Cocks–Pinch curves with efficient ate pairing

Simon Masson
Joint work with A. Guillevic, E. Thomé

Thales – LORIA

December 11, 2018

Pairings on elliptic curves

Definition

A pairing on an elliptic curve E is a bilinear non-degenerate application $e : E \times E \longrightarrow \mathbb{F}_{p^k}^{\times}$

Pairings on elliptic curves

Definition

A pairing on an elliptic curve E is a bilinear non-degenerate application $e : E \times E \longrightarrow \mathbb{F}_{p^k}^{\times}$

For some particular $P, Q \in E$ and $a, b \in \mathbb{Z}$,

Pairings on elliptic curves

Definition

A pairing on an elliptic curve E is a bilinear non-degenerate application $e : E \times E \longrightarrow \mathbb{F}_{p^k}^{\times}$

For some particular $P, Q \in E$ and $a, b \in \mathbb{Z}$,

$$e(aP, bQ)$$

Pairings on elliptic curves

Definition

A pairing on an elliptic curve E is a bilinear non-degenerate application $e : E \times E \longrightarrow \mathbb{F}_{p^k}^{\times}$

For some particular $P, Q \in E$ and $a, b \in \mathbb{Z}$,

$$e(aP, bQ) = e(P, bQ)^a$$

Pairings on elliptic curves

Definition

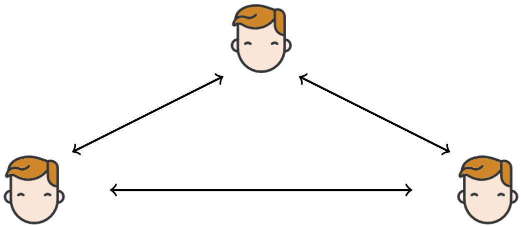
A pairing on an elliptic curve E is a bilinear non-degenerate application $e : E \times E \longrightarrow \mathbb{F}_{p^k}^{\times}$

For some particular $P, Q \in E$ and $a, b \in \mathbb{Z}$,

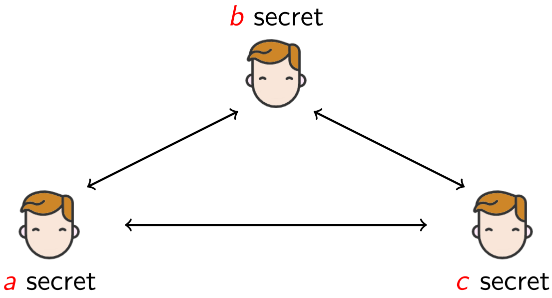
$$e(aP, bQ) = e(P, bQ)^a = e(P, Q)^{ab}$$

Application 1. Tripartite one round key exchange. (Joux 2000)

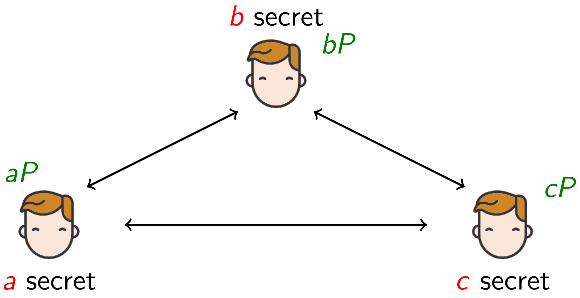
Application 1. Tripartite one round key exchange. (Joux 2000)



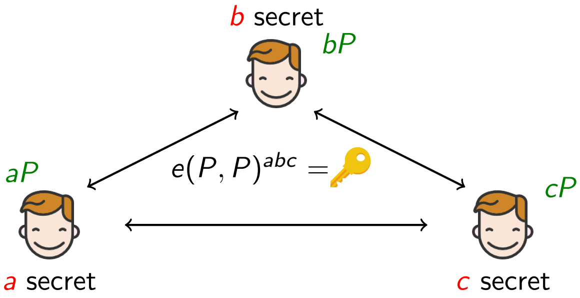
Application 1. Tripartite one round key exchange. (Joux 2000)



Application 1. Tripartite one round key exchange. (Joux 2000)



Application 1. Tripartite one round key exchange. (Joux 2000)



Application 2. BLS signature
$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.

Application 2. BLS signature
$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.
Secret key: $s_k \in \{2, \ldots, n-1\}$.
Public key: $P_k = [s_k]P$.

Application 2. BLS signature

$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.

Secret key: $s_k \in \{2, \ldots, n-1\}$.

Public key: $P_k = [s_k]P$.

Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.

Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

Application 2. BLS signature

$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.

Secret key: $s_k \in \{2, \ldots, n-1\}$.

Public key: $P_k = [s_k]P$.

Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.

Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

$$e(P_k, H(M)) = e([s_k]P, H(M)) = e(P, [s_k]H(M)) = e(P, \sigma)$$

Application 2. BLS signature

$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.

Secret key: $s_k \in \{2, \ldots, n-1\}$.

Public key: $P_k = [s_k]P$.

Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.

Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

$$e(P_k, H(M)) = e([s_k]P, H(M)) = e(P, [s_k]H(M)) = e(P, \sigma)$$

Application 3. Blind signature An authority has secret key s_k and public key $P_k = [s_k]P$ as before.

Application 2. BLS signature
$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.
Secret key: $s_k \in \{2, \ldots, n-1\}$.
Public key: $P_k = [s_k]P$.
Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.
Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

$$e(P_k, H(M)) = e([s_k]P, H(M)) = e(P, [s_k]H(M)) = e(P, \sigma)$$

Application 3. Blind signature An authority has secret key s_k and public key $P_k = [s_k]P$ as before.
Compute $H(M)$ and send $Q = H(M) + [r]P$ for $r \in_R \{2, \ldots, n-1\}$ to the authority.

Application 2. BLS signature
$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.
Secret key: $s_k \in \{2, \ldots, n-1\}$.
Public key: $P_k = [s_k]P$.
Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.
Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

$$e(P_k, H(M)) = e([s_k]P, H(M)) = e(P, [s_k]H(M)) = e(P, \sigma)$$

Application 3. Blind signature An authority has secret key s_k and public key $P_k = [s_k]P$ as before.
Compute $H(M)$ and send $Q = H(M) + [r]P$ for $r \in_R \{2, \ldots, n-1\}$ to the authority.
Autorithy answer: $A = [s_k]Q$.

Application 2. BLS signature
$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.
Secret key: $s_k \in \{2, \ldots, n-1\}$.
Public key: $P_k = [s_k]P$.
Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.
Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

$$e(P_k, H(M)) = e([s_k]P, H(M)) = e(P, [s_k]H(M)) = e(P, \sigma)$$

Application 3. Blind signature An authority has secret key s_k and public key $P_k = [s_k]P$ as before.
Compute $H(M)$ and send $Q = H(M) + [r]P$ for $r \in_R \{2, \ldots, n-1\}$ to the authority.
Autority answer: $A = [s_k]Q$.
Blind (BLS) signature: $\sigma = A - [r]P_k = [s_k]H(M)$.

Application 2. BLS signature
$H : \{0,1\}^* \to \langle P \rangle$ is a hash function, with $P \in E(\mathbb{F}_p)$ of prime order n.
Secret key: $s_k \in \{2, \ldots, n-1\}$.
Public key: $P_k = [s_k]P$.
Signing a message $M \in \{0,1\}^*$: $\sigma = [s_k]H(M)$.
Verifying the signature: $e(P_k, H(M)) \stackrel{?}{=} e(P, \sigma)$.

$$e(P_k, H(M)) = e([s_k]P, H(M)) = e(P, [s_k]H(M)) = e(P, \sigma)$$

Application 3. Blind signature An authority has secret key s_k and public key $P_k = [s_k]P$ as before.
Compute $H(M)$ and send $Q = H(M) + [r]P$ for $r \in_R \{2, \ldots, n-1\}$ to the authority.
Autority answer: $A = [s_k]Q$.
Blind (BLS) signature: $\sigma = A - [r]P_k = [s_k]H(M)$.
Verification: same as for BLS.

Application 4. Identity based encryption
$H_1 : \{0,1\}^* \to E$ and $H_2 : \mathbb{F}_{p^k} \to \{0,1\}^n$ are hash functions.
The PKG has a secret key s and a public key $P_k = [s]P$.
$Q_{\text{id}} = H_1(id)$ and $S_{\text{id}} = [s]Q_{\text{id}}$ is obtained from the PKG.

Application 4. Identity based encryption

$H_1 : \{0,1\}^* \to E$ and $H_2 : \mathbb{F}_{p^k} \to \{0,1\}^n$ are hash functions.

The PKG has a secret key s and a public key $P_k = [s]P$.

$Q_{\text{id}} = H_1(id)$ and $S_{\text{id}} = [s]Q_{\text{id}}$ is obtained from the PKG.

- Encryption

Set $r \in_R \{2, \ldots, n-1\}$

Compute $g_{\text{id}} = e(Q_{\text{id}}, P_k)$

Send $(u, v) = ([r]P, m \oplus H_2(g_{\text{id}}^r))$.

Application 4. Identity based encryption

$H_1 : \{0,1\}^* \to E$ and $H_2 : \mathbb{F}_{p^k} \to \{0,1\}^n$ are hash functions.

The PKG has a secret key s and a public key $P_k = [s]P$.

$Q_{\text{id}} = H_1(id)$ and $S_{\text{id}} = [s]Q_{\text{id}}$ is obtained from the PKG.

- Encryption

Set $r \in_R \{2, \dots, n-1\}$

Compute $g_{\text{id}} = e(Q_{\text{id}}, P_k)$

Send $(u, v) = ([r]P, m \oplus H_2(g_{\text{id}}^r))$.

- Decryption

Recover $m = v \oplus H_2(e(S_{\text{id}}, u))$.

Application 4. Identity based encryption

$H_1 : \{0,1\}^* \to E$ and $H_2 : \mathbb{F}_{p^k} \to \{0,1\}^n$ are hash functions.

The PKG has a secret key s and a public key $P_k = [s]P$.

$Q_{\mathrm{id}} = H_1(id)$ and $S_{\mathrm{id}} = [s]Q_{\mathrm{id}}$ is obtained from the PKG.

- Encryption

Set $r \in_R \{2, \ldots, n-1\}$

Compute $g_{\mathrm{id}} = e(Q_{\mathrm{id}}, P_k)$

Send $(u, v) = ([r]P, m \oplus H_2(g_{\mathrm{id}}^r))$.

- Decryption

Recover $m = v \oplus H_2(e(S_{\mathrm{id}}, u))$.

$$e(S_{\mathrm{id}}, u) = e([s]Q_{\mathrm{id}}, [r]P) = e(Q_{\mathrm{id}}, P)^{rs} = e(Q_{\mathrm{id}}, P_k)^r$$

Tate and ate pairing

1. Tate and ate pairing
2. Pairing-friendly curves for 128 bits of security
3. Timings and comparisons

The Tate and ate pairings are computed in two steps:

1. Evaluating a function at a point of the curve (Miller loop)
2. Exponentiating to the power $(p^k - 1)/r$ (final exponentiation).

The Tate and ate pairings are computed in two steps:

1. Evaluating a function at a point of the curve (Miller loop)
2. Exponentiating to the power $(p^k - 1)/r$ (final exponentiation).

Definition

For $P, Q \in E[r]$ such that $\pi_p(P) = P$, $\pi_p(Q) = [p]Q$,

$$\mathrm{Tate}(P, Q) := f_{r,P}(Q)^{(p^k-1)/r} \qquad \mathrm{ate}(P, Q) := f_{t-1,Q}(P)^{(p^k-1)/r}$$

Miller loop step.

Miller loop step.

Definition

The Miller loop computes the function $f_{s,Q}$ such that Q is a zero of order s, and $[s]Q$ is a pole of order 1, i.e

$$\operatorname{div}(f_{s,Q}) = s(Q) - ([s]Q) - (s-1)\mathcal{O}$$

Miller loop step.

Definition

The Miller loop computes the function $f_{s,Q}$ such that Q is a zero of order s, and $[s]Q$ is a pole of order 1, i.e

$$\mathrm{div}(f_{s,Q}) = s(Q) - ([s]Q) - (s-1)\mathcal{O}$$

Miller loop for Tate.
Compute $x = f_{r,P}(Q)$ with $P \in E(\mathbb{F}_p)[r]$ and $Q \in E(\mathbb{F}_{p^k})[r]$.

Miller loop step.

Definition

The Miller loop computes the function $f_{s,Q}$ such that Q is a zero of order s, and $[s]Q$ is a pole of order 1, i.e

$$\mathrm{div}(f_{s,Q}) = s(Q) - ([s]Q) - (s-1)\mathcal{O}$$

Miller loop for Tate.
Compute $x = f_{r,P}(Q)$ with $P \in E(\mathbb{F}_p)[r]$ and $Q \in E(\mathbb{F}_{p^k})[r]$.

Miller loop for ate.
For ate: compute $x = f_{t-1,Q}(P)$ with $P \in E(\mathbb{F}_p)[r]$ and $Q \in E(\mathbb{F}_{p^k})[r]$.

Algorithm: $\textsc{MillerLoop}(s, P, Q)$ – Compute $f_{s,Q}(P)$.

$f \leftarrow 1$
$S \leftarrow Q$
for b bit of s from second MSB to LSB **do**
  $f \leftarrow f^2 \cdot \ell_{S,S}(P)/v_{2S}(P)$
  $S \leftarrow [2]S$
  if $b = 1$ **then**
    $f \leftarrow f \cdot \ell_{S,Q}(P)/v_{S+Q}(P)$
    $S \leftarrow S + Q$
  end if
end for
return f such that $\mathrm{div}(f_{s,Q}) = s(Q) - ([s]Q) - (s-1)\mathcal{O}$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{\boxed{1}01}^{2}$$

$$f = 1$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{1\boxed{0}1}^{2}$$

$$f = 1$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{1\boxed{0}1}^{2}$$

$$f = 1^2$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{1\boxed{0}1}^{2}$$

$$f = 1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{10\boxed{1}}^{2}$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{10\boxed{1}}^{2}$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P)$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{10\boxed{1}}^{2}$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q)$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q) + 2(2Q) + (-4Q)$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q) + 2(2Q) + (-4Q) + (Q) + (4Q) + (-5Q)$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q) + 2(2Q) + (-4Q) + (Q) + (4Q) + (-5Q)$$

$$-2(2Q) - 2(-2Q) - 2(\mathcal{O})$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q) + 2(2Q) + (-4Q) + (Q) + (4Q) + (-5Q)$$

$$-2(2Q) - 2(-2Q) - 2(\mathcal{O}) - (4Q) - (-4Q) - (\mathcal{O})$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q) + 2(2Q) + (-4Q) + (Q) + (4Q) + (-5Q)$$

$$-2(2Q) - 2(-2Q) - 2(\mathcal{O}) - (4Q) - (-4Q) - (\mathcal{O}) - (5Q) - (-5Q) - (\mathcal{O})$$

Example: $f_{5,Q}(P)$.

$$s = 5 = \overline{101}^2$$

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

Divisor:

$$4(Q) + 2(-2Q) + 2(2Q) + (-4Q) + (Q) + (4Q) + (-5Q)$$

$$-2(2Q) - 2(-2Q) - 2(\mathcal{O}) - (4Q) - (-4Q) - (\mathcal{O}) - (5Q) - (-5Q) - (\mathcal{O})$$

$$\mathrm{div}(f) = 5(Q) - (5Q) - 4(\mathcal{O})$$

Final exponentiation step.

Final exponentiation step.

$f_{r,P}(Q)$ and $f_{t-1,Q}(P)$ are r-th roots of unity.
We obtain a unique coset elevating to the power $(p^k - 1)/r$.

Final exponentiation step.
$f_{r,P}(Q)$ and $f_{t-1,Q}(P)$ are r-th roots of unity.
We obtain a unique coset elevating to the power $(p^k - 1)/r$.

$$(f_{r,P}(Q)u^r)^{(p^k-1)/r} = f_{r,P}(Q)^{(p^k-1)/r}u^{p^k-1} = f_{r,P}(Q)^{(p^k-1)/r}$$

Final exponentiation step.

$f_{r,P}(Q)$ and $f_{t-1,Q}(P)$ are r-th roots of unity.
We obtain a unique coset elevating to the power $(p^k-1)/r$.

$$(f_{r,P}(Q)u^r)^{(p^k-1)/r} = f_{r,P}(Q)^{(p^k-1)/r}u^{p^k-1} = f_{r,P}(Q)^{(p^k-1)/r}$$

$(p^k-1)/r$ is very large so the exponentiation is expensive. 

Final exponentiation step.

$f_{r,P}(Q)$ and $f_{t-1,Q}(P)$ are r-th roots of unity.
We obtain a unique coset elevating to the power $(p^k-1)/r$.

$$(f_{r,P}(Q)u^r)^{(p^k-1)/r} = f_{r,P}(Q)^{(p^k-1)/r}u^{p^k-1} = f_{r,P}(Q)^{(p^k-1)/r}$$

$(p^k-1)/r$ is very large so the exponentiation is expensive.

Proposition

For x in a subfield of $\mathbb{F}_{p^k}^\times$, $x^{\frac{p^k-1}{r}} = 1$.

Final exponentiation step.

$f_{r,P}(Q)$ and $f_{t-1,Q}(P)$ are r-th roots of unity.
We obtain a unique coset elevating to the power $(p^k-1)/r$.

$$(f_{r,P}(Q)u^r)^{(p^k-1)/r} = f_{r,P}(Q)^{(p^k-1)/r}u^{p^k-1} = f_{r,P}(Q)^{(p^k-1)/r}$$

$(p^k-1)/r$ is very large so the exponentiation is expensive.

Proposition

For x in a subfield of $\mathbb{F}_{p^k}^\times$, $x^{\frac{p^k-1}{r}} = 1$.

Factors in subfields do not need to be computed !

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :
  $$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x,y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$
  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.
- When $4 \mid k$ and $b = 0$ or $6 \mid k$ and $a = 0$, the curve has a quartic or a sextic twist.
  $Q \in E(\mathbb{F}_{p^k}) \simeq E'(\mathbb{F}_{p^{k/4}})$ or $E'(\mathbb{F}_{p^{k/6}})$. Line computations are more efficient.

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.
- When $4 \mid k$ and $b = 0$ or $6 \mid k$ and $a = 0$, the curve has a quartic or a sextic twist.
  $Q \in E(\mathbb{F}_{p^k}) \simeq E'(\mathbb{F}_{p^{k/4}})$ or $E'(\mathbb{F}_{p^{k/6}})$. Line computations are more efficient.

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.
- When $4 \mid k$ and $b = 0$ or $6 \mid k$ and $a = 0$, the curve has a quartic or a sextic twist.
  $Q \in E(\mathbb{F}_{p^k}) \simeq E'(\mathbb{F}_{p^{k/4}})$ or $E'(\mathbb{F}_{p^{k/6}})$. Line computations are more efficient.

$$f = (1^2 \cdot \ell_{Q,Q}(P)/v_{2Q}(P))^2 \cdot \ell_{2Q,2Q}(P)/v_{4Q}(P) \cdot \ell_{4Q,Q}(P)/v_{5Q}(P)$$

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.
- When $4 \mid k$ and $b = 0$ or $6 \mid k$ and $a = 0$, the curve has a quartic or a sextic twist.
  $Q \in E(\mathbb{F}_{p^k}) \simeq E'(\mathbb{F}_{p^{k/4}})$ or $E'(\mathbb{F}_{p^{k/6}})$. Line computations are more efficient.

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P) \cdot \ell_{4Q,Q}(P)$$

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.
- When $4 \mid k$ and $b = 0$ or $6 \mid k$ and $a = 0$, the curve has a quartic or a sextic twist.
  $Q \in E(\mathbb{F}_{p^k}) \simeq E'(\mathbb{F}_{p^{k/4}})$ or $E'(\mathbb{F}_{p^{k/6}})$. Line computations are more efficient.

$$f = \left(1^2 \cdot \ell_{Q,Q}(P)\right)^2 \cdot \ell_{2Q,2Q}(P) \cdot \ell_{4Q,Q}(P)$$

- When k is even, say $\mathbb{F}_{p^k} = \mathbb{F}_{p^{k/2}}(\sqrt{\alpha})$.
  Quadratic twist :

$$\begin{array}{ccc} E' & \xrightarrow{\sim} & E \\ (x, y) & \longmapsto & (\alpha x, \sqrt{\alpha}^3 y) \end{array}$$

  The isomorphism is defined over $\mathbb{F}_{p^k}$ and E has full r-torsion defined over $\mathbb{F}_{p^k}$.
  $Q \in E(\mathbb{F}_{p^k})[r]$ is seen as twist$(\tilde{Q})$ with $\tilde{Q}$ with two coordinates in $\mathbb{F}_{p^{k/2}}$.
  Vertical lines $v_S(P) = x_S - x_P \in \mathbb{F}_{p^{k/2}}$ because $x_S \in \mathbb{F}_{p^{k/2}}$ and $P \in E(\mathbb{F}_p)$.
- When $4 \mid k$ and $b = 0$ or $6 \mid k$ and $a = 0$, the curve has a quartic or a sextic twist.
  $Q \in E(\mathbb{F}_{p^k}) \simeq E'(\mathbb{F}_{p^{k/4}})$ or $E'(\mathbb{F}_{p^{k/6}})$. Line computations are more efficient.

$$f = \ell_{Q,Q}(P)^2 \ell_{2Q,2Q}(P) \ell_{4Q,Q}(P)$$

$$\frac{p^k - 1}{r} = \frac{p^k - 1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$$\frac{p^k - 1}{r} = \frac{p^k - 1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$\frac{p^k-1}{\Phi_k(p)}$ is a polynomial in p with very small coefficients.
Easy exponentiation with Frobenius when $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k - \alpha)$:

$$\frac{p^k-1}{r} = \frac{p^k-1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$\frac{p^k-1}{\Phi_k(p)}$ is a polynomial in p with very small coefficients.

Easy exponentiation with Frobenius when $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k - \alpha)$:

$a^p = \left(\sum_{i=0}^{k-1} a_i x^i\right)^p = \sum_{i=0}^{k-1} a_i x^{ip}$ and x^{ip} can be precomputed.

$$\frac{p^k - 1}{r} = \frac{p^k - 1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$\frac{p^k-1}{\Phi_k(p)}$ is a polynomial in p with very small coefficients.

Easy exponentiation with Frobenius when $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k - \alpha)$:

$a^p = \left(\sum_{i=0}^{k-1} a_i x^i\right)^p = \sum_{i=0}^{k-1} a_i x^{ip}$ and x^{ip} can be precomputed.

A Frobenius costs $k - 1$ multiplications over $\mathbb{F}_p$.

$$\frac{p^k - 1}{r} = \frac{p^k - 1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$\frac{p^k-1}{\Phi_k(p)}$ is a polynomial in p with very small coefficients.
Easy exponentiation with Frobenius when $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k - \alpha)$:
$a^p = \left(\sum_{i=0}^{k-1} a_i x^i\right)^p = \sum_{i=0}^{k-1} a_i x^{ip}$ and x^{ip} can be precomputed.
A Frobenius costs $k-1$ multiplications over $\mathbb{F}_p$.

Last part $\frac{\Phi_k(p)}{r}$: more expensive, decompose into polynomials and compute efficiently with Horner rule:

$$\frac{p^k-1}{r} = \frac{p^k-1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$\frac{p^k-1}{\Phi_k(p)}$ is a polynomial in p with very small coefficients.
Easy exponentiation with Frobenius when $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k-\alpha)$:
$a^p = \left(\sum_{i=0}^{k-1} a_i x^i\right)^p = \sum_{i=0}^{k-1} a_i x^{ip}$ and x^{ip} can be precomputed.
A Frobenius costs $k-1$ multiplications over $\mathbb{F}_p$.

Last part $\frac{\Phi_k(p)}{r}$: more expensive, decompose into polynomials and compute efficiently with Horner rule:

$$a^{\sum_{i=0}^{3} x_i p^i} = ((((a^{x_3})^p)a^{x_2})^p a^{x_1})^p a^{x_0}$$

$$\frac{p^k-1}{r} = \frac{p^k-1}{\Phi_k(p)} \cdot \frac{\Phi_k(p)}{r}$$

$\frac{p^k-1}{\Phi_k(p)}$ is a polynomial in p with very small coefficients.

Easy exponentiation with Frobenius when $\mathbb{F}_{p^k} = \mathbb{F}_p[x]/(x^k - \alpha)$:

$a^p = \left(\sum_{i=0}^{k-1} a_i x^i\right)^p = \sum_{i=0}^{k-1} a_i x^{ip}$ and x^{ip} can be precomputed.

A Frobenius costs $k-1$ multiplications over $\mathbb{F}_p$.

Last part $\frac{\Phi_k(p)}{r}$: more expensive, decompose into polynomials and compute efficiently with Horner rule:

$$a^{\sum_{i=0}^{3} x_i p^i} = ((((a^{x_3})^p)a^{x_2})^p a^{x_1})^p a^{x_0}$$

Few exponentiations by x_i, multiplications and Frobenius.

Pairing-friendly curves for 128 bits of security

An elliptic curve E defined over $\mathbb{F}_p$, of trace t and discriminant D is pairing-friendly of embedding degree k if

- p, r are primes and t is relatively prime to p
- r divides $p+1-t$ and p^k-1 but does not divide p^i-1 for $1 \leq i < k$
- $4p - t^2 = Dy^2$ for a sufficiently small positive integer D and an integer y.

An elliptic curve E defined over $\mathbb{F}_p$, of trace t and discriminant D is pairing-friendly of embedding degree k if

- p, r are primes and t is relatively prime to p
- r divides $p+1-t$ and p^k-1 but does not divide p^i-1 for $1 \leq i < k$
- $4p - t^2 = Dy^2$ for a sufficiently small positive integer D and an integer y.

Cyclotomic families.

An elliptic curve E defined over $\mathbb{F}_p$, of trace t and discriminant D is pairing-friendly of embedding degree k if

- p, r are primes and t is relatively prime to p
- r divides $p+1-t$ and p^k-1 but does not divide p^i-1 for $1 \leq i < k$
- $4p - t^2 = Dy^2$ for a sufficiently small positive integer D and an integer y.

Cyclotomic families.

1. Find $r(x) \in \mathbb{Z}[x]$ such that $K := \mathbb{Q}[x]/(r(x))$ is a number field containing $\sqrt{-D}$ and $\mathbb{Q}(\zeta_k)$ for a chosen primitive k-th root ζ_k.

An elliptic curve E defined over $\mathbb{F}_p$, of trace t and discriminant D is pairing-friendly of embedding degree k if

- p, r are primes and t is relatively prime to p
- r divides $p+1-t$ and p^k-1 but does not divide p^i-1 for $1 \leq i < k$
- $4p-t^2 = Dy^2$ for a sufficiently small positive integer D and an integer y.

Cyclotomic families.

1. Find $r(x) \in \mathbb{Z}[x]$ such that $K := \mathbb{Q}[x]/(r(x))$ is a number field containing $\sqrt{-D}$ and $\mathbb{Q}(\zeta_k)$ for a chosen primitive k-th root ζ_k.
2. Let $t(x), y(x) \in \mathbb{Q}[x]$ mapping respectively to $\zeta_k + 1 \in K, (\zeta_k - 1)/\sqrt{-D} \in K$.

An elliptic curve E defined over $\mathbb{F}_p$, of trace t and discriminant D is pairing-friendly of embedding degree k if

- p, r are primes and t is relatively prime to p
- r divides $p+1-t$ and p^k-1 but does not divide p^i-1 for $1 \leq i < k$
- $4p-t^2 = Dy^2$ for a sufficiently small positive integer D and an integer y.

Cyclotomic families.

1. Find $r(x) \in \mathbb{Z}[x]$ such that $K := \mathbb{Q}[x]/(r(x))$ is a number field containing $\sqrt{-D}$ and $\mathbb{Q}(\zeta_k)$ for a chosen primitive k-th root ζ_k.
2. Let $t(x), y(x) \in \mathbb{Q}[x]$ mapping respectively to $\zeta_k + 1 \in K, (\zeta_k - 1)/\sqrt{-D} \in K$.
3. Let $p(x) \in \mathbb{Q}[x]$ be given by $(t(x)^2 + Dy(x)^2)/4$.

An elliptic curve E defined over $\mathbb{F}_p$, of trace t and discriminant D is pairing-friendly of embedding degree k if

- p, r are primes and t is relatively prime to p
- r divides $p+1-t$ and p^k-1 but does not divide p^i-1 for $1 \leq i < k$
- $4p - t^2 = Dy^2$ for a sufficiently small positive integer D and an integer y.

Cyclotomic families.

1. Find $r(x) \in \mathbb{Z}[x]$ such that $K := \mathbb{Q}[x]/(r(x))$ is a number field containing $\sqrt{-D}$ and $\mathbb{Q}(\zeta_k)$ for a chosen primitive k-th root ζ_k.
2. Let $t(x), y(x) \in \mathbb{Q}[x]$ mapping respectively to $\zeta_k + 1 \in K, (\zeta_k - 1)/\sqrt{-D} \in K$.
3. Let $p(x) \in \mathbb{Q}[x]$ be given by $(t(x)^2 + Dy(x)^2)/4$.

If $p(x)$ represents primes, choosing $x_0 \in \mathbb{Z}$ such that $y(x_0) \in \mathbb{Z}$ gives a pairing-friendly elliptic curve of embedding degree k, defined over $\mathbb{F}_{p(x_0)}$, of trace $t(x_0)$, with a subgroup of order $r(x_0)$ and discriminant D.

How to get the equation of the curve ? (Complex multiplication method).

How to get the equation of the curve ? (Complex multiplication method).

1. Compute the discriminant D of the curve : $p+1-t=-Dy^2$ with D square-free.

```
sage:  (p+1-t).square_free_part()
```

How to get the equation of the curve ? (Complex multiplication method).

1. Compute the discriminant D of the curve : $p+1-t=-Dy^2$ with D square-free.

```
sage: (p+1-t).square_free_part()
```

2. Compute the Hilbert class polynomial $H_D(X)$ whose roots are the j-invariants of curves with discriminant D.

```
sage: hilbert_class_polynomial(D)
```

How to get the equation of the curve ? (Complex multiplication method).

1. Compute the discriminant D of the curve : $p + 1 - t = -Dy^2$ with D square-free.

`sage: (p+1-t).square_free_part()`

2. Compute the Hilbert class polynomial $H_D(X)$ whose roots are the j-invariants of curves with discriminant D.

`sage: hilbert_class_polynomial(D)`

3. Compute a curve whose j-invariant is one of these roots.

`sage: EllipticCurve_from_j(j0).`

Example.

Example.
Barreto-Naehrig curves are elliptic curves of embedding degree $k = 12$, parametrized by

$$p(x) = 36x^4 + 36x^3 + 24x^2 + 6x + 1$$

$$r(x) = 36x^4 + 36x^3 + 18x^2 + 6x + 1$$

$$t(x) = 6x^2 + 1$$

For some integer x_0, $(p(x_0), r(x_0), t(x_0))$ parametrizes a pairing-friendly elliptic curve.

Example.

Barreto-Naehrig curves are elliptic curves of embedding degree $k = 12$, parametrized by

$$p(x) = 36x^4 + 36x^3 + 24x^2 + 6x + 1$$

$$r(x) = 36x^4 + 36x^3 + 18x^2 + 6x + 1$$

$$t(x) = 6x^2 + 1$$

For some integer x_0, $(p(x_0), r(x_0), t(x_0))$ parametrizes a pairing-friendly elliptic curve.

What about efficiency of the pairing computation ?

Miller loop.

k is even $\implies$ no vertical lines.

$6 \mid k$ and $D = 3 \implies$ twist of degree 6: $E(\mathbb{F}_{p^{12}})[r] \simeq E'(\mathbb{F}_{p^2})[r]$.

Miller loop.

k is even $\Longrightarrow$ no vertical lines.

$6 \mid k$ and $D = 3 \Longrightarrow$ twist of degree 6: $E(\mathbb{F}_{p^{12}})[r] \simeq E'(\mathbb{F}_{p^2})[r]$.

Final exponentiation.

$$\frac{p^{12}-1}{r} = (p^6-1)(p^2+1)\frac{p^4-p^2+1}{r}$$

Miller loop.
k is even $\Longrightarrow$ no vertical lines.
$6 \mid k$ and $D = 3 \Longrightarrow$ twist of degree 6: $E(\mathbb{F}_{p^{12}})[r] \simeq E'(\mathbb{F}_{p^2})[r]$.
Final exponentiation.

$$\frac{p^{12}-1}{r} = (p^6-1)(p^2+1)\frac{p^4-p^2+1}{r}$$

$y = (x^{p^6-1})^{p^2+1}$ is easy with Frobenius powers.

Miller loop.
k is even $\implies$ no vertical lines.
$6 \mid k$ and $D = 3 \implies$ twist of degree 6: $E(\mathbb{F}_{p^{12}})[r] \simeq E'(\mathbb{F}_{p^2})[r]$.
Final exponentiation.

$$\frac{p^{12}-1}{r} = (p^6-1)(p^2+1)\frac{p^4-p^2+1}{r}$$

$y = (x^{p^6-1})^{p^2+1}$ is easy with Frobenius powers.
$\frac{p^4-p^2+1}{r}$ is specific because $p = p(x_0)$ and $r = r(x_0)$.

Miller loop.
k is even $\Longrightarrow$ no vertical lines.
$6 \mid k$ and $D = 3 \Longrightarrow$ twist of degree 6: $E(\mathbb{F}_{p^{12}})[r] \simeq E'(\mathbb{F}_{p^2})[r]$.
Final exponentiation.

$$\frac{p^{12}-1}{r} = (p^6-1)(p^2+1)\frac{p^4-p^2+1}{r}$$

$y = (x^{p^6-1})^{p^2+1}$ is easy with Frobenius powers.
$\frac{p^4-p^2+1}{r}$ is specific because $p = p(x_0)$ and $r = r(x_0)$.
$y^{\frac{p(x_0)^4-p(x_0)^2+1}{r(x_0)}} = y^{p^3+\lambda_2(x_0)p^2+\lambda_1(x_0)p+\lambda_0(x_0)}$: few exponentiations by x_0.

Miller loop.
k is even $\Longrightarrow$ no vertical lines.
$6 \mid k$ and $D = 3 \Longrightarrow$ twist of degree 6: $E(\mathbb{F}_{p^{12}})[r] \simeq E'(\mathbb{F}_{p^2})[r]$.
Final exponentiation.

$$\frac{p^{12}-1}{r} = (p^6-1)(p^2+1)\frac{p^4-p^2+1}{r}$$

$y = (x^{p^6-1})^{p^2+1}$ is easy with Frobenius powers.
$\frac{p^4-p^2+1}{r}$ is specific because $p = p(x_0)$ and $r = r(x_0)$.
$y^{\frac{p(x_0)^4-p(x_0)^2+1}{r(x_0)}} = y^{p^3+\lambda_2(x_0)p^2+\lambda_1(x_0)p+\lambda_0(x_0)}$: few exponentiations by x_0.
Efficient pairing. But how secure are these curves ?

Security of pairing curves.

$$e : E(\mathbb{F}_p) \times E(\mathbb{F}_{p^k}) \longrightarrow \mathbb{F}_{p^k}$$

Security of pairing curves.

$$e : E(\mathbb{F}_p) \times E(\mathbb{F}_{p^k}) \longrightarrow \mathbb{F}_{p^k}$$

- Security against DLP in elliptic curve: best attack in $\mathcal{O}(\sqrt{r})$.
  $\log_2(r) = 256$ for 128 bits of security.

Security of pairing curves.

$$e : E(\mathbb{F}_p) \times E(\mathbb{F}_{p^k}) \longrightarrow \mathbb{F}_{p^k}$$

- Security against DLP in elliptic curve: best attack in $\mathcal{O}(\sqrt{r})$. $\log_2(r) = 256$ for 128 bits of security.
- Security against DLP in $\mathbb{F}_{p^k}$: Number Field Sieve attacks in progress.
  - special prime p $\Longrightarrow$ 1993: Special NFS attack
  - $k > 1$ $\Longrightarrow$ 2015: Tower NFS attack
  - composite k and special p $\Longrightarrow$ 2016: STNFS attack

Security of pairing curves.

$$e : E(\mathbb{F}_p) \times E(\mathbb{F}_{p^k}) \longrightarrow \mathbb{F}_{p^k}$$

- Security against DLP in elliptic curve: best attack in $\mathcal{O}(\sqrt{r})$.
  $\log_2(r) = 256$ for 128 bits of security.
- Security against DLP in $\mathbb{F}_{p^k}$: Number Field Sieve attacks in progress.
  special prime p $\Longrightarrow$ 1993: Special NFS attack
  $k > 1$ $\Longrightarrow$ 2015: Tower NFS attack
  composite k and special p $\Longrightarrow$ 2016: STNFS attack

  BN curves are threatened by STNFS... 
  Need a 5500 bits field $\mathbb{F}_{p^{12}}$ to get 128 bits of security.

Generation of curves with given prime k, square-free D and no structure on p.

Algorithm: $\textsc{Cocks-Pinch}(k, D)$ – Compute a pairing-friendly curve $E/\mathbb{F}_p$ of trace t with a subgroup of order r, such that $t^2 - Dy^2 = 4p$.

Set a prime r such that $k \mid r - 1$ and $\sqrt{-D} \in \mathbb{F}_r$
Set T such that $r \mid \Phi_k(T)$
$t \leftarrow T + 1$
$y \leftarrow (t - 2)/\sqrt{-D}$
Lift $t, y \in \mathbb{Z}$ such that $t^2 + Dy^2 \equiv 0 \mod 4$
$p \leftarrow (t^2 + Dy^2)/4$
if p is prime **then return** $[p, t, y, r]$ **else** Repeat with another r.

Generation of curves with given prime k, square-free D and no structure on p.

Algorithm: $\textsc{Cocks-Pinch}(k, D)$ – Compute a pairing-friendly curve $E/\mathbb{F}_p$ of trace t with a subgroup of order r, such that $t^2 - Dy^2 = 4p$.

Set a prime r such that $k \mid r - 1$ and $\sqrt{-D} \in \mathbb{F}_r$
Set T such that $r \mid \Phi_k(T)$
$t \leftarrow T + 1$
$y \leftarrow (t - 2)/\sqrt{-D}$
Lift $t, y \in \mathbb{Z}$ such that $t^2 + Dy^2 \equiv 0 \mod 4$
$p \leftarrow (t^2 + Dy^2)/4$
if p is prime **then return** $[p, t, y, r]$ **else** Repeat with another r.

Large trace $t \implies$ the ate pairing is not very efficient 

Generation of curves with given prime k, square-free D and no structure on p.

Algorithm: $\textsc{Cocks-Pinch}(k, D)$ – Compute a pairing-friendly curve $E/\mathbb{F}_p$ of trace t with a subgroup of order r, such that $t^2 - Dy^2 = 4p$.

Set a small T
Set a prime r such that $k \mid r-1$, $\sqrt{-D} \in \mathbb{F}_r$ and $r \mid \Phi_k(T)$
$t \leftarrow T + 1$
$y \leftarrow (t-2)/\sqrt{-D}$
Lift $t, y \in \mathbb{Z}$ such that $t^2 + Dy^2 \equiv 0 \mod 4$
$p \leftarrow (t^2 + Dy^2)/4$
if p is prime **then return** $[p, t, y, r]$ **else** Repeat with another r.

Large trace $t \implies$ the ate pairing is not very efficient

Fix: first fix a small T and then choose r. $t = T + 1$ is small

Generation of curves with given prime k, square-free D and no structure on p.

Algorithm: $\textsc{Cocks-Pinch}(k, D)$ – Compute a pairing-friendly curve $E/\mathbb{F}_p$ of trace t with a subgroup of order r, such that $t^2 - Dy^2 = 4p$.

Set a small T
Set a prime r such that $k \mid r-1$, $\sqrt{-D} \in \mathbb{F}_r$ and $r \mid \varphi_k(T)$
$t \leftarrow T + 1$
$y \leftarrow (t-2)/\sqrt{-D}$
Lift $t, y \in \mathbb{Z}$ such that $t^2 + Dy^2 \equiv 0 \mod 4$
$p \leftarrow (t^2 + Dy^2)/4$
if p is prime and $p = 1 \mod k$ **then return** $[p, t, y, r]$ **else** Repeat with another r.

Large trace $t \implies$ the ate pairing is not very efficient

Fix: first fix a small T and then choose r. $t = T + 1$ is small $\mathbb{F}_{p^k} = \mathbb{F}_p[u]/(u^k - \alpha)$

Parameter choices for 128 bits of security:

- Size of T.

Parameter choices for 128 bits of security:

- Size of T.
  $\log_2(r) \approx \log_2(\Phi_k(T)) = \varphi(k)\log_2(T) \Longrightarrow \log_2(T) = 256/\varphi(k)$.

Parameter choices for 128 bits of security:

- Size of T.
  $\log_2(r) \approx \log_2(\Phi_k(T)) = \varphi(k)\log_2(T) \Longrightarrow \log_2(T) = 256/\varphi(k)$.

$$k = 5, \log_2(T) = 64 \qquad k = 6, \log_2(T) = 52$$

$$k = 7, \log_2(T) = 43 \qquad k = 8, \log_2(T) = 37$$

Parameter choices for 128 bits of security:

- Size of T.
  $\log_2(r) \approx \log_2(\Phi_k(T)) = \varphi(k)\log_2(T) \Longrightarrow \log_2(T) = 256/\varphi(k)$.

$$k = 5, \log_2(T) = 64 \qquad k = 6, \log_2(T) = 52$$

$$k = 7, \log_2(T) = 43 \qquad k = 8, \log_2(T) = 37$$

- Low hamming weight of T (Miller loop).

Parameter choices for 128 bits of security:

- Size of T.
  $\log_2(r) \approx \log_2(\Phi_k(T)) = \varphi(k)\log_2(T) \Longrightarrow \log_2(T) = 256/\varphi(k)$.

$$k = 5, \log_2(T) = 64 \qquad k = 6, \log_2(T) = 52$$

$$k = 7, \log_2(T) = 43 \qquad k = 8, \log_2(T) = 37$$

- Low hamming weight of T (Miller loop).
- When lifting in $\mathbb{Z}$, add a multiple of r in y

$$y = y + h_y \cdot r$$

such that p is large enough to resist NFS attacks.

128-bit security for finite field extensions.

128-bit security for finite field extensions.

Our variant of Cocks-Pinch generates pairing-friendly curves with a "non-special" prime: p is not parametrized by a (one variable) polynomial with small coefficents.

128-bit security for finite field extensions.

Our variant of Cocks-Pinch generates pairing-friendly curves with a "non-special" prime: p is not parametrized by a (one variable) polynomial with small coefficents.

Remark

Sometimes (for instance $k = 8$) p is parametrized by a *two-variables* polynomial: $p \in \mathbb{Z}[T, h_y]$, but today NFS-variants do not use this property.

128-bit security for finite field extensions.

Our variant of Cocks-Pinch generates pairing-friendly curves with a "non-special" prime: p is not parametrized by a (one variable) polynomial with small coefficents.

Remark

Sometimes (for instance $k = 8$) p is parametrized by a *two-variables* polynomial: $p \in \mathbb{Z}[T, h_y]$, but today NFS-variants do not use this property.

Field	DL attack	Field size needed for 128-bit security	$\log_2(p)$ induced
$\mathbb{F}_{p^5}$	TNFS	3320	664
$\mathbb{F}_{p^6}$	exTNFS	4032	672
$\mathbb{F}_{p^7}$	TNFS	3584	512
$\mathbb{F}_{p^8}$	exTNFS	4352	544

Timings and comparisons

RELIC. `https://github.com/relic-toolkit/relic.git`

RELIC. `https://github.com/relic-toolkit/relic.git`

Efficient library for cryptography

RELIC. `https://github.com/relic-toolkit/relic.git`

Efficient library for cryptography, state of the art for pairing computation.

RELIC. https://github.com/relic-toolkit/relic.git

Efficient library for cryptography, state of the art for pairing computation.

Implementation for BN and BLS curves

RELIC. `https://github.com/relic-toolkit/relic.git`

Efficient library for cryptography, state of the art for pairing computation.

Implementation for BN and BLS curves

Fast $\mathbb{F}_p$ arithmetic in assembly instructions for some given p.

RELIC. `https://github.com/relic-toolkit/relic.git`

Efficient library for cryptography, state of the art for pairing computation.

Implementation for BN and BLS curves

Fast $\mathbb{F}_p$ arithmetic in assembly instructions for some given p.

How to compare curves.

RELIC. `https://github.com/relic-toolkit/relic.git`

Efficient library for cryptography, state of the art for pairing computation.

Implementation for BN and BLS curves

Fast $\mathbb{F}_p$ arithmetic in assembly instructions for some given p.

How to compare curves.

1. Bench $\mathbb{F}_p$ arithmetic and pairing computation for new BN and BLS primes.

RELIC. `https://github.com/relic-toolkit/relic.git`

Efficient library for cryptography, state of the art for pairing computation.

Implementation for BN and BLS curves

Fast $\mathbb{F}_p$ arithmetic in assembly instructions for some given p.

How to compare curves.

1. Bench $\mathbb{F}_p$ arithmetic and pairing computation for new BN and BLS primes.
2. Bench $\mathbb{F}_p$ arithmetic for our non-special primes of different sizes.

RELIC. https://github.com/relic-toolkit/relic.git

Efficient library for cryptography, state of the art for pairing computation.

Implementation for BN and BLS curves

Fast $\mathbb{F}_p$ arithmetic in assembly instructions for some given p.

How to compare curves.

1. Bench $\mathbb{F}_p$ arithmetic and pairing computation for new BN and BLS primes.
2. Bench $\mathbb{F}_p$ arithmetic for our non-special primes of different sizes.
3. Count the number of $\mathbb{F}_p$ multiplications to get an estimation of the cost.

New curves for 128 bits of security.

We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.
We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.
We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.

We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.
We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.

We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.

We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

New curves for 128 bits of security.
We generate curves of embedding degree $5, 6, 7$ and 8 with the previous algorithm.

Curve	this work				BN	BLS	–
k	5	6	7	8	12	12	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	3072
$\log_2(p)$	664	672	512	544	462	461	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	42.7ms

Thank you for your attention.

Thank you for your attention.

Thank you for your attention.

Thank you for your attention.

Thank you for your attention.

Thank you for your attention.

Thank you for your attention.

Curve	this work				BN	BLS	KSS	–
k	5	6	7	8	12	12	16	1
$\mathbb{F}_{p^k}$ *size*	3320	4032	3584	4352	5544	5532	5424	3072
$\log_2(p)$	664	672	512	544	462	461	339	3072
$\mathbb{F}_p$ mul.	230ns	230ns	130ns	154ns	130ns	130ns	69ns	4882ns
Miller length	64-bit	128-bit	43-bit	64-bit	117-bit	77-bit	35-bit	256-bit
Mill. field	3320	672	3584	1088	924	922	1356	3072
Miller step	3.4ms	1.1ms	2.1ms	0.7ms	1.6ms	1.0ms	0.5ms	22.7ms
Expo. step	2.5ms	0.9ms	1.9ms	1.0ms	0.7ms	0.8ms	1.3ms	20.0ms
Total	5.9ms	2.0ms	4.0ms	1.7ms	2.3ms	1.8ms	1.8ms	42.7ms