Some BATs with $\mathtt{mp}\mathbb{F}_q$

Emmanuel Thomé, Pierrick Gaudry

INRIA LORRAINE

CACAO

BATs

We present here data for two BATs :

- Bernstein's `curve25519` implemented with $\texttt{mp}\mathbb{F}_q$.
- `surf127eps` (also with $\texttt{mp}\mathbb{F}_q$).

curve25519 + $\mathtt{mp}\mathbb{F}_q$

curve25519 is described by Bernstein ; see paper & talks.

- $y^2 = x^3 + 486662x^2 + x$ over $\mathbb{F}_p$, $p = 2^{255} - 19$.
- We work with the x-coordinate only.

  Exponentiation uses Montgomery ladder.
- Both the curve and its twist have order tiny $\times$ BIGprime.
- $\Rightarrow$ any 128-bit bitstring is a valid x-coordinate.
- SIZE(pubkey) = SIZE(seckey) = 32 bytes.

Results with `curve25519` + $\texttt{mp}\mathbb{F}_q$

We only implemented a key sharing BAT ; `keypair` + `sharedsecret`.

Time for an exponentiation, comparing apples and oranges :

- Reference implementation on 32-bit machines using FPU (djb)

  Athlon : 620,000 cycles ; Pentium-M : 640,000 cycles.

- With $\texttt{mp}\mathbb{F}_q$ on 64-bit machines using integer arithmetic.

  Opteron : 307,000 cycles ; Core2 : 386,000 cycles.

surf127eps

This is genus 2 curve without a curve :

- The Jacobian of a genus 2 curve is a projective variety of dimension 2.
- Here we handle the variety directly as a projective surface :

  Projective tuples $(x : y : z : t)$ satisfying $f(x, y, z, t) = 0$.
- Well, almost : opposite points are bundled together.

Reference : Gaudry.

Hadamard transform

We meet several times the following transform :

$$H(a,b,c,d) = \begin{pmatrix} a+b+c+d, \\ a+b-c-d, \\ a-b+c-d, \\ a-b-c+d \end{pmatrix}$$

Definition of the Kummer surface

Input : $\mathbb{F}_q$ (here q odd) + (almost) arbitrary scalars a, b, c, d.

We define the surface $\mathcal{K}(a, b, c, d)$.

$$(a, b, c, d) \rightsquigarrow (E, F, G, H).$$

$$\begin{aligned}\mathcal{K}: \quad x^4 + y^4 + z^4 + t^4 + 2Exyzt &= F \cdot (x^2t^2 + y^2z^2) \\ &+ G \cdot (x^2z^2 + y^2t^2) \\ &+ H \cdot (x^2y^2 + z^2t^2).\end{aligned}$$

The equation is only needed for finding a random point on $\mathcal{K}$, and validation.

Doubling on $\mathcal{K}$

Doubling is well-defined on $\mathcal{K}$.

Define $(4A^2, 4B^2, 4C^2, 4D^2) = H(a^2, b^2, c^2, d^2)$.

Let $P = (x : y : z : t)$ a point on $\mathcal{K}$.

$$(\lambda_0 : \lambda_1 : \lambda_2 : \lambda_3) := H(x^2 : y^2 : z^2 : t^2),$$
$$(\hat{x} : \hat{y} : \hat{z} : \hat{t}) := (\lambda_0/A : \lambda_1/B : \lambda_2/C : \lambda_3/D)$$

$$(\mu_0 : \mu_1 : \mu_2 : \mu_3) := H(\hat{x}^2 : \hat{y}^2 : \hat{z}^2 : \hat{t}^2),$$
$$(x' : y' : z' : t') := (\mu_0/a : \mu_1/b : \mu_2/c : \mu_3/d).$$

Then $2P$ is $(x' : y' : z' : t')$.

(Pseudo)-addition on $\mathcal{K}$

P and $-P$ are bundled in $\mathcal{K}$: addition is defined only up to sign.

We have a formula for $\{P, Q\} \to \{P + Q, P - Q\}$.

Let $P = (x : y : z : t)$ and $Q = (\bar{x} : \bar{y} : \bar{z} : \bar{t})$.

$$(x : y : z : t) \rightsquigarrow (\hat{x} : \hat{y} : \hat{z} : \hat{t}),$$

$$(\bar{x} : \bar{y} : \bar{z} : \bar{t}) \rightsquigarrow (\hat{\bar{x}} : \hat{\bar{y}} : \hat{\bar{z}} : \hat{\bar{t}}),$$

$$(x_+x_- : y_+y_- : z_+z_- : t_+t_-) = H(\bar{x}\hat{\bar{x}} : \bar{y}\hat{\bar{y}} : \bar{z}\hat{\bar{z}} : \overline{t\hat{t}}).$$

Cutting down operation count

These formulae are amenable to optimizations.

- It is possible to work with squares of coordinates only.
- While computing $\hat{x}^2 = \lambda_0^2/A^2$ and $\hat{x}\hat{\bar{x}} = \lambda_0\hat{\lambda}_0/A^2$, rather multiply by $1/A^2$ only once : $1sM + 2M$ vs $2sM + M + S$.
- Everything being projective, replace $(1/a : 1/b : 1/c : 1/d) = (1 : acd : abd : abc)$.
- In the context of a Montgomery ladder, $(x_- : \cdots)$ is the base point : we can be choose $x_- = 1$.

Operation count per bit of exponent

Cost of $(nP, (n+1)P) \ldots (n'P, (n'+1)P)$ with $n' = 2n$ or $2n+1$:

- 9S+10M+6sM for minimal operation count.
- 12S+7M+9sM might be preferred is S is fast and sM is for free.
- sM : multiplication by « small » $a^2b^2c^2$, $A^2B^2C^2$, ...

  We can freely choose a, b, c, d so these can be made small. Really ?

Problem : In genus 2, can't count for arbitrary curves (record is 162 bits).

- Either work hard to improve point counting.
- Or use CM and have a, b, c, d large, and less control over q.

What's in `surf127eps`

`surf127eps` is built on a genus 2 CM curve (by $\mathbb{Q}\left(i\sqrt{5+\sqrt{53}}\right)$)

CM theory gives q and $\#\mathcal{K}$.

- Here $\#\mathcal{K} = 16 \cdot (\text{250-bit prime})$. $\Rightarrow$ « security » 2^{125}.
- Secret key : an integer ; $\text{SIZE}(\text{seckey}) = 32\text{bytes}$.
- Public key : a point. $(1 : y : z : t) \Rightarrow \text{SIZE}(\text{pubkey}) = 48\text{bytes}$.
  (could compress to $(1 : y : z : \cdot)$; t is the root of a degree 4 eqn).

CACAO

Results with `surf127eps`

We only implemented a key sharing BAT ; `keypair` + `sharedsecret`.

Time for an scalar multiplication :

- Opteron : 279,000 cycles (`curve25519` : 307,000)

  8.6 kMUL/s (7.8 kMUL/s) @ 2.4GHz

- Core2 : 410,000 cycles (`curve25519` : 386,000)

  6.5 kMUL/s (6.9 kMUL/s) @ 2.667GHz

- NB : significantly better than the BAT from 20070214.

- Can be improved with small a, b, c, d. Possible using curves having real multiplication by $\sqrt{2}$. This imposes a nice constraint on a, b, c, d.