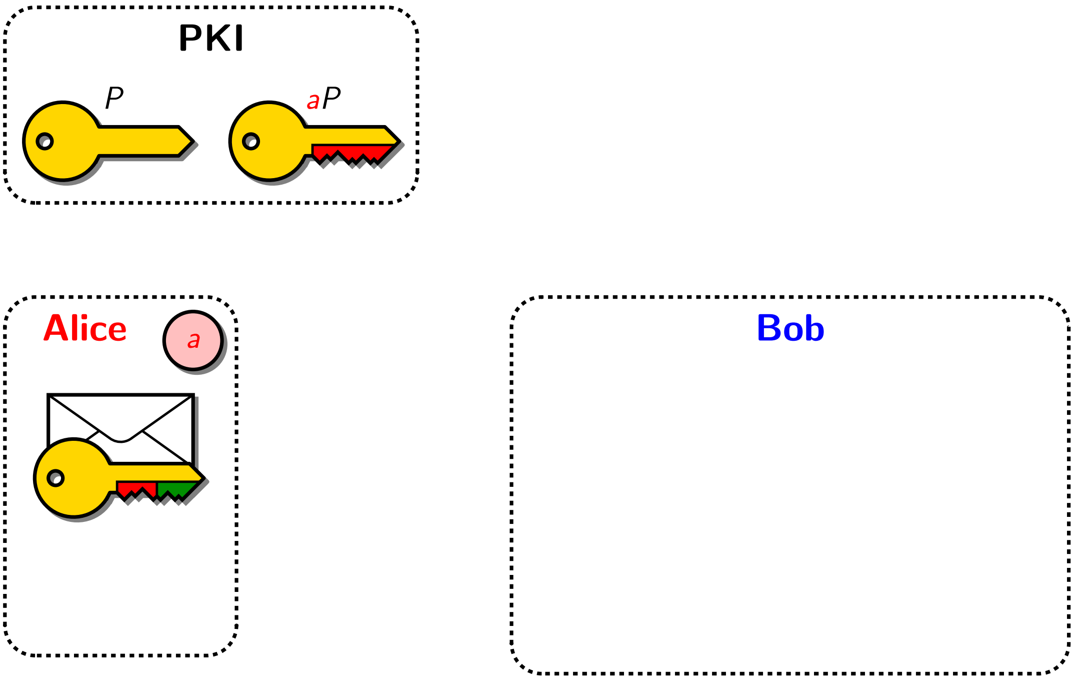
Short signature (Boneh, Lynn & Shacham, 2001)



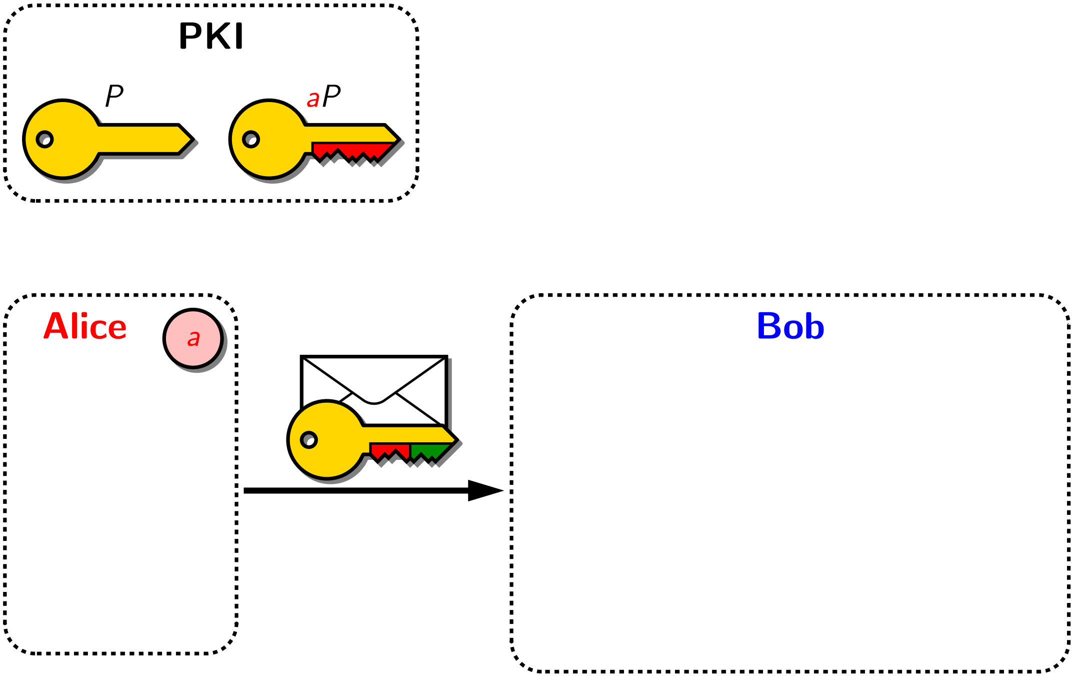
Short signature (Boneh, Lynn & Shacham, 2001)



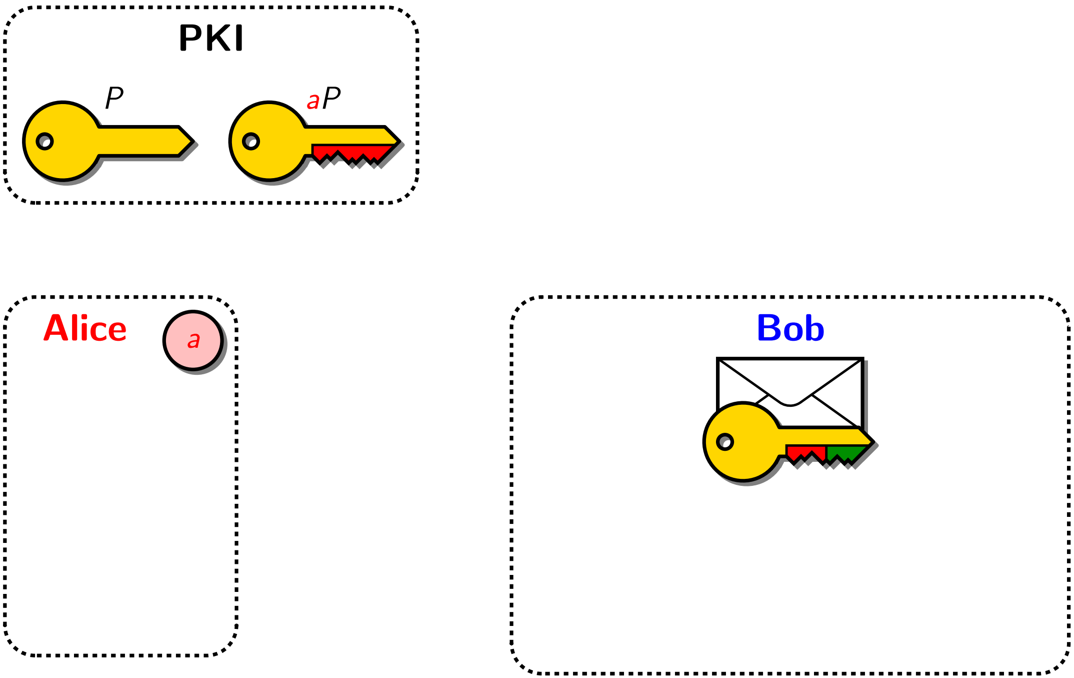
Short signature (Boneh, Lynn & Shacham, 2001)



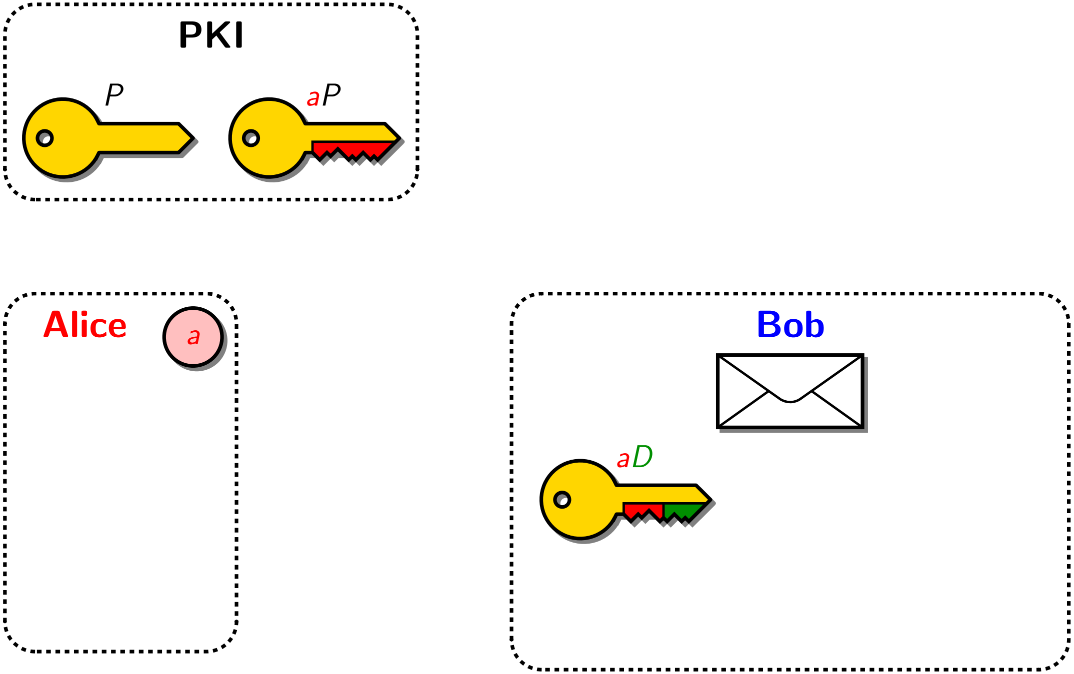
Short signature (Boneh, Lynn & Shacham, 2001)



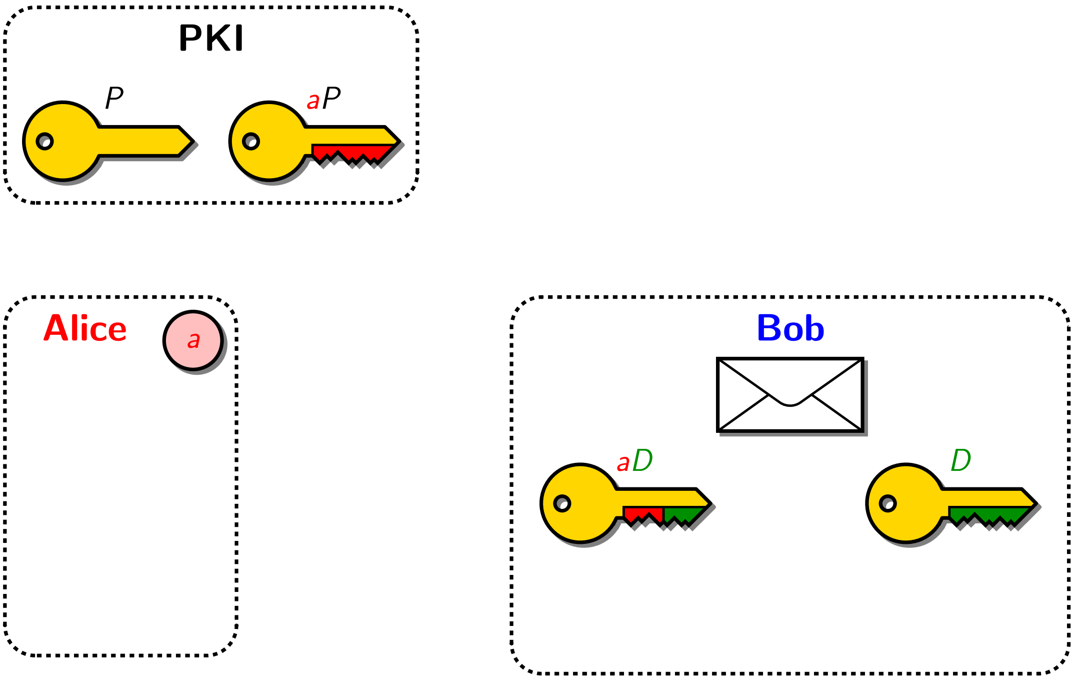
Short signature (Boneh, Lynn & Shacham, 2001)



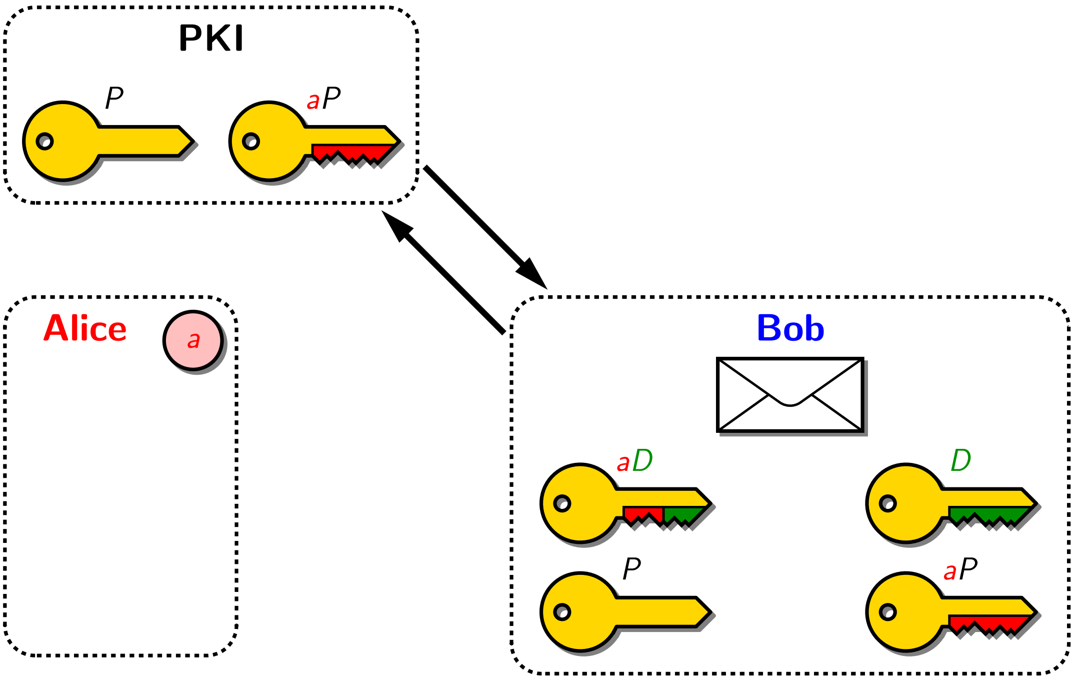
Short signature (Boneh, Lynn & Shacham, 2001)



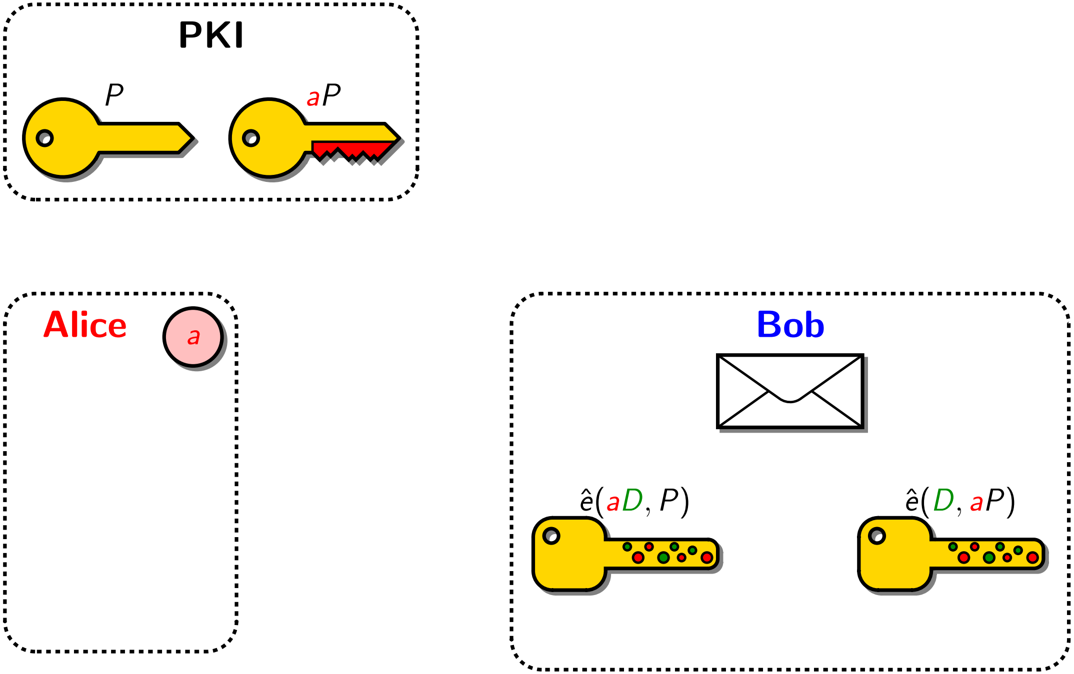
Short signature (Boneh, Lynn & Shacham, 2001)



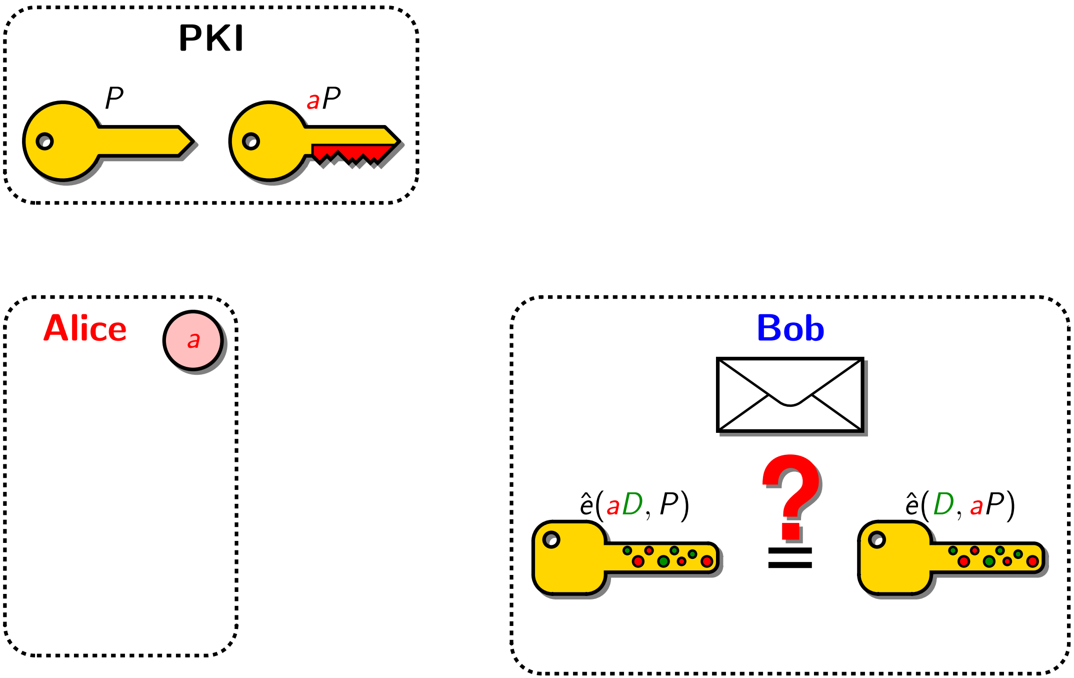
Short signature (Boneh, Lynn & Shacham, 2001)



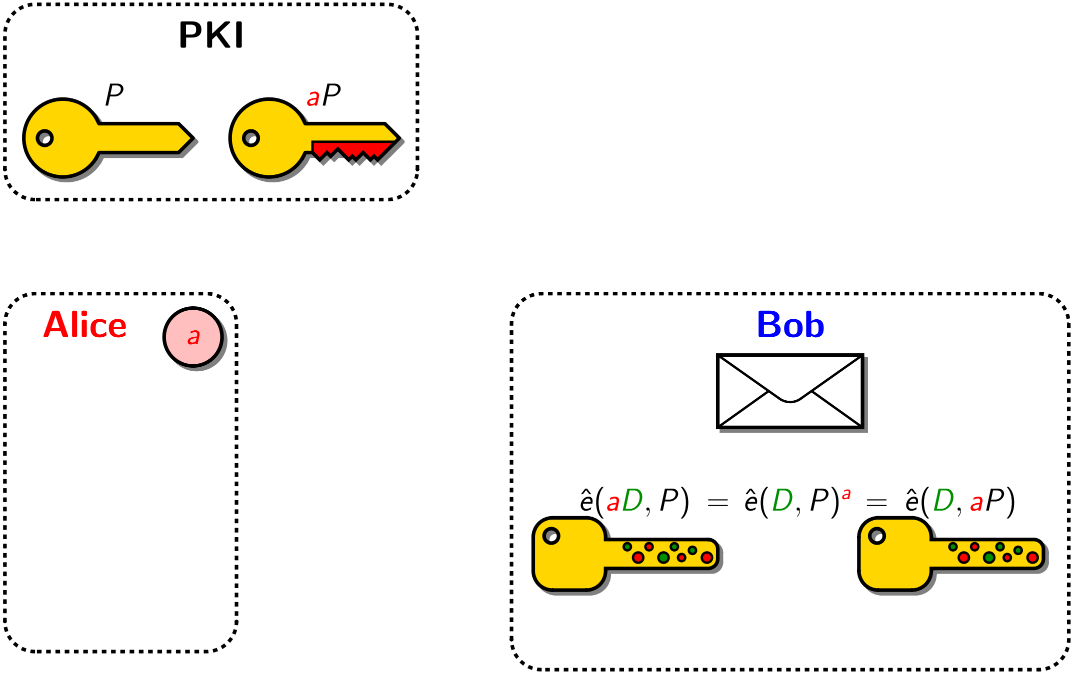
Short signature (Boneh, Lynn & Shacham, 2001)



Short signature (Boneh, Lynn & Shacham, 2001)



Short signature (Boneh, Lynn & Shacham, 2001)



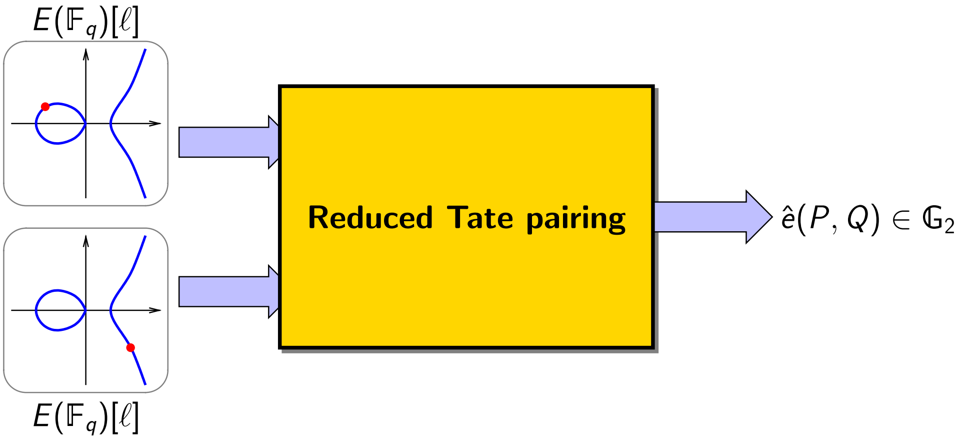
Context

$P \in \mathbb{G}_1$
$Q \in \mathbb{G}_1$
Bilinear pairing ê
$\hat{e}(P, Q) \in \mathbb{G}_2$

Context

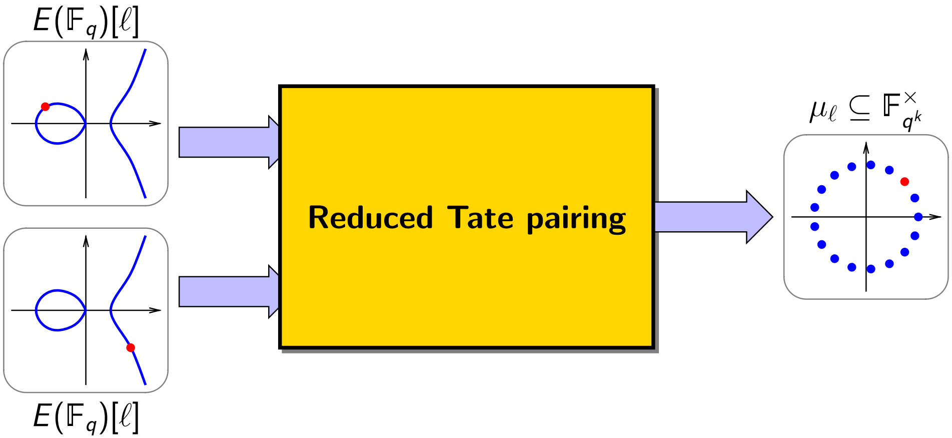
- Reduced Tate pairing

Context



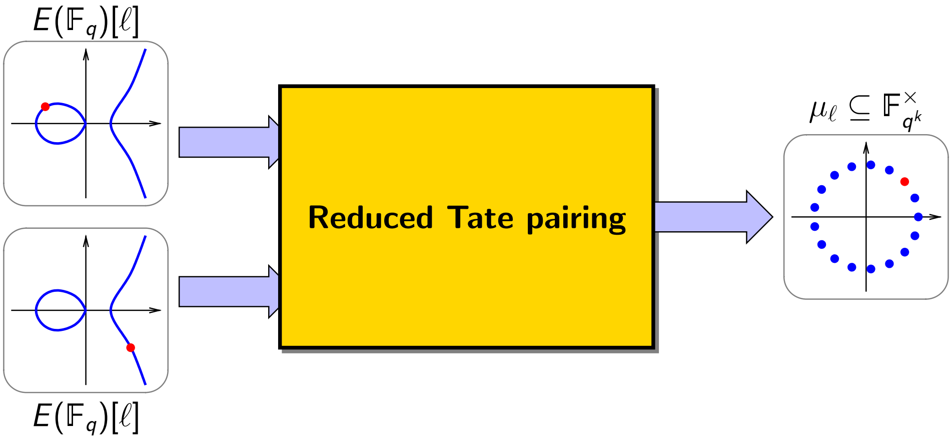
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_q)[\ell]$

Context



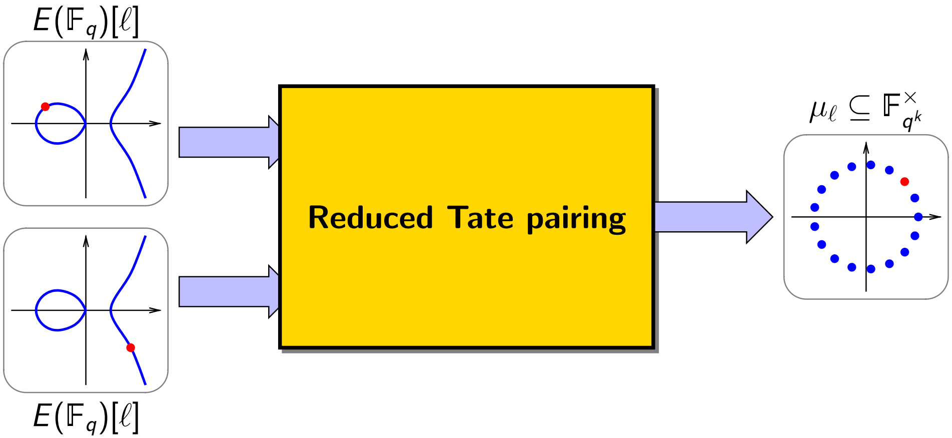
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_q)[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{q^k}^\times$

Context



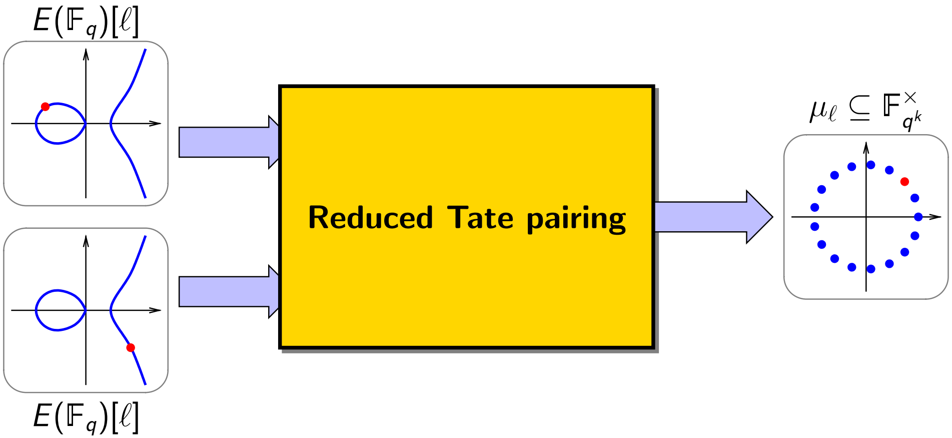
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_q)[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{q^k}^\times$
- Which elliptic curves?

Context



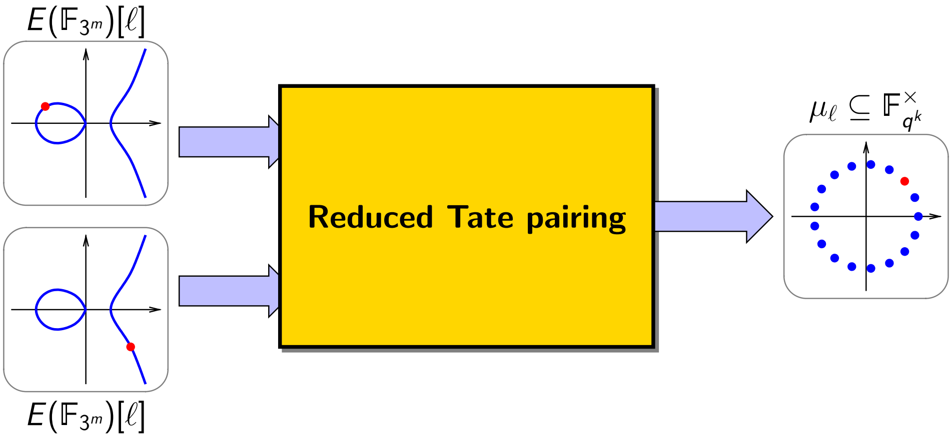
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_q)[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{q^k}^\times$
- Which elliptic curves? Supersingular curves
  - easier arithmetic on the curve
  - lower security

Context



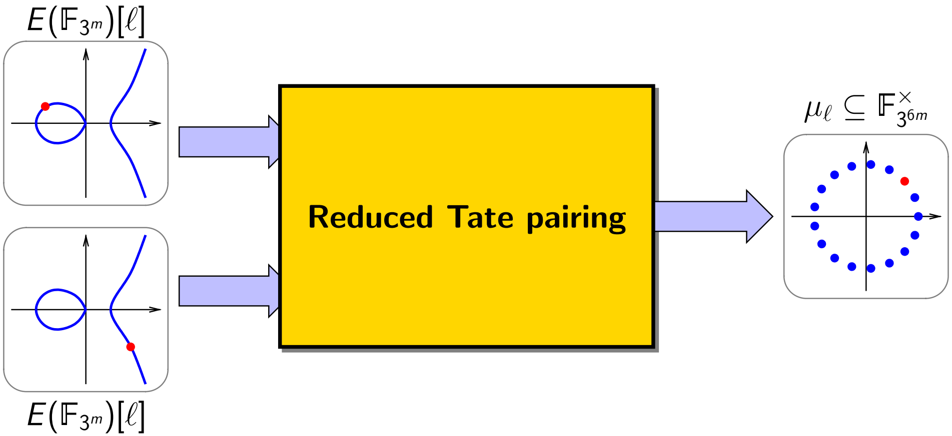
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_q)[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{q^k}^\times$
- Which elliptic curves? Supersingular curves
  - easier arithmetic on the curve
  - lower security $\Rightarrow$ characteristic 3 ($k = 6$)

Context



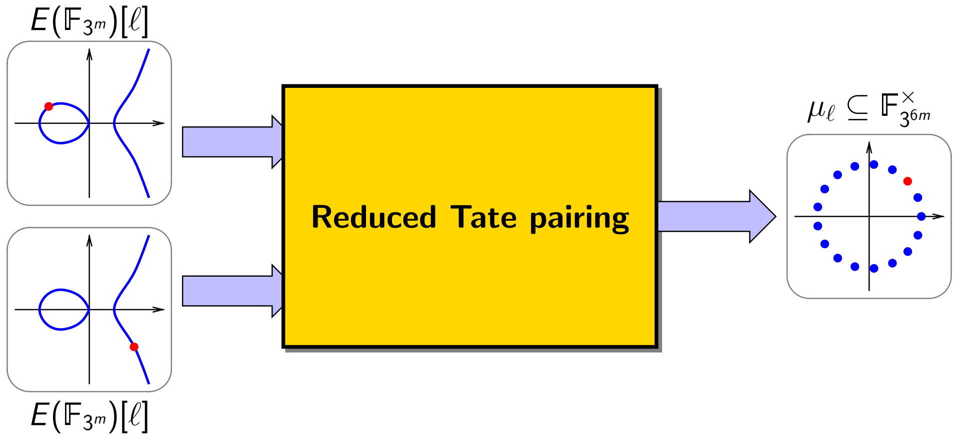
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_{3^m})[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{q^k}^\times$
- Which elliptic curves? Supersingular curves
  - easier arithmetic on the curve
  - lower security $\Rightarrow$ characteristic 3 ($k = 6$)

Context



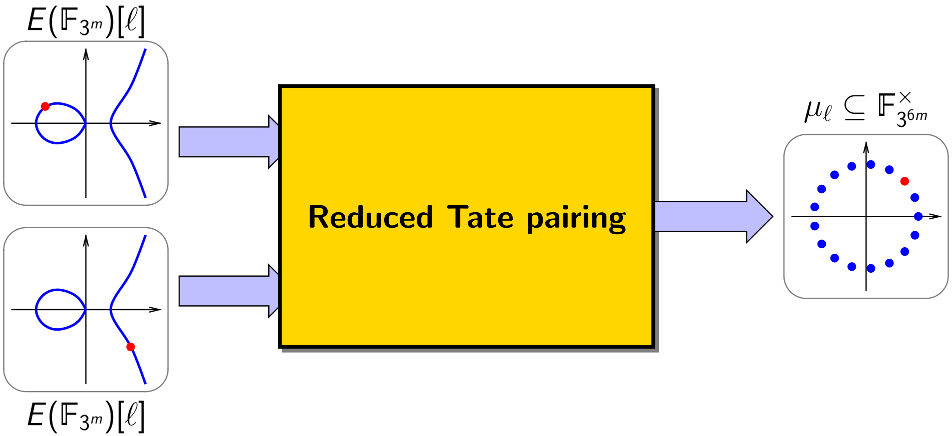
- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_{3^m})[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{3^{6m}}^\times$
- Which elliptic curves? Supersingular curves
  - easier arithmetic on the curve
  - lower security $\Rightarrow$ characteristic 3 ($k = 6$)

Context



- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_{3^m})[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}_{3^{6m}}^\times$
- Which elliptic curves? Supersingular curves
  - easier arithmetic on the curve
  - lower security $\Rightarrow$ characteristic 3 ($k = 6$)
- Need for a dedicated hardware accelerator
  - area optimized (embedded systems, RFID, ...)
  - speed optimized (bank servers, ...)

Context



- Reduced Tate pairing
  - input: two points P and Q in $E(\mathbb{F}_{3^m})[\ell]$
  - output: an ℓ-th root of unity in the extension $\mathbb{F}^\times_{3^{6m}}$
- Which elliptic curves? Supersingular curves
  - easier arithmetic on the curve
  - lower security $\Rightarrow$ characteristic 3 ($k = 6$)
- Need for a dedicated hardware accelerator
  - area optimized (embedded systems, RFID, ...)
  - speed optimized (bank servers, ...)

Outline of the talk

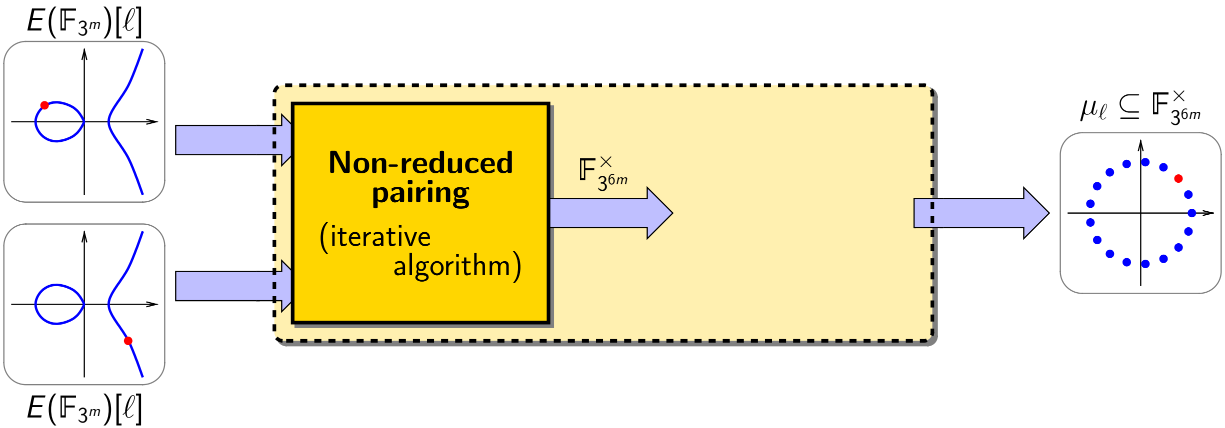
- Pairing-based cryptography
- Hardware accelerator for the Tate pairing
- Implementation results
- Concluding thoughts

Reduced Tate pairing

Reduced Tate pairing

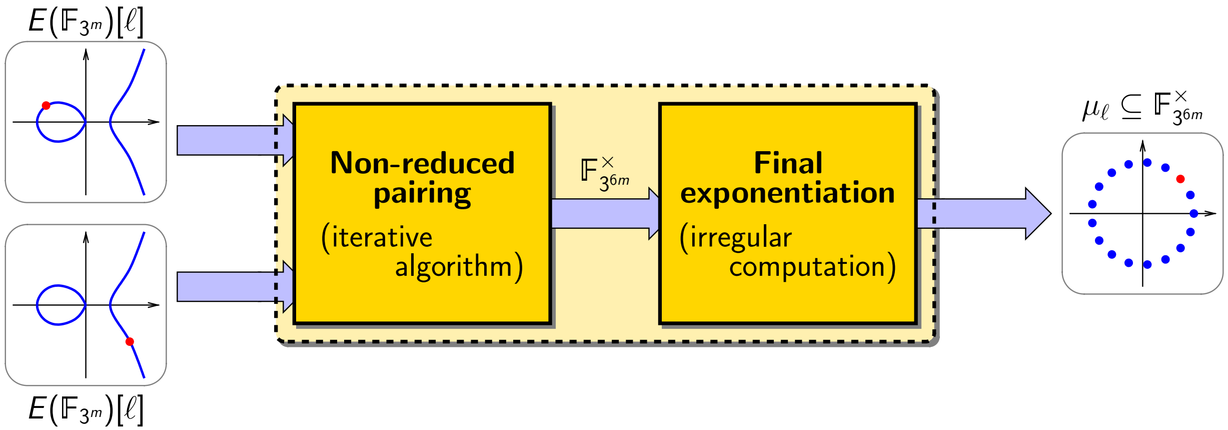
- Two very different steps

Reduced Tate pairing



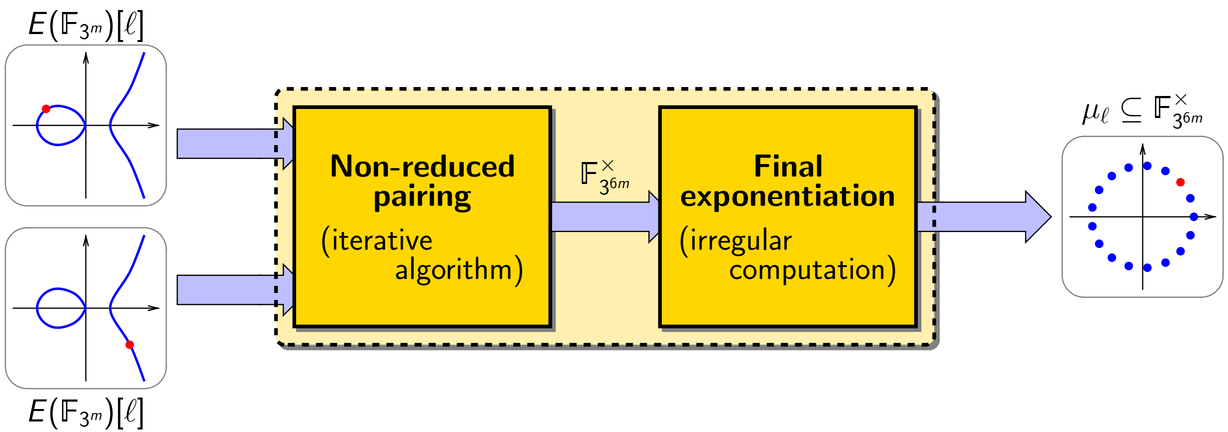
- Two very different steps

Reduced Tate pairing



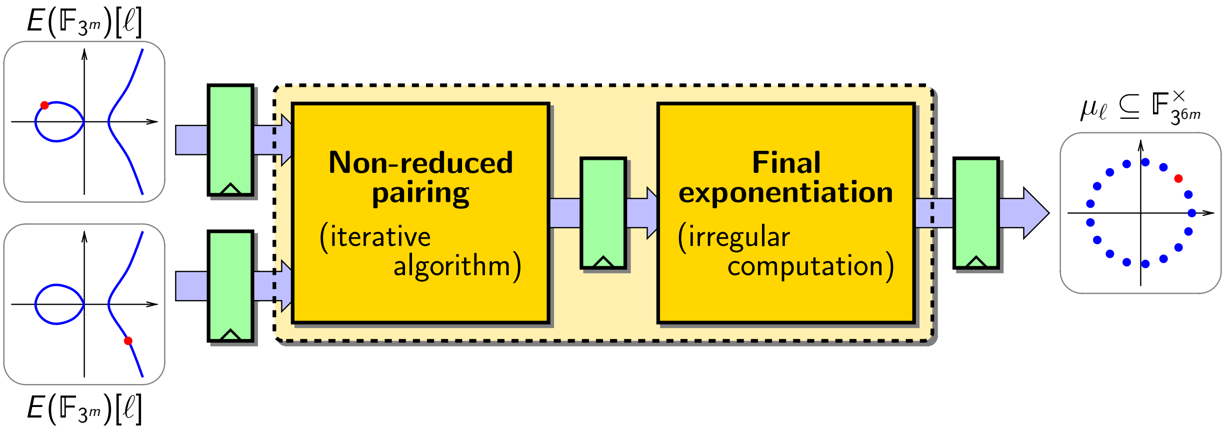
- Two very different steps

Reduced Tate pairing



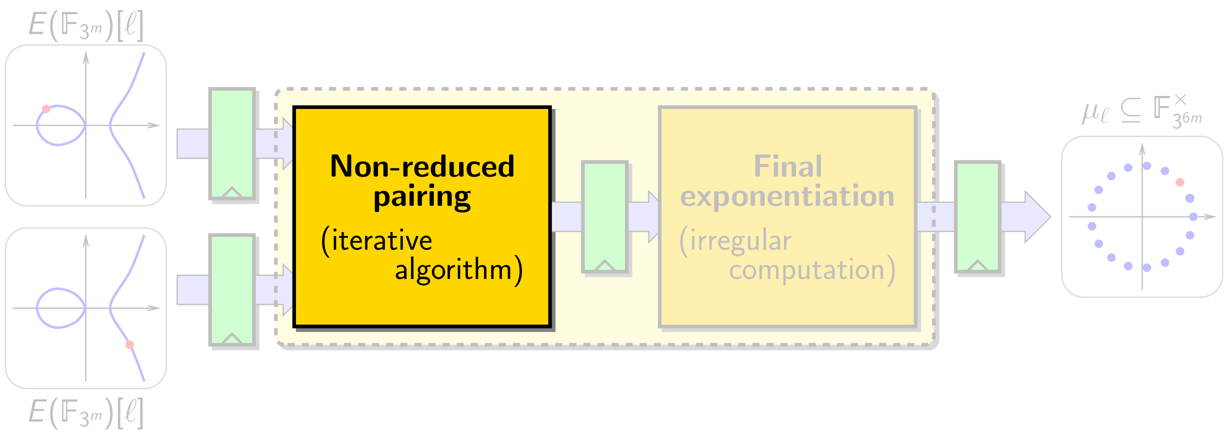
- Two very different steps
- Idea: use two distinct coprocessors

Reduced Tate pairing



- Two very different steps
- Idea: use two distinct coprocessors
  - pipeline the two computations
  - balance the latencies

Reduced Tate pairing



- Two very different steps
- Idea: use two distinct coprocessors
  - pipeline the two computations
  - balance the latencies

Computing the non-reduced pairing

- η_T pairing: shorter loop

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

$$x_P \leftarrow \sqrt[3]{x_P} \quad ; \quad y_P \leftarrow \sqrt[3]{y_P}$$
$$x_Q \leftarrow x_Q^3 \quad ; \quad y_Q \leftarrow y_Q^3$$

$$t \leftarrow x_P + x_Q \qquad u \leftarrow y_P y_Q$$
$$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$$

$$R \leftarrow R \cdot S$$

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  ① update of point coordinates

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

① $x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ $2\ \sqrt[3]{\cdot}$

$x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$ $2\ (\cdot)^3$

$t \leftarrow x_P + x_Q \qquad u \leftarrow y_P y_Q$

$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$

$R \leftarrow R \cdot S$

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  - ① update of point coordinates
  - ② computation of line equation

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

① $x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ — $2\ \sqrt[3]{\cdot}$
$x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$ — $2\ (\cdot)^3$

② $t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$
$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$ — $2\times, 2+$

$R \leftarrow R \cdot S$

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  - ① update of point coordinates
  - ② computation of line equation
  - ③ accumulation of the new factor

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

①	$x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ $x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$		$2\ \sqrt[3]{\cdot}$ $2\ (\cdot)^3$
②	$t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$ $S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$		$2\times, 2+$
③	$R \leftarrow R \cdot S$		$1\times\ (\mathbb{F}_{3^{6m}})$

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  ① update of point coordinates
  ② computation of line equation
  ③ accumulation of the new factor
- Multiplication is critical
- Fully parallel, pipelined multiplier over $\mathbb{F}_{3^m}$

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

① $x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ — $2\ \sqrt[3]{\cdot}$
$x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$ — $2\ (\cdot)^3$

② $t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$
$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$ — $2\times, 2+$

③ $R \leftarrow R \cdot S$ — $1\times\ (\mathbb{F}_{3^{6m}})$

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  ① update of point coordinates
  ② computation of line equation
  ③ accumulation of the new factor
- Multiplication is critical
- Fully parallel, pipelined multiplier over $\mathbb{F}_{3^m}$
- Sparse multiplication over $\mathbb{F}_{3^{6m}}$

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

① $x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ — $2\ \sqrt[3]{\cdot}$
$x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$ — $2\ (\cdot)^3$

② $t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$ — $2\times, 2+$
$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$

③ $R \leftarrow R \cdot S$ — $1\times\ (\mathbb{F}_{3^{6m}})$

end for

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  ① update of point coordinates
  ② computation of line equation
  ③ accumulation of the new factor
- Multiplication is critical
- Fully parallel, pipelined multiplier over $\mathbb{F}_{3^m}$

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

① $x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ — $2\ \sqrt[3]{\cdot}$
$x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$ — $2\ (\cdot)^3$

② $t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$ — $2\times, 2+$
$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$

③ $R \leftarrow R \cdot S$ — $12\times, 59+$

end for

- Sparse multiplication over $\mathbb{F}_{3^{6m}}$
  - $12\times$ and $59+$ over $\mathbb{F}_{3^m}$ (Gorla *et al.*, SAC 2007)

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  ① update of point coordinates
  ② computation of line equation
  ③ accumulation of the new factor
- Multiplication is critical
- Fully parallel, pipelined multiplier over $\mathbb{F}_{3^m}$

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

①	$x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ $x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$	$2\ \sqrt[3]{\cdot}$ $2\ (\cdot)^3$
②	$t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$ $S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$	$2\times, 2+$
③	$R \leftarrow R \cdot S$	$15\times, 29+$

end for

- Sparse multiplication over $\mathbb{F}_{3^{6m}}$
  - $12\times$ and $59+$ over $\mathbb{F}_{3^m}$ (Gorla *et al.*, SAC 2007)
  - $15\times$ and $29+$ over $\mathbb{F}_{3^m}$ (Beuchat *et al.*, ARITH 18)

Computing the non-reduced pairing

- η_T pairing: shorter loop
- Based on Miller's algorithm:
  ① update of point coordinates
  ② computation of line equation
  ③ accumulation of the new factor
- Multiplication is critical
- Fully parallel, pipelined multiplier over $\mathbb{F}_{3^m}$

for $i \leftarrow 0$ **to** $(m-1)/2$ **do**

① $x_P \leftarrow \sqrt[3]{x_P}$; $y_P \leftarrow \sqrt[3]{y_P}$ — $2\ \sqrt[3]{\cdot}$
$x_Q \leftarrow x_Q^3$; $y_Q \leftarrow y_Q^3$ — $2\ (\cdot)^3$

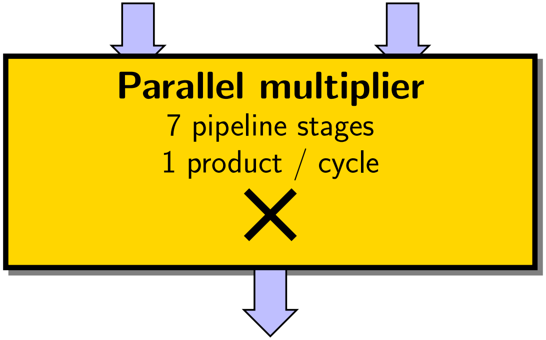
② $t \leftarrow x_P + x_Q$; $u \leftarrow y_P y_Q$
$S \leftarrow -t^2 \pm u\sigma - t\rho - \rho^2$ — $2\times, 2+$

③ $R \leftarrow R \cdot S$ — $15\times, 29+$

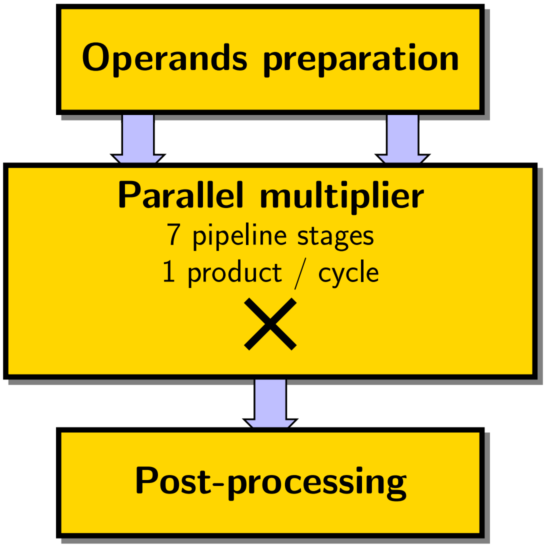
end for

- Sparse multiplication over $\mathbb{F}_{3^{6m}}$
  - $12\times$ and $59+$ over $\mathbb{F}_{3^m}$ (Gorla *et al.*, SAC 2007)
  - $15\times$ and $29+$ over $\mathbb{F}_{3^m}$ (Beuchat *et al.*, ARITH 18)
- Objective: keep the multiplier pipeline busy
  - 7-stage pipeline
  - one product per cycle
  - 17 cycles per iteration

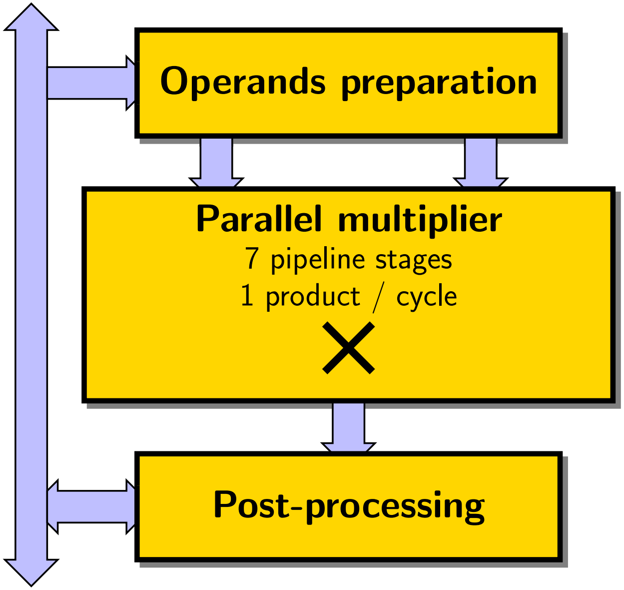
Coprocessor for the non-reduced pairing



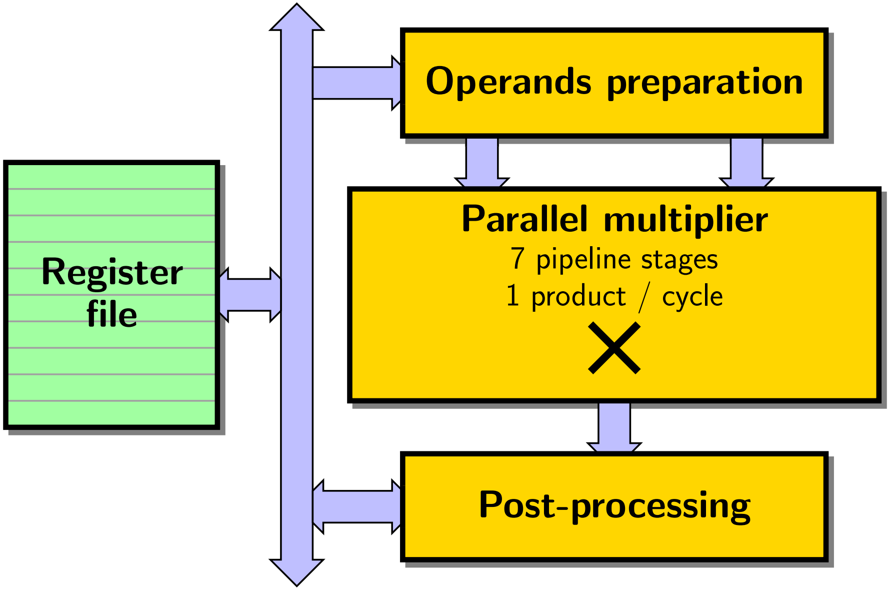
Coprocessor for the non-reduced pairing



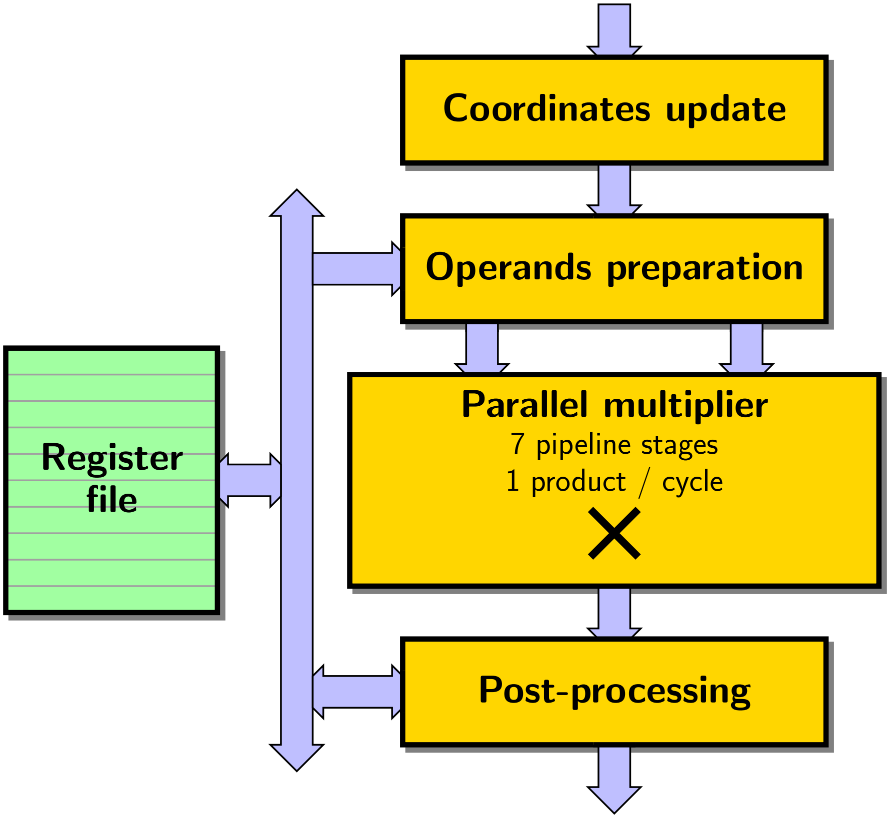
Coprocessor for the non-reduced pairing



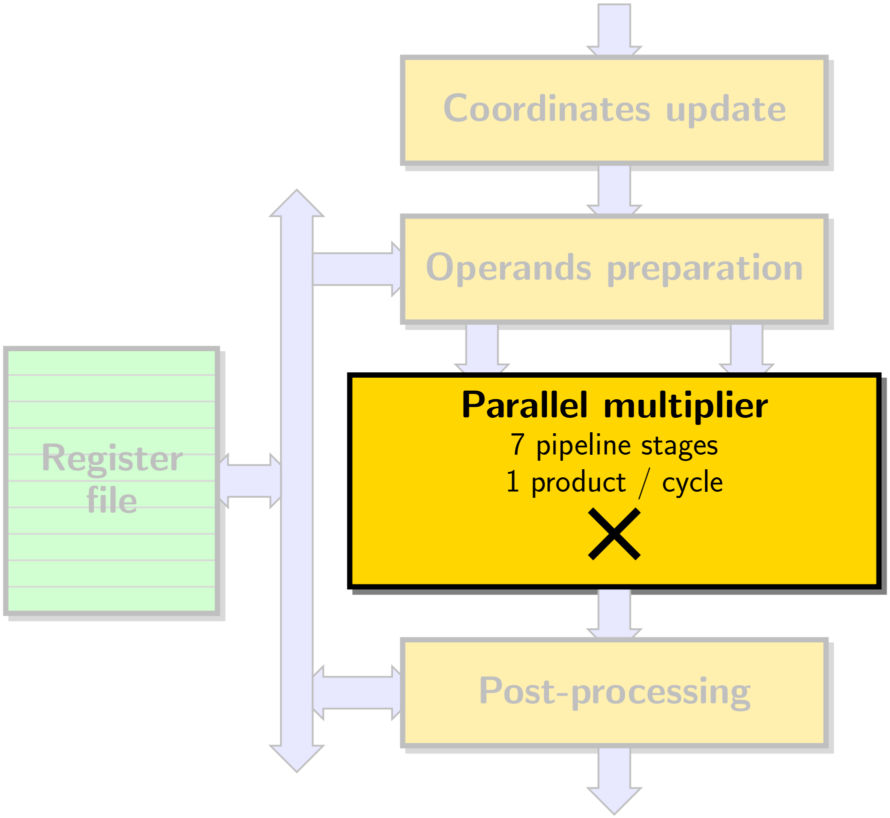
Coprocessor for the non-reduced pairing



Coprocessor for the non-reduced pairing



Coprocessor for the non-reduced pairing



Our parallel multiplier

- Polynomial basis:

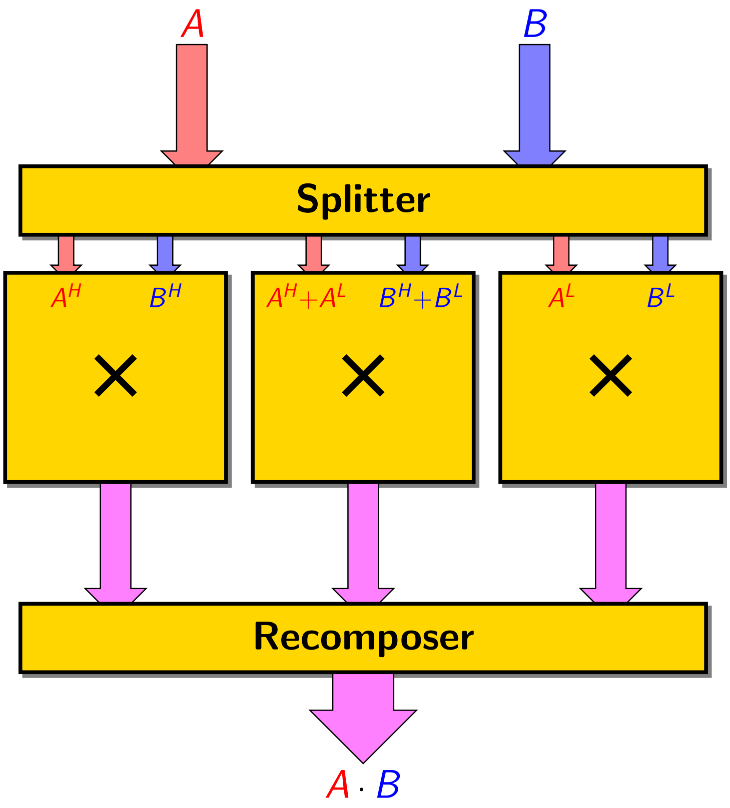
$$\mathbb{F}_{3^m} \cong \mathbb{F}_3[x]/(f(x))$$

Our parallel multiplier

- Polynomial basis:

$$\mathbb{F}_{3^m} \cong \mathbb{F}_3[x]/(f(x))$$

- Karatsuba scheme
- Fully parallel

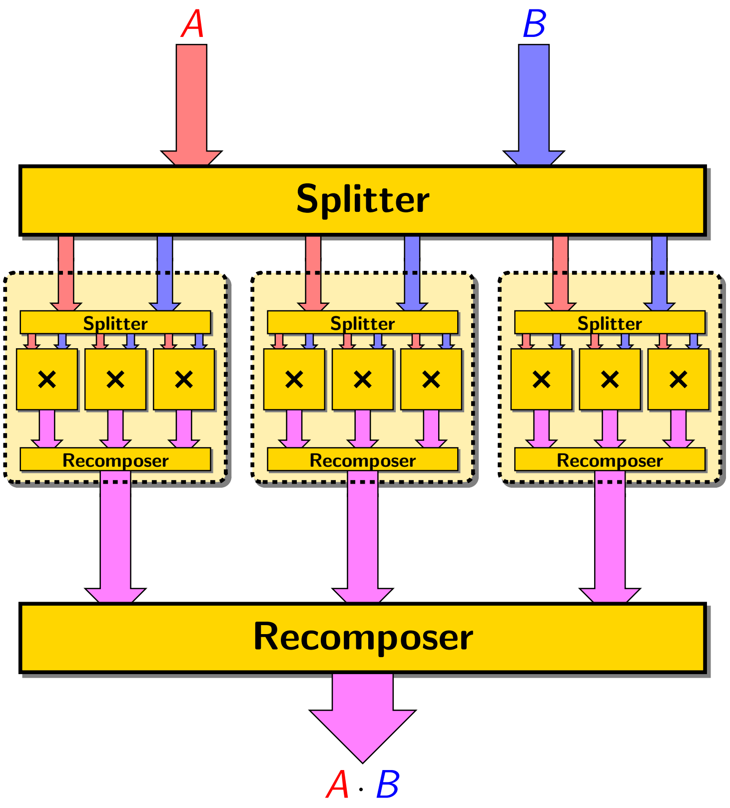


Our parallel multiplier

- Polynomial basis:

$$\mathbb{F}_{3^m} \cong \mathbb{F}_3[x]/(f(x))$$

- Karatsuba scheme
- Fully parallel
- Recursive scheme

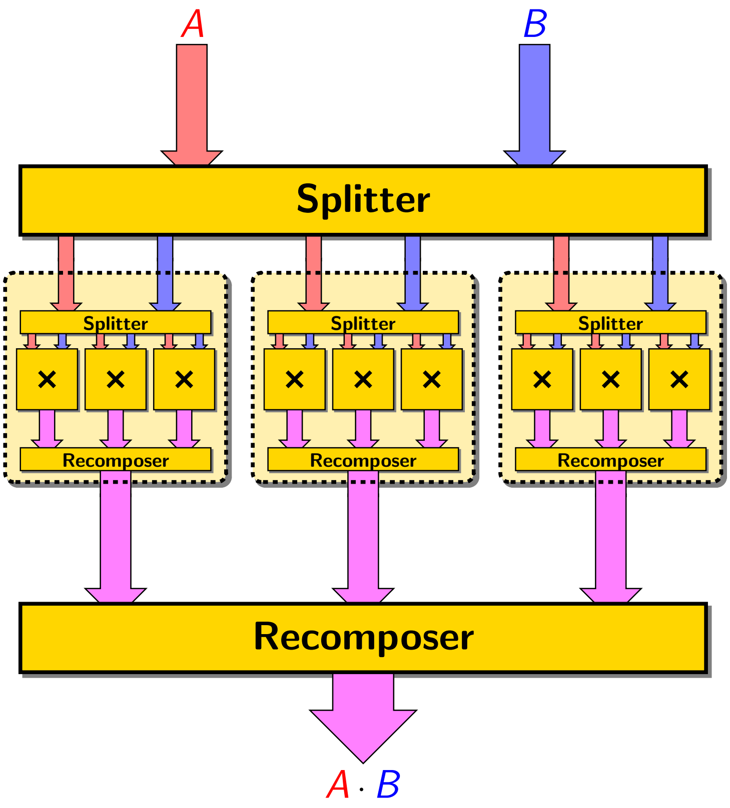


Our parallel multiplier

- Polynomial basis:

$$\mathbb{F}_{3^m} \cong \mathbb{F}_3[x]/(f(x))$$

- Karatsuba scheme
- Fully parallel
- Recursive scheme
- Some variations:
  - 3-way split
  - odd–even split
  - select the best method for each stage

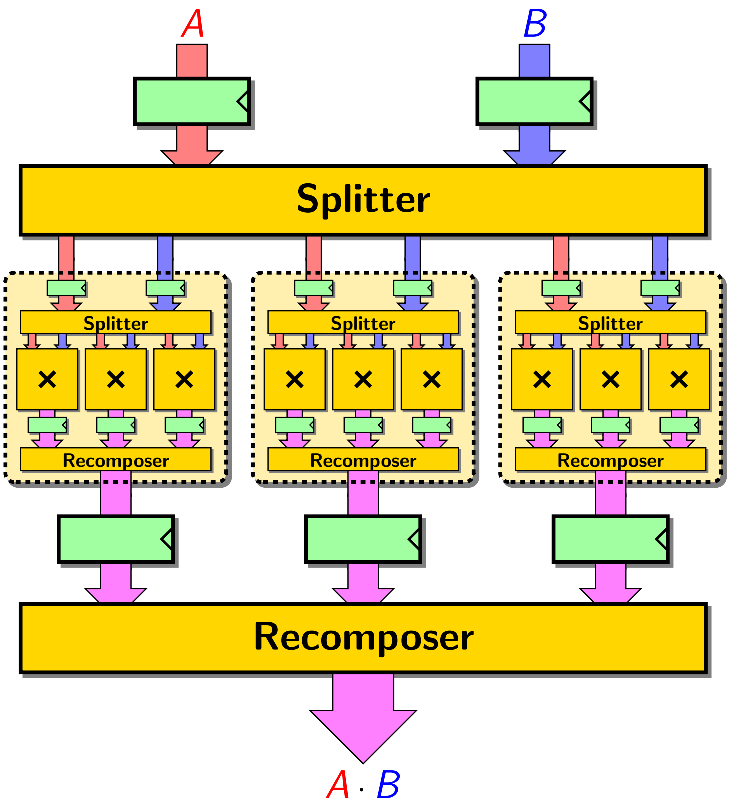


Our parallel multiplier

- Polynomial basis:

$$\mathbb{F}_{3^m} \cong \mathbb{F}_3[x]/(f(x))$$

- Karatsuba scheme
- Fully parallel
- Recursive scheme
- Some variations:
  - 3-way split
  - odd–even split
  - select the best method for each stage
- Pipelined: optional registers

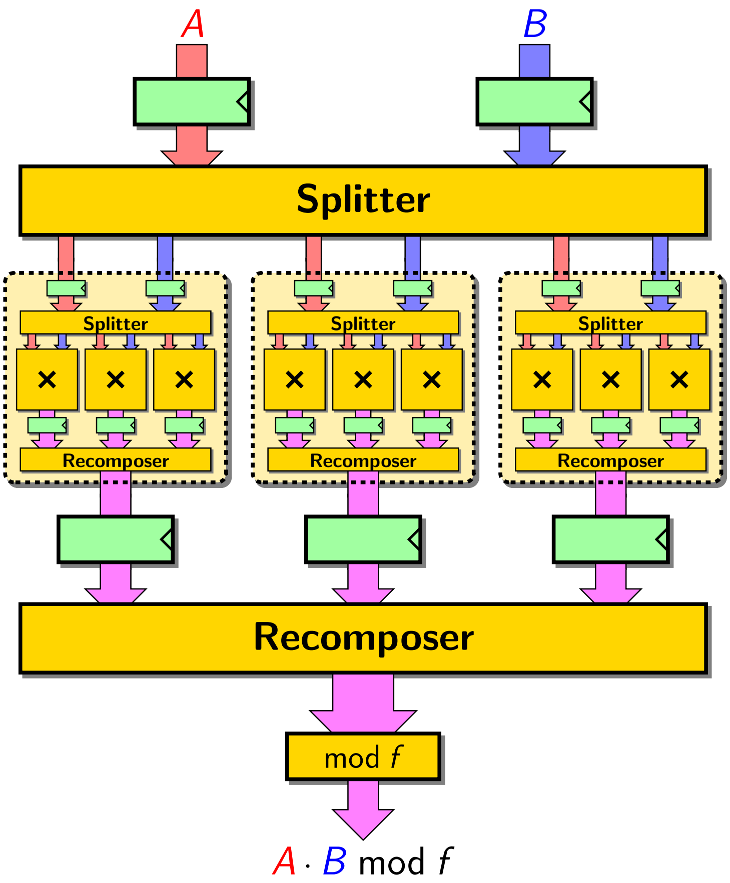


Our parallel multiplier

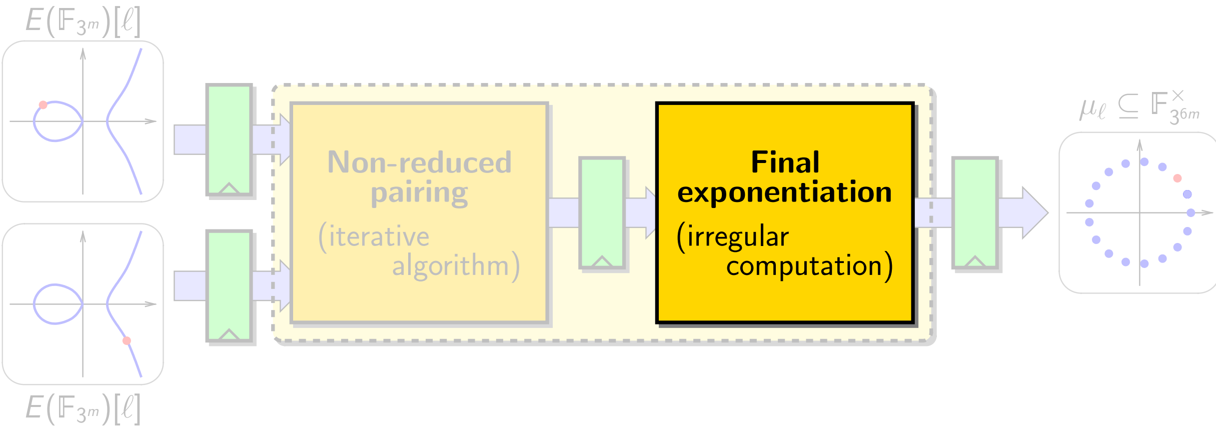
- Polynomial basis:

$$\mathbb{F}_{3^m} \cong \mathbb{F}_3[x]/(f(x))$$

- Karatsuba scheme
- Fully parallel
- Recursive scheme
- Some variations:
  - 3-way split
  - odd–even split
  - select the best method for each stage
- Pipelined: optional registers
- Final reduction modulo f



Final exponentiation



- Design rationale:
  - as small as possible
  - at least as fast as the computation of the non-reduced pairing

Coprocessor for the final exponentiation

- Highly sequential computation
- Very heterogeneous

Coprocessor for the final exponentiation

$$\left.\begin{array}{l}\blacktriangleright\ \text{Highly sequential computation} \\[1em] \blacktriangleright\ \text{Very heterogeneous}\end{array}\right\} \Rightarrow \begin{array}{c}\text{general-purpose} \\ \text{finite-field arithmetic} \\ \text{processor}\end{array}$$

Coprocessor for the final exponentiation

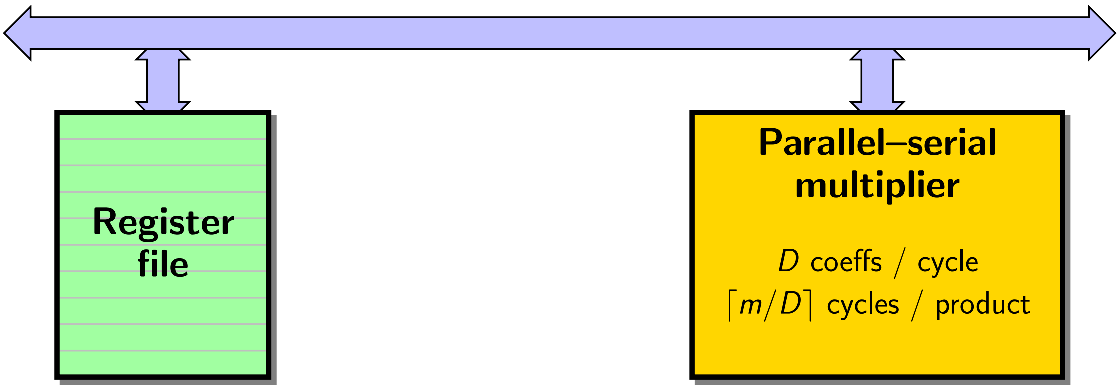
- Highly sequential computation
- Very heterogeneous

$\Rightarrow$ general-purpose finite-field arithmetic processor

Coprocessor for the final exponentiation

- Highly sequential computation
- Very heterogeneous

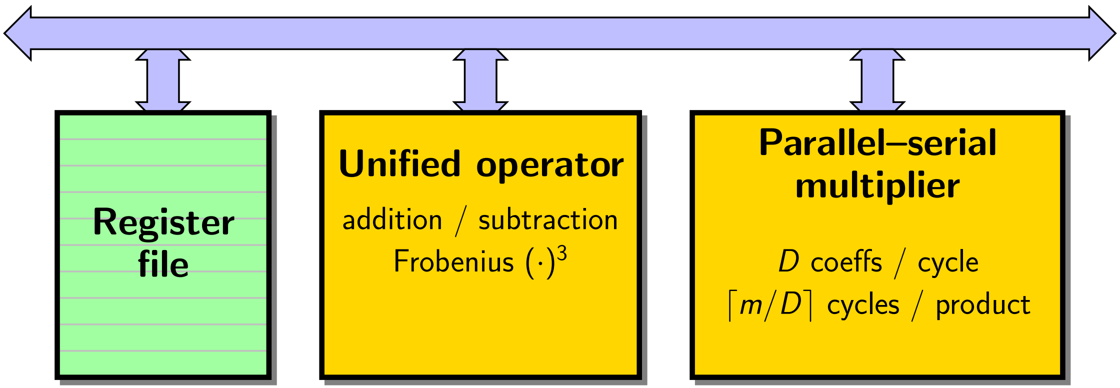
} $\Rightarrow$ general-purpose finite-field arithmetic processor



Coprocessor for the final exponentiation

- Highly sequential computation
- Very heterogeneous

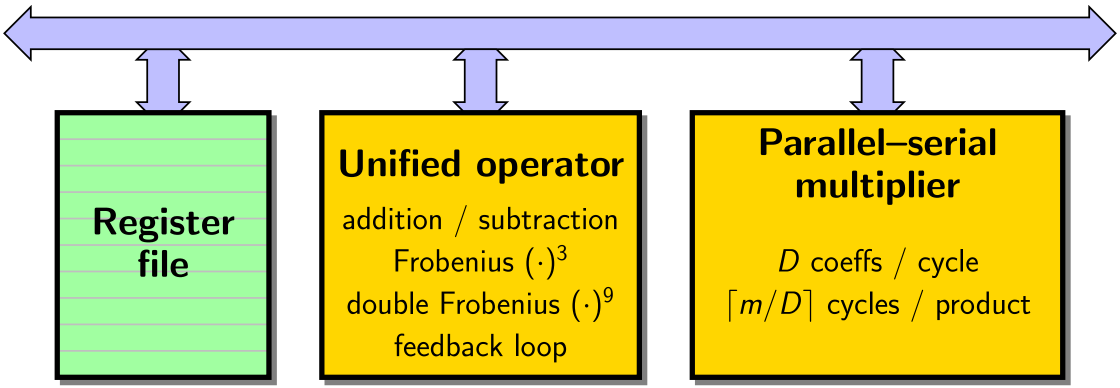
$\Rightarrow$ general-purpose finite-field arithmetic processor



Coprocessor for the final exponentiation

- Highly sequential computation
- Very heterogeneous

$\Rightarrow$ general-purpose finite-field arithmetic processor

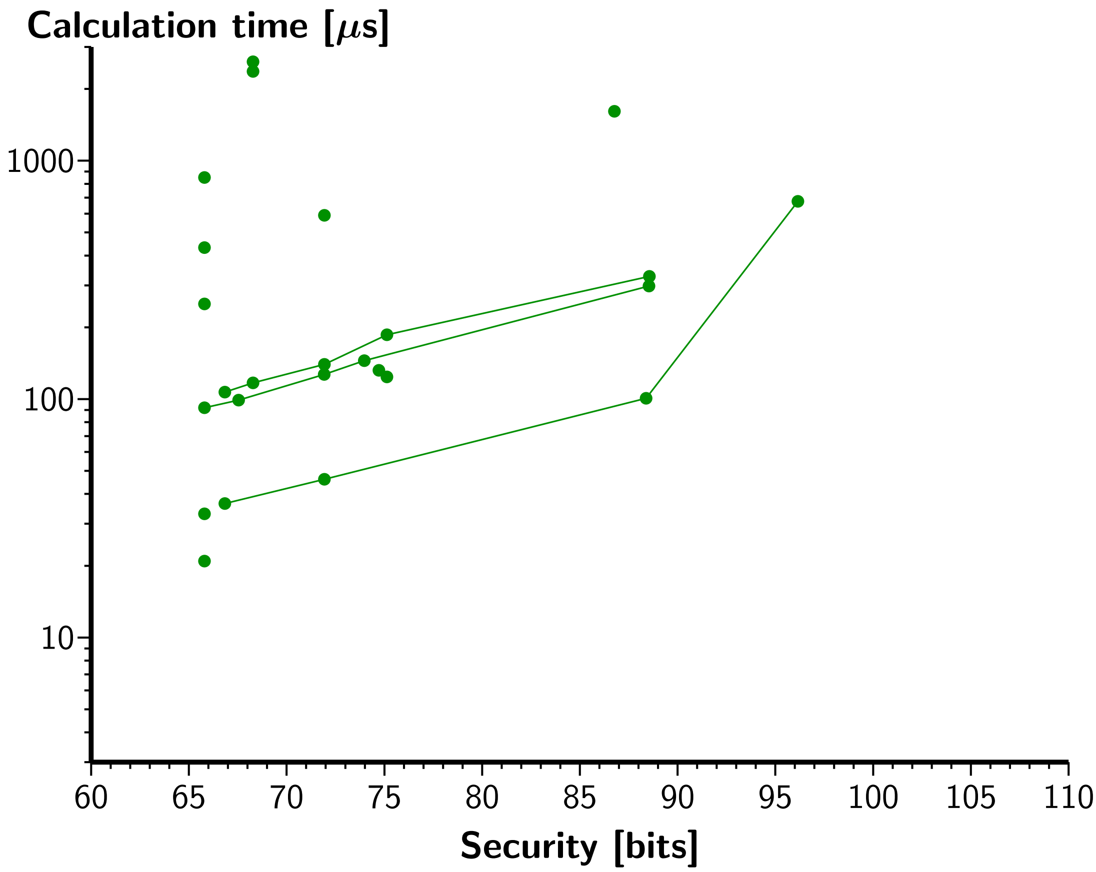


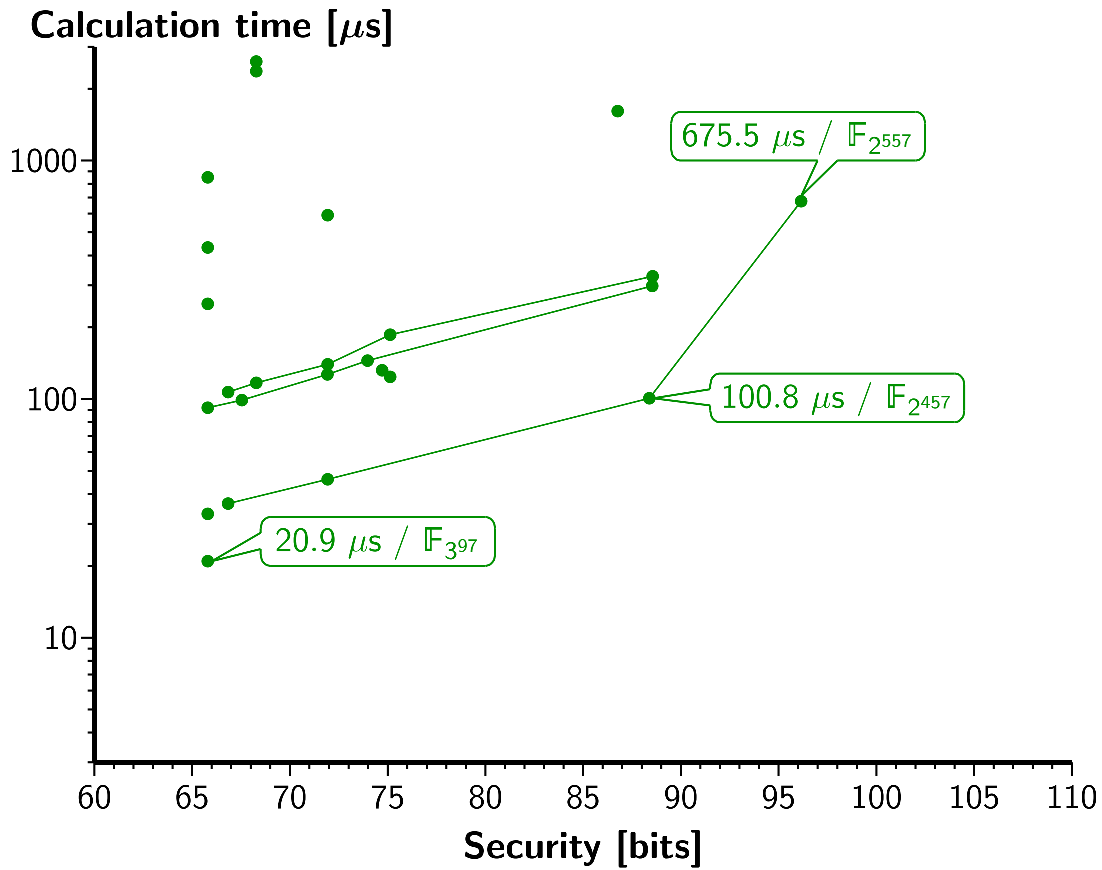
Outline of the talk

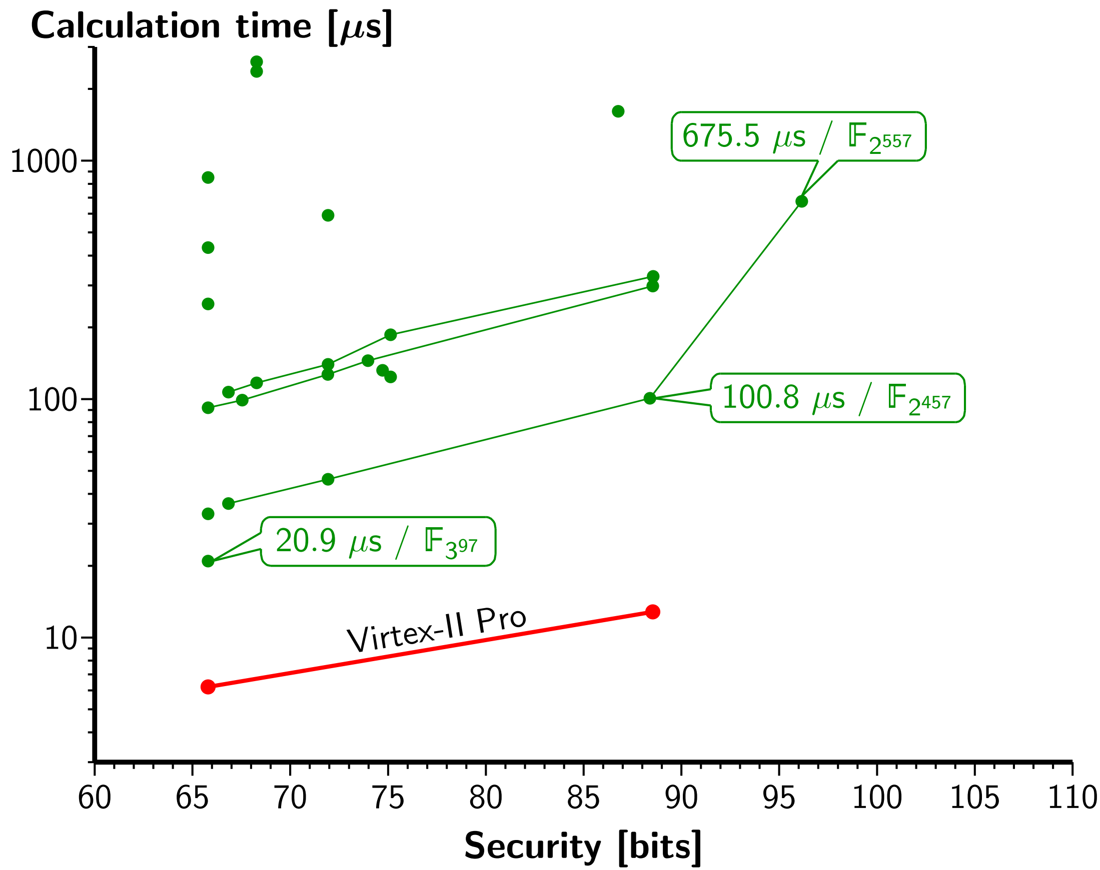
- Pairing-based cryptography
- Hardware accelerator for the Tate pairing
- Implementation results
- Concluding thoughts

Experimental setup

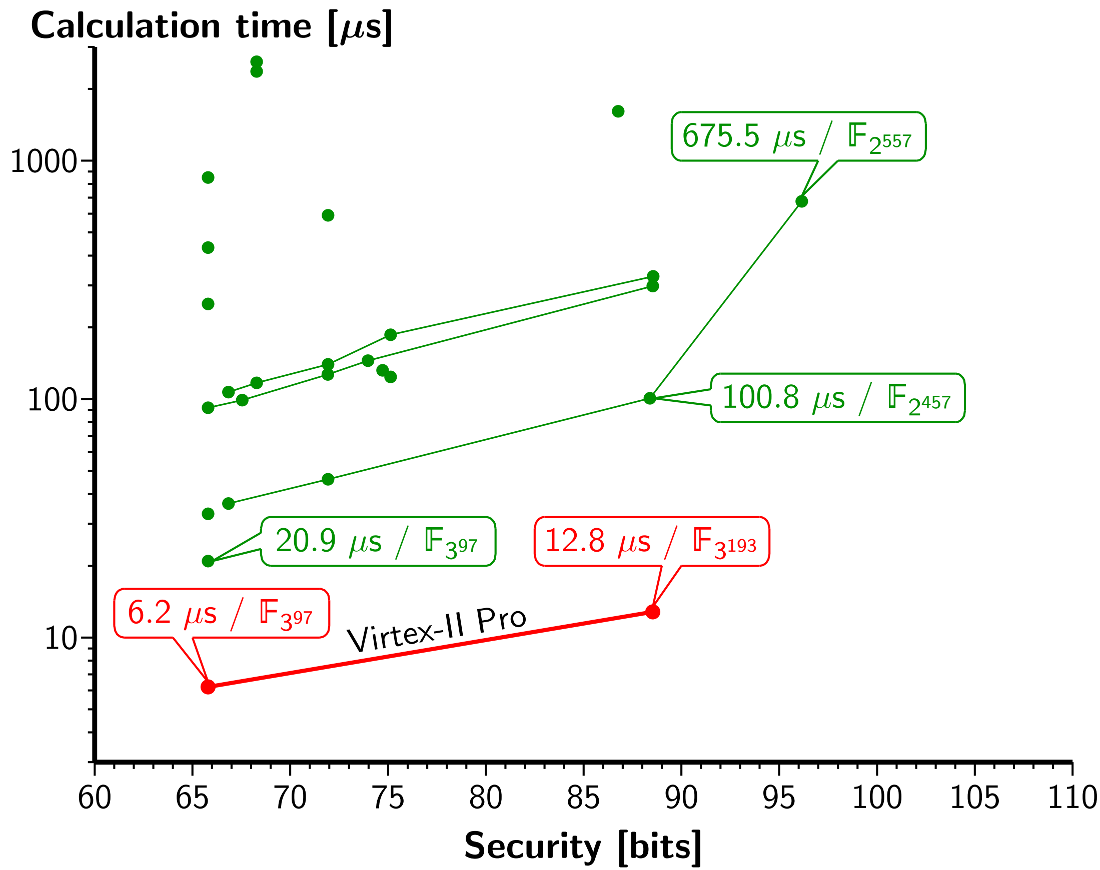
- Full Tate pairing computation:
  - non-reduced pairing and
  - final exponentiation
- Prototyped on Xilinx Virtex-II Pro and Virtex-4 LX FPGAs
- Post–place-and-route timing and area estimations

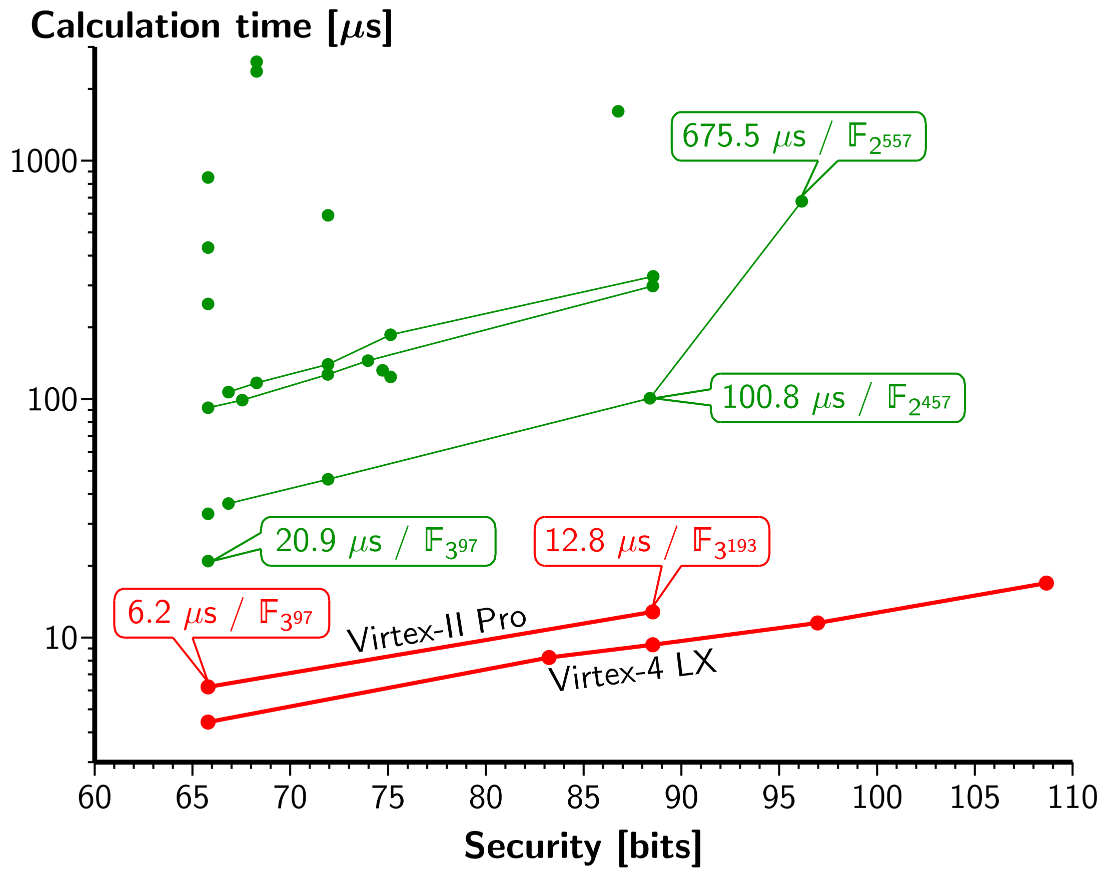

Calculation time [μs]
1000
100
10
60
65
70
75
80
85
90
95
100
105
110
Security [bits]


Calculation time [μs]
1000
100
10
675.5 μs / $\mathbb{F}_{2^{557}}$
100.8 μs / $\mathbb{F}_{2^{457}}$
20.9 μs / $\mathbb{F}_{3^{97}}$
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

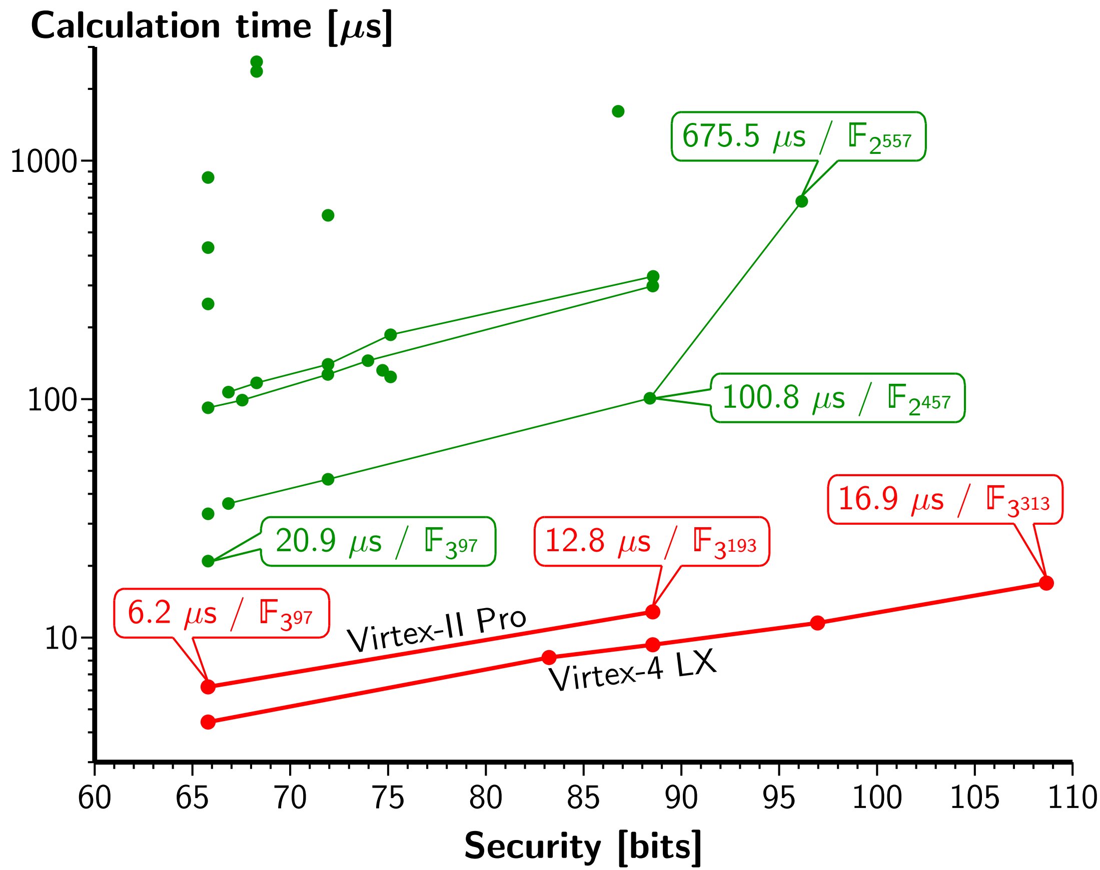


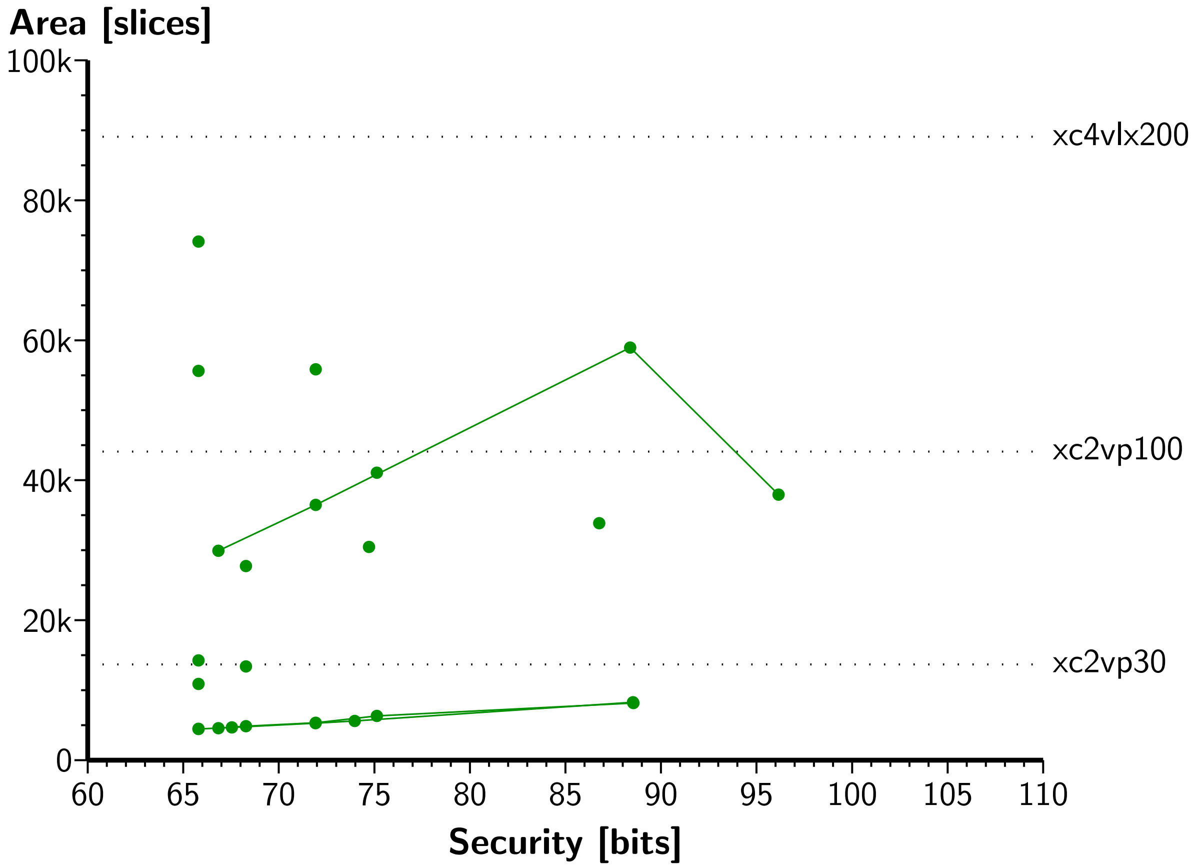
Calculation time [μs]
1000
100
10
675.5 μs / $\mathbb{F}_{2^{557}}$
100.8 μs / $\mathbb{F}_{2^{457}}$
20.9 μs / $\mathbb{F}_{3^{97}}$
Virtex-II Pro
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

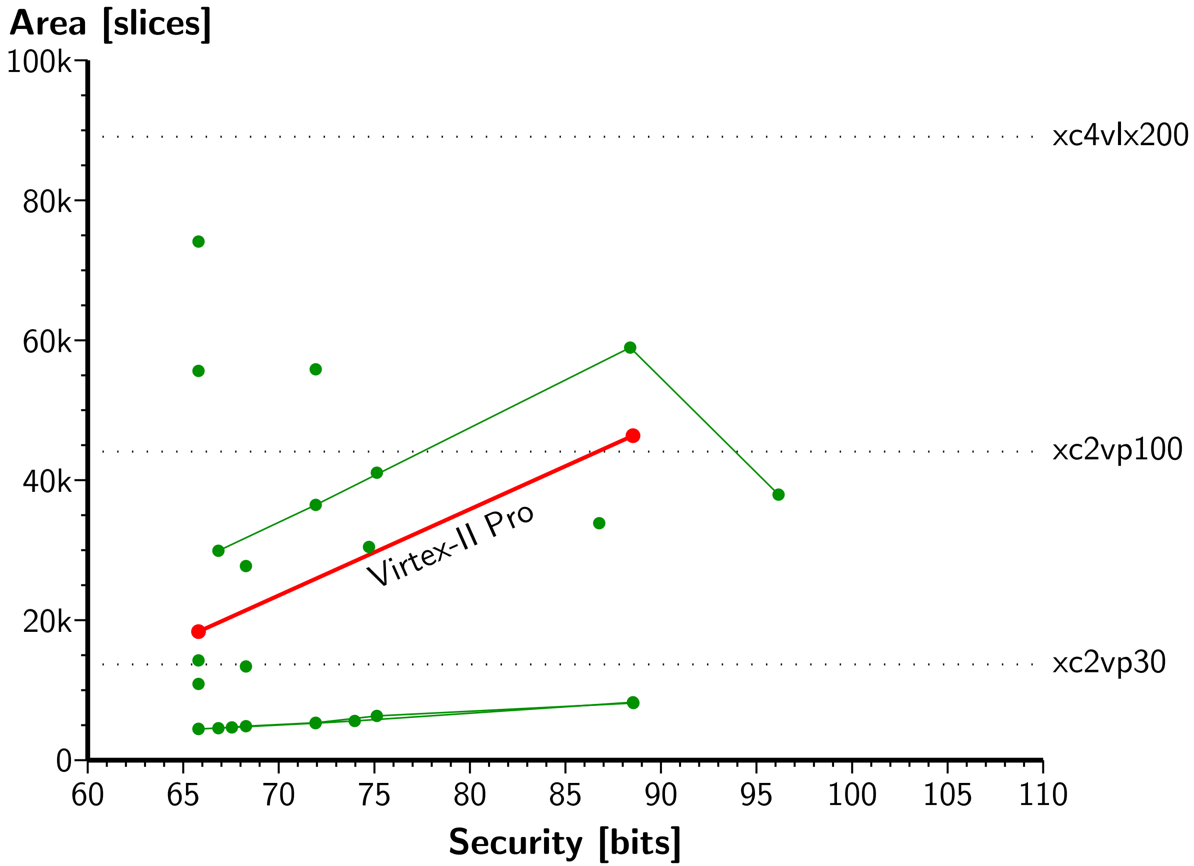

Calculation time [μs]
1000
100
10
675.5 μs / $\mathbb{F}_{2^{557}}$
100.8 μs / $\mathbb{F}_{2^{457}}$
20.9 μs / $\mathbb{F}_{3^{97}}$
12.8 μs / $\mathbb{F}_{3^{193}}$
6.2 μs / $\mathbb{F}_{3^{97}}$
Virtex-II Pro
60 65 70 75 80 85 90 95 100 105 110
Security [bits]

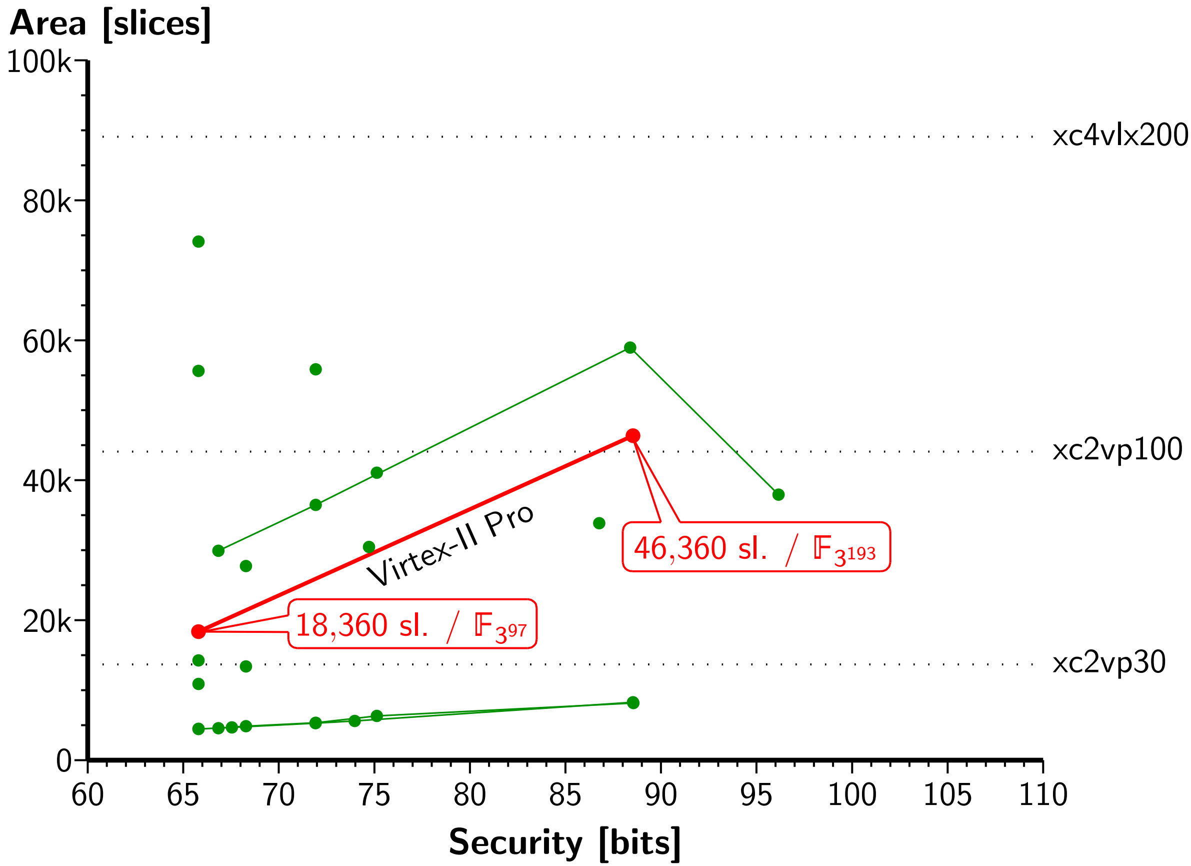


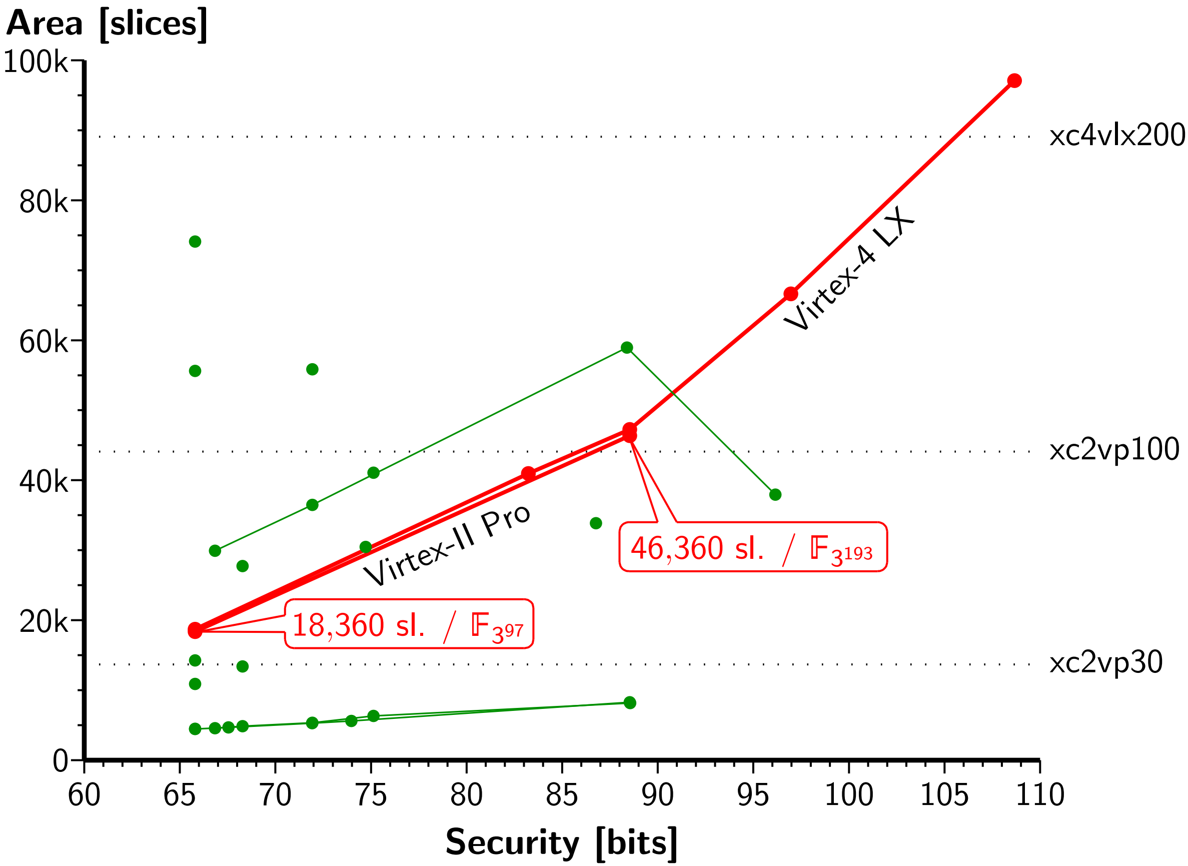
Calculation time [μs]
1000
100
10
675.5 μs / $\mathbb{F}_{2^{557}}$
100.8 μs / $\mathbb{F}_{2^{457}}$
20.9 μs / $\mathbb{F}_{3^{97}}$
12.8 μs / $\mathbb{F}_{3^{193}}$
6.2 μs / $\mathbb{F}_{3^{97}}$
Virtex-II Pro
Virtex-4 LX
60
65
70
75
80
85
90
95
100
105
110
Security [bits]


Calculation time [μs]
1000
100
10
675.5 μs / $\mathbb{F}_{2^{557}}$
100.8 μs / $\mathbb{F}_{2^{457}}$
20.9 μs / $\mathbb{F}_{3^{97}}$
16.9 μs / $\mathbb{F}_{3^{313}}$
12.8 μs / $\mathbb{F}_{3^{193}}$
6.2 μs / $\mathbb{F}_{3^{97}}$
Virtex-II Pro
Virtex-4 LX
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

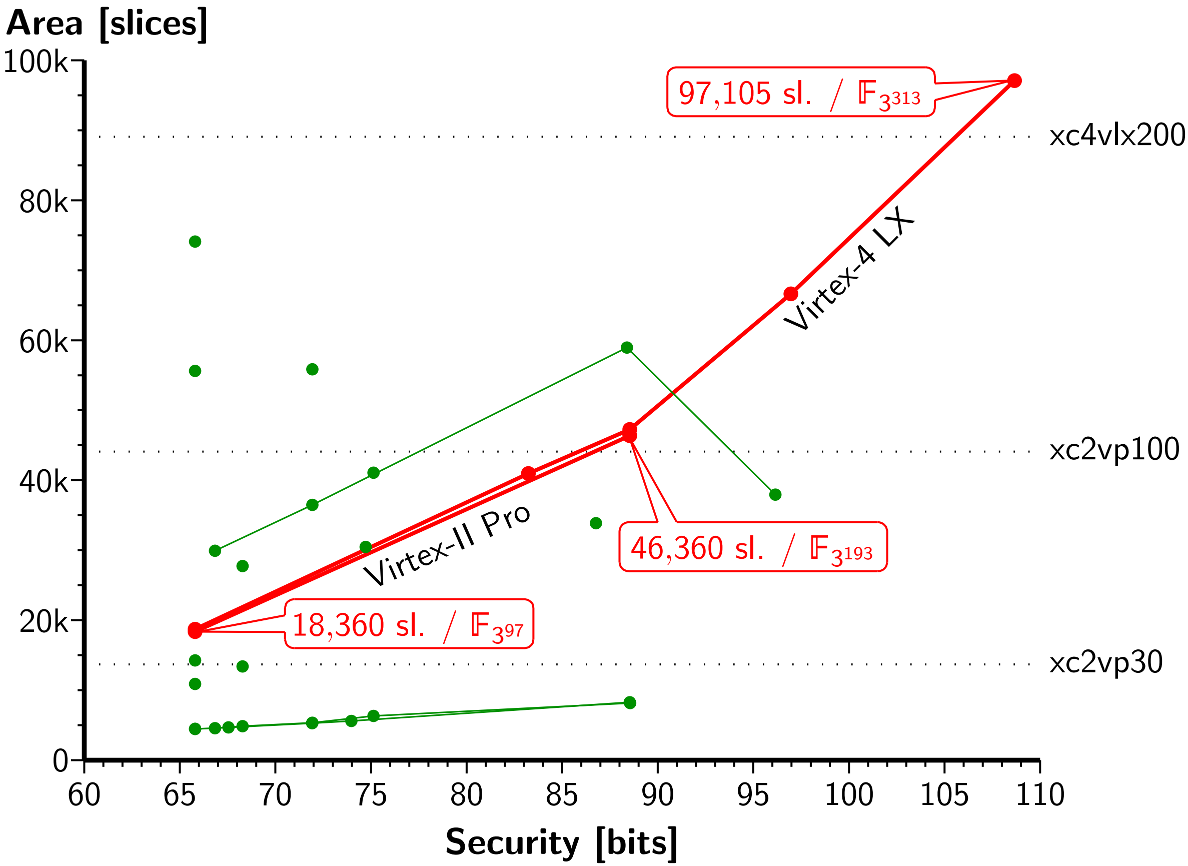

Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc2vp100
xc2vp30
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

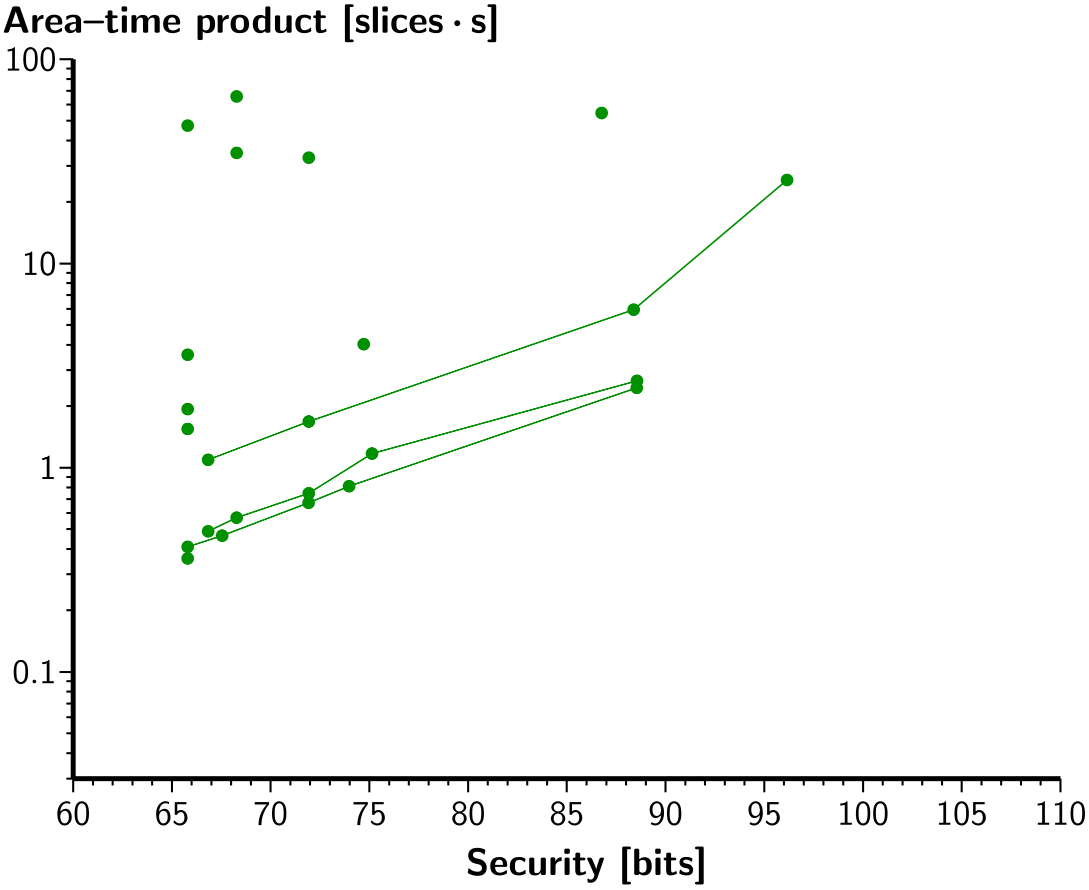

Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc2vp100
xc2vp30
Virtex-II Pro
60
65
70
75
80
85
90
95
100
105
110
Security [bits]


Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc2vp100
xc2vp30
Virtex-II Pro
46,360 sl. / $\mathbb{F}_{3^{193}}$
18,360 sl. / $\mathbb{F}_{3^{97}}$
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

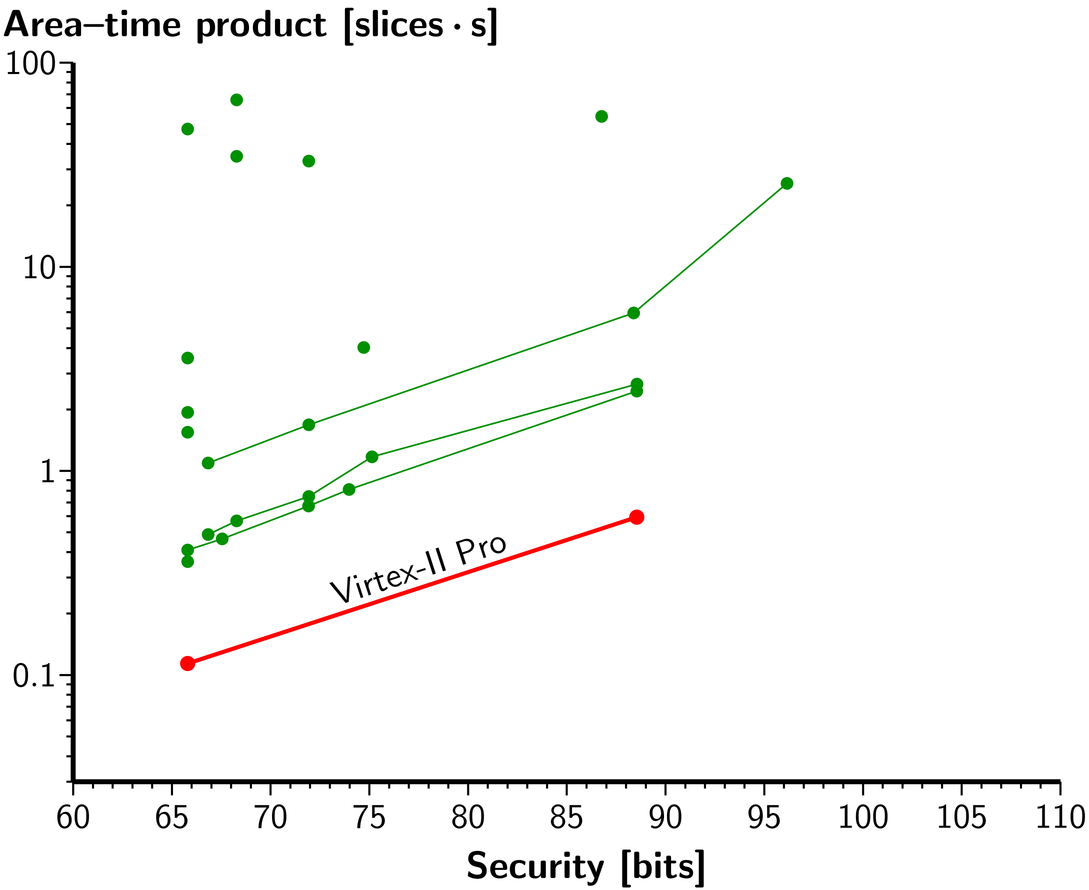


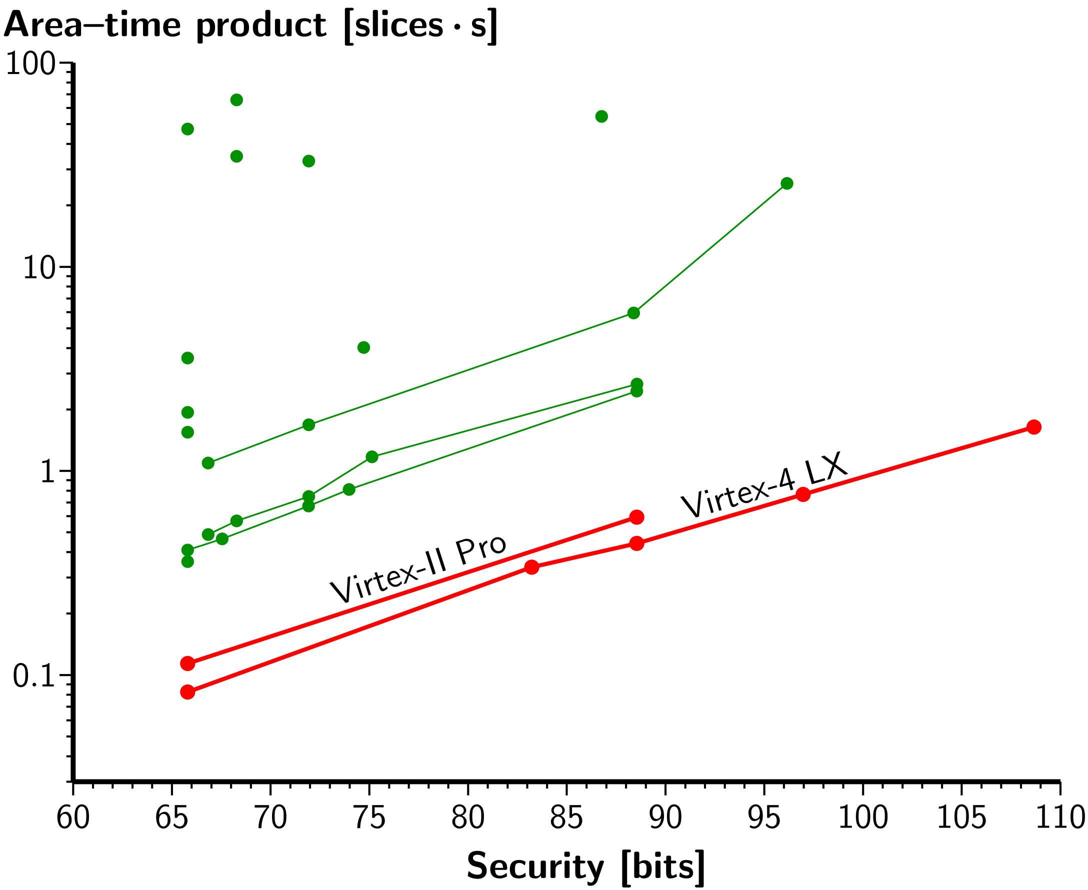
Area [slices]
100k
80k
60k
40k
20k
0
xc4vlx200
xc2vp100
xc2vp30
Virtex-4 LX
Virtex-II Pro
46,360 sl. / $\mathbb{F}_{3^{193}}$
18,360 sl. / $\mathbb{F}_{3^{97}}$
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

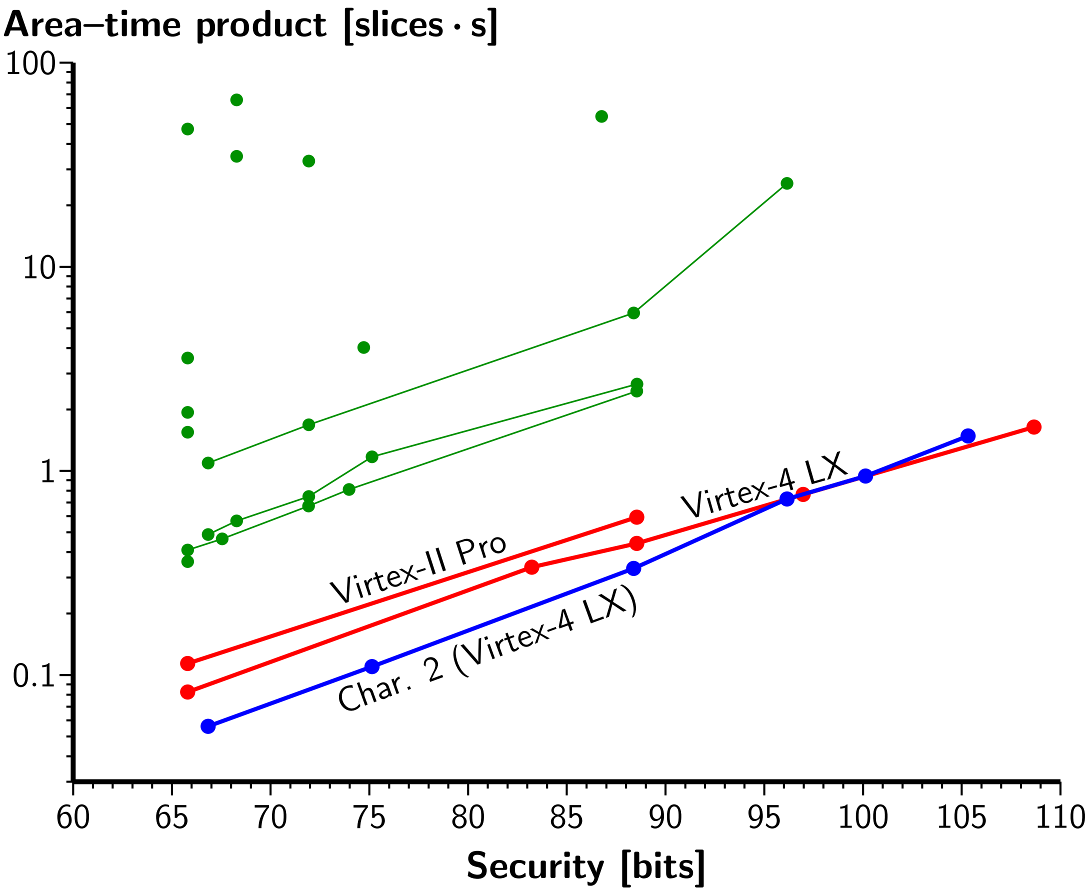

Area [slices]
100k
80k
60k
40k
20k
0
97,105 sl. / $\mathbb{F}_{3^{313}}$
xc4vlx200
Virtex-4 LX
xc2vp100
Virtex-II Pro
46,360 sl. / $\mathbb{F}_{3^{193}}$
18,360 sl. / $\mathbb{F}_{3^{97}}$
xc2vp30
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

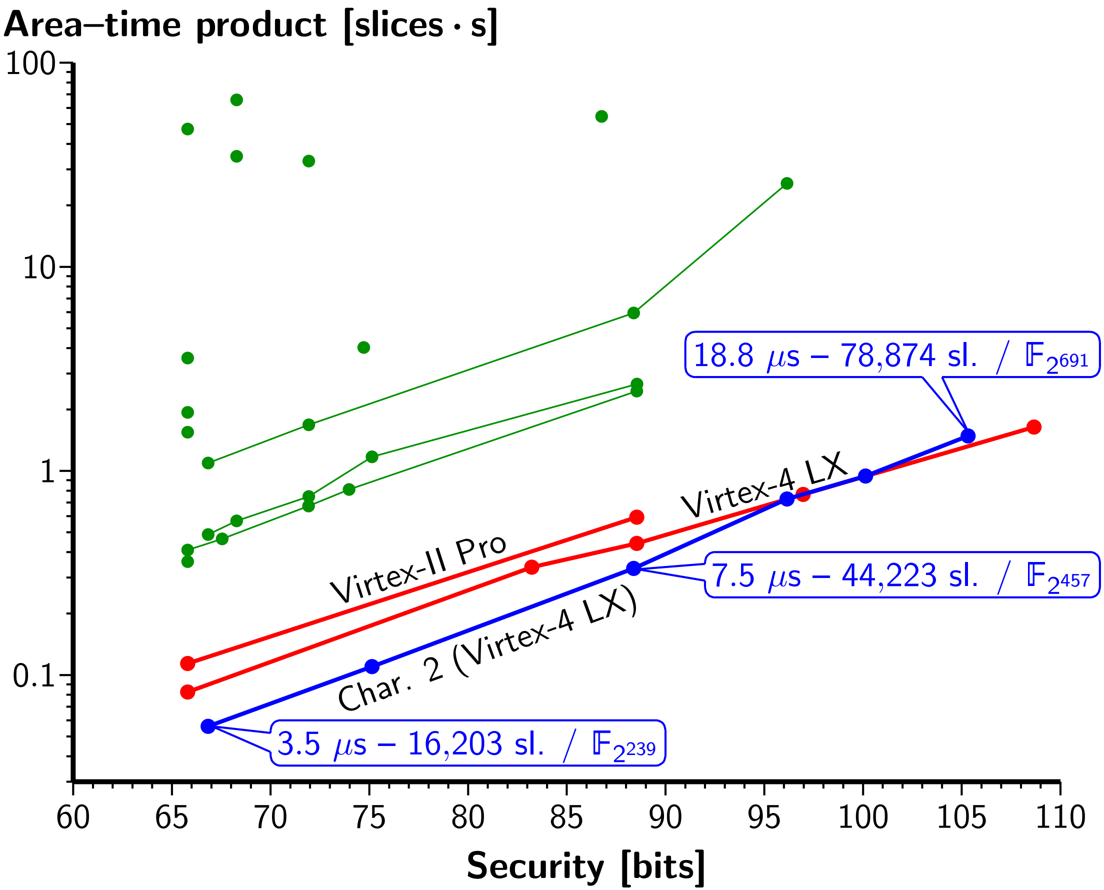


Area–time product [slices · s]
100
10
1
0.1
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

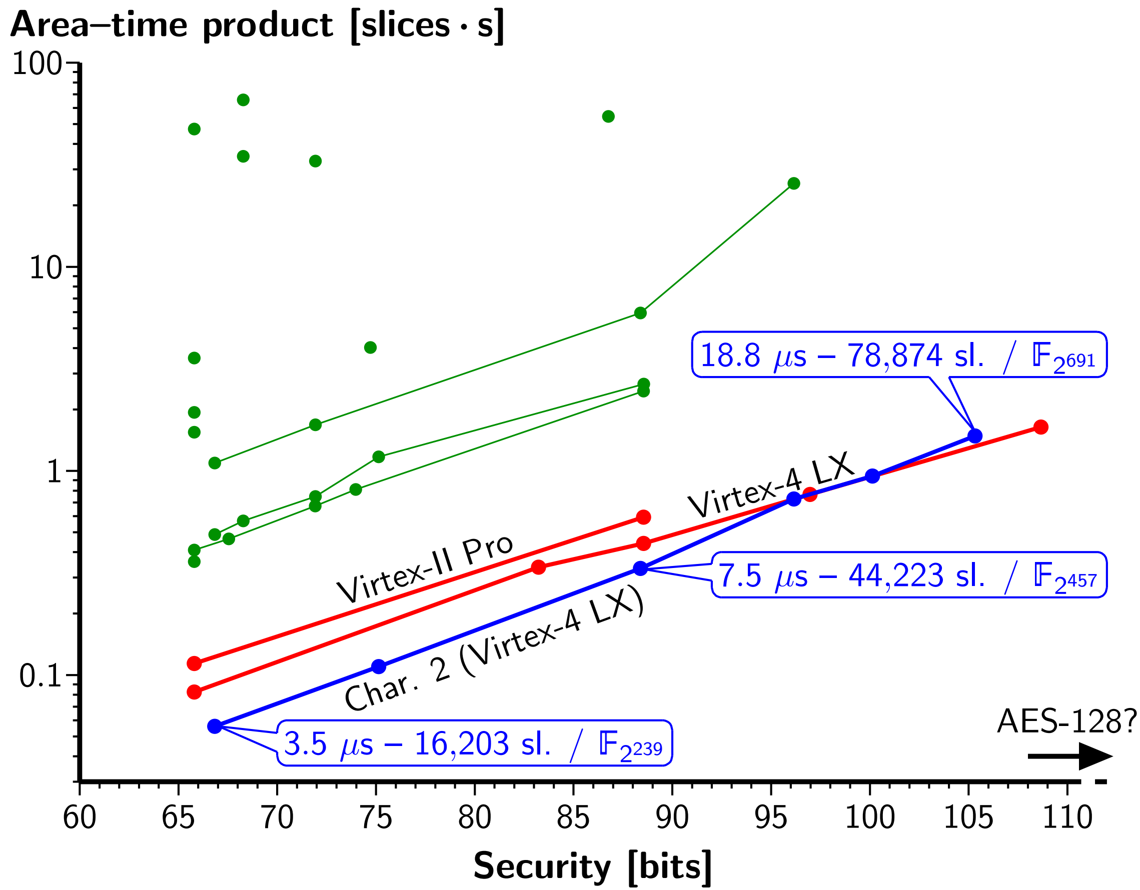

Area–time product [slices · s]
100
10
1
0.1
Virtex-II Pro
60
65
70
75
80
85
90
95
100
105
110
Security [bits]


Area–time product [slices · s]
100
10
1
0.1
Virtex-II Pro
Virtex-4 LX
60
65
70
75
80
85
90
95
100
105
110
Security [bits]


Area–time product [slices · s]
100
10
1
0.1
60
65
70
75
80
85
90
95
100
105
110
Security [bits]
Virtex-II Pro
Virtex-4 LX
Char. 2 (Virtex-4 LX)



Area–time product [slices · s]
100
10
1
0.1
60
65
70
75
80
85
90
95
100
105
110
Security [bits]
18.8 μs – 78,874 sl. / $\mathbb{F}_{2^{691}}$
Virtex-4 LX
Virtex-II Pro
7.5 μs – 44,223 sl. / $\mathbb{F}_{2^{457}}$
Char. 2 (Virtex-4 LX)
3.5 μs – 16,203 sl. / $\mathbb{F}_{2^{239}}$



Area–time product [slices · s]
100
10
1
0.1
18.8 μs – 78,874 sl. / $\mathbb{F}_{2^{691}}$
Virtex-4 LX
Virtex-II Pro
7.5 μs – 44,223 sl. / $\mathbb{F}_{2^{457}}$
Char. 2 (Virtex-4 LX)
AES-128?
3.5 μs – 16,203 sl. / $\mathbb{F}_{2^{239}}$
60
65
70
75
80
85
90
95
100
105
110
Security [bits]

Outline of the talk

- Pairing-based cryptography
- Hardware accelerator for the Tate pairing
- Implementation results
- Concluding thoughts

Conclusion

- A new architecture for pairing computation
  - two specialized coprocessors
  - bet on parallelizing multiplier
  - based on Karatsuba multiplication scheme
  - importance of architecture–algorithm co-design
  - careful bubble-free scheduling of Miller's loop

Conclusion

- A new architecture for pairing computation
  - two specialized coprocessors
  - bet on parallelizing multiplier
  - based on Karatsuba multiplication scheme
  - importance of architecture–algorithm co-design
  - careful bubble-free scheduling of Miller's loop
- High-performance accelerator
  - the fastest coprocessor (17 μs for 109 bits of security)
  - the best area–time trade-off
  - scales to higher security levels

Future work

- Fully parallel multipliers
  - try other algorithms: Toom–Cook, Montgomery's formulae

Future work

- Fully parallel multipliers
  - try other algorithms: Toom–Cook, Montgomery's formulae
- Final-exponentiation coprocessor
  - full-featured finite-field processor
  - compute the full pairing with it (work in progress)

Future work

- Fully parallel multipliers
  - try other algorithms: Toom–Cook, Montgomery's formulae
- Final-exponentiation coprocessor
  - full-featured finite-field processor
  - compute the full pairing with it (work in progress)
- Toward AES-128 security level
  - characteristic 2 (recently submitted)
  - genus-2 supersingular curves in characteristic 2 (work in progress)
  - Barreto–Naehrig curves

Thank you for your attention

Questions?