Elliptic curves in Nemo

Jean Kieffer

École normale supérieure de Paris & INRIA

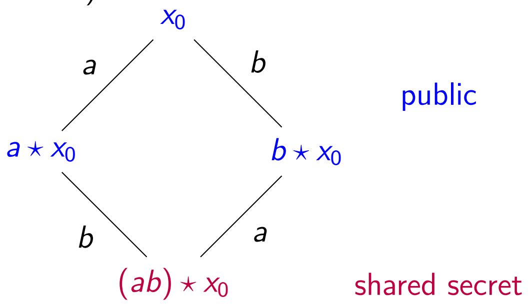
August 3, 2017

Key exchange from hard homogeneous spaces

Let G be an abelian group acting on a set X with some given point x_0. If the action is

- easy to compute (polynomial time),
- hard to invert (exponential time),

then there is an analogue of the Diffie–Hellman key exchange (Couveignes 2006).



The Couveignes–Rostovtsev–Stolbunov scheme

Question

Where can we find such an action?

The Couveignes–Rostovtsev–Stolbunov scheme

Question

Where can we find such an action?

Answer (Couveignes 2006, Rostovtsev–Stolbunov 2006)

Use the action of a class group on a set of isogenous elliptic curves.

The Couveignes–Rostovtsev–Stolbunov scheme

Question

Where can we find such an action?

Answer (Couveignes 2006, Rostovtsev–Stolbunov 2006)

Use the action of a class group on a set of isogenous elliptic curves.

Goals

- Explain what this means
- Describe the computations needed
- Discuss our EllipticCurves module in Nemo.

Elliptic curves over k

- *Elliptic curves* over a field k are algebraic curves, e.g.

  $$E \ : \ y^2 = x^3 + ax + b.$$

  They have an abelian group structure. The j-invariant

  $$j(E) = 1728 \frac{4a^3}{4a^3 + 27b^2}$$

  classifies such curves up to isomorphism.
- *Isogenies* are nonzero morphisms. Our isogenies will be defined over k. If an isogeny is given by rational fractions of degree ℓ, it is called an ℓ-isogeny.

Complex multiplication

From now on, $k = \mathbb{F}_p$ is a prime finite field.
Let $E/\mathbb{F}_p$ be an ordinary elliptic curve.

- The ring $\mathrm{End}(E)$ is isomorphic to an order in a quadratic number field. The Frobenius endomorphism is a distinguished element in $\mathrm{End}(E)$.

Complex multiplication

From now on, $k = \mathbb{F}_p$ is a prime finite field.
Let $E/\mathbb{F}_p$ be an ordinary elliptic curve.

- The ring $\mathrm{End}(E)$ is isomorphic to an order in a quadratic number field. The Frobenius endomorphism is a distinguished element in $\mathrm{End}(E)$.
- Ideals of $\mathcal{O}$ modulo principal ideals form the *class group* of $\mathcal{O}$.

Isogenies of degree ℓ starting from E correspond to ideals in $\mathcal{O}$ of norm ℓ.
For example, in the generic case, there are either zero or two isogenies of degree ℓ with domain E.

Action of the class group

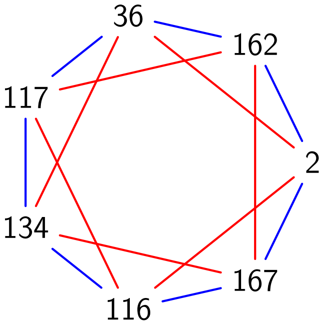
Proposition

- There is an action of the class group on a set of elliptic curves.
- Ideals of norm ℓ act as ℓ-isogenies.
- This action is simply transitive.

Therefore, in our setting, isogeny graphs are just Cayley graphs of a certain group.

Our isogeny graphs

Isogeny graph over $\mathbb{F}_{173}$ with isogenies of degree 3 (blue) and 7 (red):



This graph is *much* larger for cryptographic uses.

Representing isogenies

Let E/k be an elliptic curve, and $\ell \neq p$ be an odd prime. Giving the following is equivalent:

- An isogeny $E \to E'$ of degree ℓ
- Its kernel, which is a cyclic subgroup of E of order ℓ
- A polynomial of degree $\frac{\ell-1}{2}$ in x defining the kernel.

If we know this *kernel polynomial*, we can easily find E' using Vélu's formulas.

Representing ideals

We do *not* compute directly in the class group. Instead, we use the following representation of ideals:

If the ideal $\mathfrak{l}$ has norm ℓ, we have a natural surjection

$$\mathcal{O}/\ell\mathcal{O} \to \mathcal{O}/\mathfrak{l}\mathcal{O} \simeq \mathbb{Z}/\ell\mathbb{Z}.$$

The ideal ℓ is determined by the tuple (ℓ, v), where v is the image of the Frobenius under this surjection. We call v a *Frobenius eigenvalue*.

General algorithm

Problem

Given $E/\mathbb{F}_p$ and a prime ℓ, how can we compute the action of an ideal (ℓ, v) on E ?

General algorithm

Problem

Given $E/\mathbb{F}_p$ and a prime ℓ, how can we compute the action of an ideal (ℓ, v) on E ?

Idea

The j-invariant we want is one of the two roots of a polynomial equation, called *modular equation*: $\Phi_\ell(j(E), Y) = 0$.

General algorithm

Problem

Given $E/\mathbb{F}_p$ and a prime ℓ, how can we compute the action of an ideal (ℓ, v) on E ?

Idea

The j-invariant we want is one of the two roots of a polynomial equation, called *modular equation*: $\Phi_\ell(j(E), Y) = 0$.

Algorithm

Let E be a curve and (ℓ, v) be an ideal.

- compute and solve this equation: find j_1, j_2
- compute the kernel polynomial $K(x)$ of $E \to j_1$
- check if the Frobenius acts on it as scalar mult. by v:

$$(x^p, y^p) \stackrel{?}{=} [v] \cdot (x, y) \quad \text{mod } K(x) \text{ and curve equation.}$$

Kernel computation

Question

How can we compute the kernel polynomial $K(x)$ of $\phi : E \to j_1$?

Kernel computation

Question

How can we compute the kernel polynomial $K(x)$ of $\phi : E \to j_1$?

Idea (Elkies)

The rational fraction defining ϕ satisfies a simple differential equation. $K(x)$ appears as the denominator.

Kernel computation

Question

How can we compute the kernel polynomial $K(x)$ of $\phi : E \to j_1$?

Idea (Elkies)

The rational fraction defining ϕ satisfies a simple differential equation. $K(x)$ appears as the denominator.

Algorithm (Bostan–Morain–Salvy–Schost 2008)

- Compute power series solutions of this ODE up to a certain precision with a Newton iteration
- Recover $K(x)$ using the Berlekamp–Massey rational reconstruction algorithm.

Using Vélu's formulas

Problem

Given $E/\mathbb{F}_p$ and a prime $\ell \neq p$, how can we compute the curves linked to E by an ℓ-isogeny?

Finding roots of modular polynomials is costly : $\Phi_\ell(X, Y)$ has degree $\ell + 1$ in both variables.

Using Vélu's formulas

Problem

Given $E/\mathbb{F}_p$ and a prime $\ell \neq p$, how can we compute the curves linked to E by an ℓ-isogeny?

Finding roots of modular polynomials is costly : $\Phi_\ell(X, Y)$ has degree $\ell + 1$ in both variables.

Another solution

Suppose that K is a subgroup of order ℓ in E whose points are defined over $\mathbb{F}_p$.

- Look for ℓ-torsion points over $\mathbb{F}_p$ to find K, using scalar multiplications
- Compute the curve E/K using Vélu's formulas.

The isogeny $E \to E/K$ has degree ℓ.

Using Vélu's formulas (2)

- The previous condition may be relaxed when allowing field extensions. But. . .

Using Vélu's formulas (2)

- The previous condition may be relaxed when allowing field extensions. But...
- Using Vélu's formulas is only efficient with small-degree extensions.

Using Vélu's formulas (2)

- The previous condition may be relaxed when allowing field extensions. But. . .
- Using Vélu's formulas is only efficient with small-degree extensions.
- Using efficient arithmetic on curves is important (use other models than Weierstrass equations)

Using Vélu's formulas (2)

- The previous condition may be relaxed when allowing field extensions. But...
- Using Vélu's formulas is only efficient with small-degree extensions.
- Using efficient arithmetic on curves is important (use other models than Weierstrass equations)
- Not every curve satisfies the previous conditions for many ℓ's and small d's: we have to look for adequate curves.

Using Vélu's formulas (2)

- The previous condition may be relaxed when allowing field extensions. But...
- Using Vélu's formulas is only efficient with small-degree extensions.
- Using efficient arithmetic on curves is important (use other models than Weierstrass equations)
- Not every curve satisfies the previous conditions for many ℓ's and small d's: we have to look for adequate curves.
- In practice, we have to use both the general algorithm and Vélu's formulas.

What we would like to do

For the general method:

- Define elliptic curves over finite fields and general rings
- Define isogenies, scalar multiplication and isomorphisms
- Have a database of modular polynomials
- Find roots of polynomials over finite fields
- BMSS: ODEs in power series with Newton iterations and Berlekamp–Massey.

What we would like to do

For the general method:

- Define elliptic curves over finite fields and general rings
- Define isogenies, scalar multiplication and isomorphisms
- Have a database of modular polynomials
- Find roots of polynomials over finite fields
- BMSS: ODEs in power series with Newton iterations and Berlekamp–Massey.

For Vélu's formulas:

- Define points on elliptic curves and arithmetic operations with efficient models
- Extensions of finite fields.

What we would like to do

For the general method:

- Define elliptic curves over finite fields and general rings
- Define isogenies, scalar multiplication and isomorphisms
- Have a database of modular polynomials
- Find roots of polynomials over finite fields
- BMSS: ODEs in power series with Newton iterations and Berlekamp–Massey.

For Vélu's formulas:

- Define points on elliptic curves and arithmetic operations with efficient models
- Extensions of finite fields.

Three ways to compute scalar multiplications

Three ways to compute scalar multiplications

Sol. 1 (Nemo)

```
E = Weierstrass(...)
Fext, _ = FiniteField(p, d, "alpha")
Eext = base_extend(E, Fext)
P = rand(Eext)
p^d * P
```

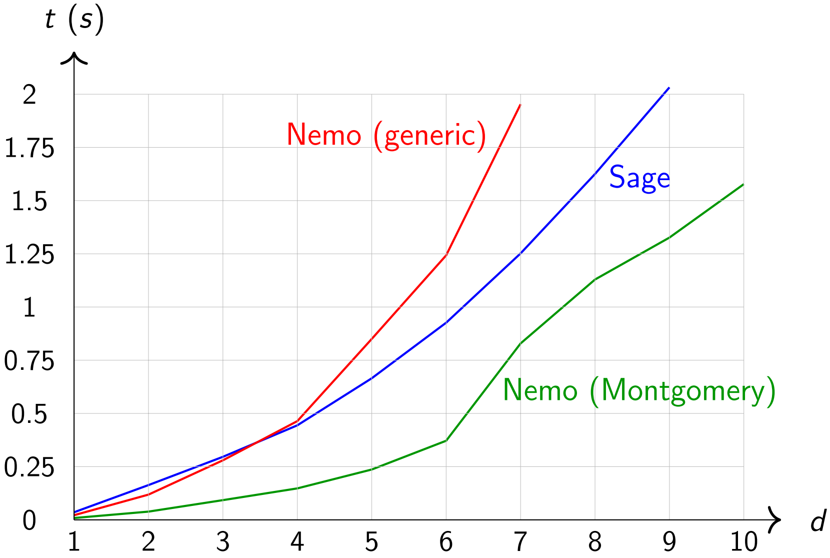
Sol. 2 (Nemo)

```
E = Montgomery(...)
Fext, _ = FiniteField(p, d, "alpha")
Eext = base_extend(E, Fext)
P = randXonly(Eext)
p^d * P
```

Sol. 3 (Sage)

```
E = EllipticCurve(...)
Fext = FiniteField(p**d, "alpha")
Eext = E.base_extend(Fext)
P = Eext.random_element()
C = p**d
C * P
```

Timing results



Further possible development

Around the previous algorithms:

- Call (system) PARI to compute the cardinality of curves over finite fields
- Have access to FLINT's root finding algorithms modulo p
- Have a decent system to handle field extensions
- Have p-adic numbers to compute isogenies in small characteristic?
- Connections with Hecke to be able to compute in endomorphism rings?

Further possible development

This module may also become useful to people learning about elliptic curves and elliptic curve cryptography:

- Implement other models for curves
- Add pairings
- ...

Conclusion

- We implemented Couveigne's proposal, but the heavy computations needed makes it uncompetitive in practive when compared with other cryptosystems.

Conclusion

- We implemented Couveigne's proposal, but the heavy computations needed makes it uncompetitive in practive when compared with other cryptosystems.
- In order to use Vélu's formulas, we have to look for adequate curves, and this requires lots of computational power.

Conclusion

- We implemented Couveigne's proposal, but the heavy computations needed makes it uncompetitive in practive when compared with other cryptosystems.
- In order to use Vélu's formulas, we have to look for adequate curves, and this requires lots of computational power.
- With the best curve we found so far, aiming at 128-bit security, we reduced the computing time from 880 to 360 seconds. Better curves would bring further improvement.

Conclusion

- We implemented Couveigne's proposal, but the heavy computations needed makes it uncompetitive in practive when compared with other cryptosystems.
- In order to use Vélu's formulas, we have to look for adequate curves, and this requires lots of computational power.
- With the best curve we found so far, aiming at 128-bit security, we reduced the computing time from 880 to 360 seconds. Better curves would bring further improvement.
- The EllipticCurves module is able to perform these computations.

Thank you!