Agence nationale de la sécurité des systèmes d'information

Université Denis Diderot - Paris 7

The SEA algorithm in PARI/GP

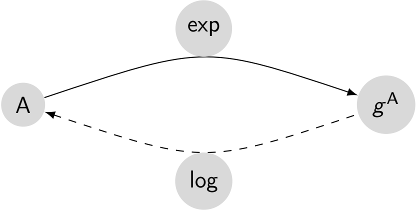
Julien Keuffer

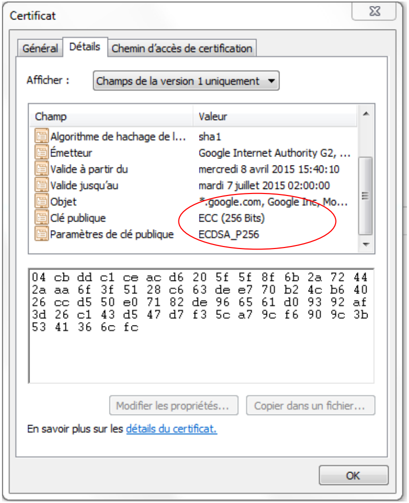
Internship advisors : Jean-Pierre Flori and Jérôme Plût

Tuesday 24th November 2015

- Diffie-Hellman key-exchange protocol
- El-Gamal cryptosystem
- Electronic signature

Security related to hardness of the discrete logarithm problem





A Google certificate

The discrete logarithm problem (DLP)

Generic attacks on discrete logarithm use at least $O(\sqrt{\#G_1})$ operations in G, where $\#G_1$ is the largest prime factor of $\#G$.

- Multiplicative group of finite fields : subexponential methods to compute logarithm.
- Elliptic curves : no known algorithm doing better for *general elliptic curves*

DLP on elliptic curves defined over $\mathbb{F}_p$

Faster methods exist for special classes of elliptic curves in which DLP can be transported to a group where it is easier to solve :

- MOV/Frey-Rück attack : transport DLP in $\mathbb{F}_q$ where $q = p^t$ and t is the smallest integer such that $p^t = 1 \mod \#E(\mathbb{F}_p)$
- Anomalous attack : $\#E = p$, DLP can be transported to $\mathbb{Z}/p\mathbb{Z}$

Why compute the number of points of an elliptic curve ?

- To ensure the difficulty of the DLP.
- Some protocols (*e.g.* ECDSA) need $\#E$ for calculations.

Finding an elliptic curve suitable for cryptography requires a lot of computations.

$\longrightarrow$ need to have a fast point counting algorithm.

PARI/GP

- SEA algorithm implemented in a PARI module : `ellsea.c`.
- Used in GP via the `ellcard()` function.
- Implementation based on Reynald Lercier's thesis (1997).
- Improvement have been proposed since.

My internship's goal

Study, implementation within PARI/GP and comparison of two articles :

- « Computing the eigenvalue in the Schoof-Elkies-Atkin algorithm using Abelian lifts » (Mihăilescu, Morain & Schost),
- « Fast algorithms for computing the eigenvalue in the Schoof-Elkies-Atkin algorithm » (Gaudry & Morain).

Schoof's algorithm

First polynomial algorithm published by Schoof in 1985.
Led to cryptography based on elliptic curves randomly selected.

Basic idea of the algorithm ($\mathbb{K} = \mathbb{F}_p$, $p > 3$, $E : y^2 = x^3 + Ax + B$) :

- Frobenius's endomorphism $\varphi : (x, y) \mapsto (x^p, y^p)$ verifies :
  $\varphi^2 - t\varphi + p\, id_E = 0$,
  t is called the *trace* of φ and is linked to $\#E(\mathbb{F}_p)$ by :

$$\boxed{\#E(\mathbb{F}_\mathrm{p}) = p + 1 - t \quad \text{and} \quad |t| \leqslant 2\sqrt{p}}$$

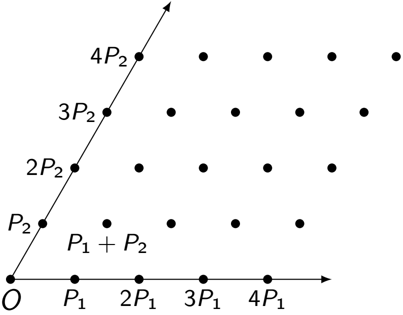
- $t \bmod \ell$ is computed for small primes ℓ,
- one is able to compute t as soon as $\prod \ell > 4\sqrt{p}$,
- number of ℓ required : $O(\log p)$, size of ℓ used : $O(\log p)$

Computation of $t \mod \ell$

- Calculations are done in $E[\ell] = \{P \in E(\overline{\mathbb{F}}_p) \text{ tq } [\ell]P = \mathcal{O}\}$ This group contains ℓ^2 points whose coordinates live in $\overline{\mathbb{F}}_p$ (for $\ell \neq p$)
- $E[\ell]$ is described by a polynomial ψ_ℓ : roots of ψ_ℓ are abscissae of $E[\ell]$ points,
- for $P \in E[\ell]$, $t \mod \ell$ is the value such that :

$$\varphi^2(P) + [p \bmod \ell]P = [t \bmod \ell]\,\varphi(P)$$

- $\deg \psi_\ell = \frac{\ell^2-1}{2} = O(\ell^2)$



Representation of $E[5]$

Schoof's algorithm

Computation of t mod ℓ

To search $t \mod \ell$, let $P \in E[\ell]$ and try all the values $\tau \in [\![0, \ell-1]\!]$ until the following relation holds :

$$\varphi^2(P) + [p \bmod \ell]P = [\tau]\varphi(P), \qquad (1)$$

- *A priori*, ℓ-torsion point coordinates belong to $\overline{\mathbb{F}}_p$,
  $\rightarrow$ must work with abstract ℓ-torsion represented by :

$$\mathcal{A}_\psi = \mathbb{F}_p[x, y]/(\psi_\ell(x), y^2 - x^3 - Ax - B).$$

- In $\mathcal{A}_\psi$, $P = (x, y)$ is a ℓ-torsion point and the equality (1) becomes :

$$(x^{p^2}, y^{p^2}) + [p \mod \ell](x, y) = [\tau](x^p, y^p) \qquad (2)$$

Schoof's algorithm

Schoof's algorithm complexity

Exponentiation dominates complexity in the algorithm

- $\ell = O(\log p)$, using $O(\log p)$ ℓ,
- for a given ℓ, computations of x^p and x^{p^2} modulo ψ_ℓ : $O(\ell^4 \log^3 p)$,
- *idem* for y^p and y^{p^2},

$\longrightarrow$ **complexity in $O(\log^8 p)$.**

Too much for an efficient use in cryptography.

Improvements by Elkies and Atkin (1)

Diagonalize the Frobenius

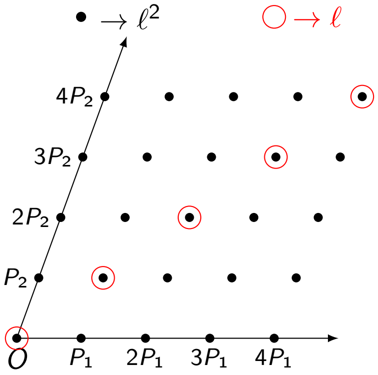
- $\varphi_{|E[\ell]}$ can be represented by a 2×2 matrix,
- The characteristic polynomial of $\varphi_{|E[\ell]}$ is $x^2 - tx + p \mod \ell$, its discriminant is $\Delta_\ell = t^2 - 4p \mod \ell$,
- case Δ_ℓ is a nonzero square in $\mathbb{F}_\ell$ then :
  - $\varphi_{|E[\ell]}$ is diagonalizable,
  - working on a one-dimensional eigenspace,
  - computing one eigenvalue λ is enough (because $t = \lambda + \frac{p}{\lambda} \mod \ell$)
- case $\Delta_\ell = 0$: $\Delta_\ell = 0 \Leftrightarrow t^2 = 4p \mod \ell$ so $t = \pm 2\sqrt{p} \mod \ell$,
- case Δ_ℓ is not a square : only a subset of possible values for t
- determination of whether Δ_ℓ is a square in $\mathbb{F}_\ell$ can be deduced from the splitting type of the ℓ-th modular polynomial : *not the topic*

A prime number ℓ such that $\varphi_{|E[\ell]}$ is diagonalizable is an *Elkies prime*, otherwise ℓ is an *Atkin prime*.

Working on an eigenspace of $E[\ell]$

- for an Elkies prime ℓ one can compute a subgroup of $E[\ell]$, denoted C_λ (one-dimensional eigenspace of $\varphi_{|E[\ell]}$)
- C_λ is described by a polynomial f_ℓ which divides ψ_ℓ,
- $\deg f_\ell = O(\ell)$ $\quad(\deg \psi_\ell = O(\ell^2))$
- let $P \in C_\lambda$, λ is the value such that :

$$\varphi(P) = [\lambda]P$$



Subgroup of $E[5]$

Improvement by Elkies and Atkin (3)

- Let $P \in C_\lambda$, searching λ such that : $\varphi(P) = [\lambda]P$,
  $\rightarrow$ working in $\mathcal{A}_f = \mathbb{F}_p[x,y]/(f_\ell(x), y^2 - x^3 - Ax - B)$.
- Search of $\lambda \in [\![1, \ell - 1]\!]$ such that : $(x^p, y^p) = [\lambda](x, y)$.

Complexity

- **Exponentiation dominates complexity** :
  Computations mod f_ℓ instead of mod ψ_ℓ
  $(\deg f_\ell = O(\ell),\ \deg \psi_\ell = O(\ell^2))$
  $\longrightarrow O(\log^6 p)$
- Computation of f_ℓ costs $O(\ell^2 \log^2 p)$

Optimised search of the eigenvalues

In the following, ℓ is an Elkies prime

Three different algorithms for eigenvalue search

- Implemented in PARI : exhaustive search.
- Optimisation 1 : baby-step giant-step algorithm(Gaudry-Morain),
- Optimisation 2 : MMS algorithm (Mihăilescu-Morain-Schost)

Implemented in PARI : exhaustive search

Principle

- only testing ordinates
  - opposite of a point is free : $P = (x, y) \Rightarrow [-1]P = (x, -y)$
  - $([i]P)_y = y \cdot (P_{i,y}(x)) \longrightarrow$ only using x,
  - Frobenius computation : $y^{p-1} = (x^3 + Ax + B)^{\frac{p-1}{2}}$
- $y^p \stackrel{?}{=} \pm(P)_y \; , \; y^p \stackrel{?}{=} \pm([2]P)_y \; , \; \dots \; , \; y^p \stackrel{?}{=} \pm([\frac{\ell-1}{2}]P)_y$

$$\longrightarrow O(\ell) \text{ operations in the curve to find } \lambda$$

Baby-step giant-step

Principle

- time-memory trade-off,
- search $1 \leqslant i, \pm j \leqslant \lceil \sqrt{\ell} \rceil$ such that : $[i]\varphi(P) = [j]P$ with $P \in C_\lambda$
- if a collision is found : $\lambda = j/i \mod \ell$
- Algorithm :
  - precompute and store multiples of P
  - compute multiples of $\varphi(P)$ and search for a collision in the table of multiples of P.
  - find the sign of the eigenvalue

$$\longrightarrow O(\sqrt{\ell}) \text{ operations in the curve to find } \lambda$$

(but need to store $O(\sqrt{\ell})$ abscissae)

Implementation of baby-step giant-step

- calculations in projective coordinates :
  - only computing *abscissae* of multiples and using division polynomials for calculations
  - abscissae are fractions $\frac{a_i}{b_i}$, storing couples (a_i, b_i),
  - equality test between two fractions (*ie* collision) evaluated with a linear form
- collision found $\longrightarrow \lambda$ known up to sign :
  - $\ell \equiv 1 \mod 4$: need to compute ordinates of the collision points to determinate the sign
    Gaudry-Morain propose a method to recover x^p from y^p with a gcd computation whose cost is inferior to the cost of computing x^p and y^p.
  - $\ell \equiv 3 \mod 4$: conclusion with Dewaghe's formula .

Dewaghe's formula

Let $\ell \equiv 3 \mod 4$, λ_0 be the eigenvalue known up to sign and r be the resultant of f_ℓ and $x^3 + Ax + b$, then :

$$\lambda = \left(\frac{\lambda_0}{\ell}\right)\left(\frac{r}{p}\right)\lambda_0. \tag{3}$$

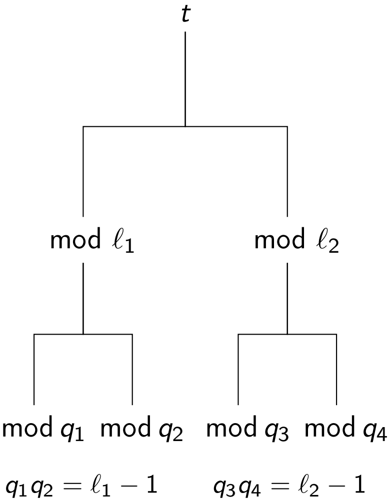
Thus, to obtain the eigenvalue one only needs to :

- compute a resultant between a degree $\frac{\ell-1}{2}$ polynomial and a degree 3 polynomial
- compute two Legendre symbols
- apply formula (3).

MMS

Principle

- $\lambda \in (\mathbb{F}_\ell)^* \Rightarrow \log(\lambda) \in \mathbb{Z}/(\ell-1)\mathbb{Z}$,
- $q_1 q_2 = \ell - 1$, $\gcd(q_1, q_2) = 1$
- search for $\log(\lambda) \bmod q_1$,
- search for $\log(\lambda) \bmod q_2$,
- $\log(\lambda) \bmod \ell - 1$ is computed with the CRT and λ is obtained.
- intensive use of modular composition



Computation of $q \mid \ell - 1$, q odd :

Let $\mathcal{A}_\lambda = \mathbb{F}_p[X]/(f_\ell)$, $n = \frac{\ell-1}{2}$ and $P \in C_\lambda$.

- $$f_\ell(X) = \prod_{a=1}^{n} (X - ([a]P)_x)$$
- $\exists C \in \mathbb{F}_p[X]$ permutating the roots of f_ℓ tq :
  $$x \to C(x) \to C^{(2)}(x) \to \ldots \to C^{(n)}(x) = x$$
- from the definition of C, $\exists v$ such that :
  $$(\varphi(P))_x = ([\lambda]P)_x = C^{(v)}(x)$$
- $\exists\, \eta_0 \in \mathcal{A}_\lambda$ such that :
  $$\eta_0 \to C(\eta_0) \to \ldots \to C^{(q)}(\eta_0) = \eta_0$$

(M is the minimal polynomial of $\eta_0, \deg(M) = q$)

$$\begin{tikzcd}
\mathbb{F}_p[X]/(f_\ell) \arrow[dd, dash, "\frac{\ell-1}{2}"'] \arrow[dr, dash] & \\
& \mathbb{F}_p[X]/(M(X)) \arrow[dl, dash, "q"] \\
\mathbb{F}_p &
\end{tikzcd}$$

Extensions of $\mathbb{F}_p$

MMS

Morain-Mihăilescu-Schost algorithm (q odd)

- let c be a generator of $(\mathbb{Z}/\ell\mathbb{Z})^*$ and $x = \log_c \lambda$.
- let q odd such that : $q \mid \ell - 1$,
- denote $q' = \frac{\ell-1}{2q}$ so $H = < c^q >$, $K = < c^{q'} >$. Compute :

$$\eta_0 = \sum_{a \in H} g_a(x) = \sum_{j=0}^{q'-1} g_{h^j}(x) \text{ and } \eta_1 = \eta_0(g_k(x)) \quad (\text{for } x \in \mathcal{A}_\lambda)$$

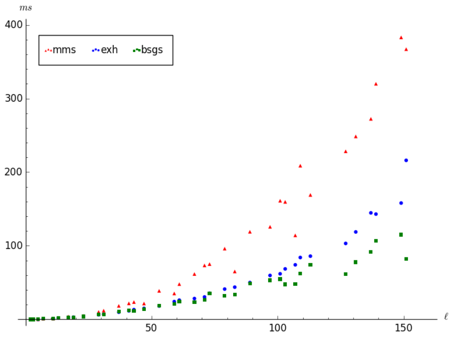
where $g_a \in \mathcal{A}_\lambda$ and $g_a(x) = ([a]P)_x$, $P = (x, y) \in C_\lambda$

- there exists $C \in \mathbb{F}_p[X]$ such that : $C(\eta_0) = \eta_1$,
- compute M, η_0 minimal polynomial, *whose degree is* q,
- computation of X^p modulo M and iterates of C (for composition) leads to x mod q
- using the CRT to conclude

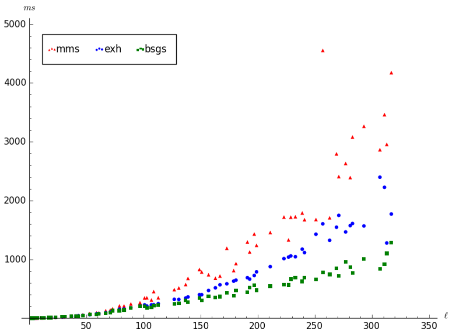
MMS

- finding v such that$(\varphi(P))_x = ([\lambda]P)_x = C^{(v)}(x)$ uses a baby step giant step algorithm
- computing are more expensive when q is even :
  requires constructions dealing with the ordinates, cost roughly doubles
- complexity is hard to evaluate : two different contributions, not always the same dominating

Comparison of the methods



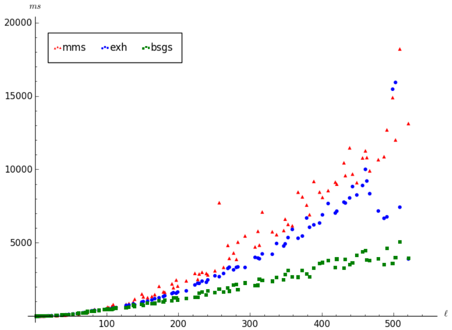
256-bits curves



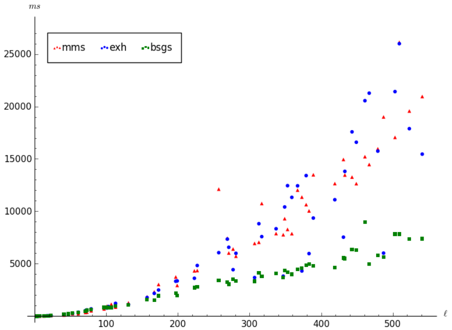
512-bits curves

Relation between the time (*ms*) to compute the eigenvalue and ℓ.
(100 curves measured)

Comparison of the methods



768-bits curves



a 1024-bits curve

Relation between the time (*ms*) to compute the eigenvalue and ℓ.

Comparison of the methods

Conclusion

- BSGS is a significative improvement compared to exhaustive search,
  - BSGS becomes rapidly quicker,
  - clear difference for 300-bits curves and for larger curves,
- MMS is said to be quicker than BSGS in the article but not for cryptographic sizes (at least with my implementation) :
  - benchmarks on the paper are made on a 8000-bits curve !
  - however optimisations of my implementation are possible.
- some ide as to improve my implementation of MMS :
  - suggested in the article : some computations made with an even q can be used for computations with odd q,
  - comp are the different factorizations of $\ell - 1$ and use the optimal decomposition : $\ell - 1 = q_1 q_2$.

Conclusion

Some ideas to improve SEA in PARI/GP

- any improvement of polynomial arithmetic will improve performance of the SEA algorithm
- BSGS can be applied in the isogeny cycles case : once $\lambda \mod \ell$ is found it is sometimes possible to find $\lambda \mod \ell^m$ for some integers m.
- for an Elkies prime, finding the factor f_ℓ of the division polynomial requires to compute an isogenous curve of degree ℓ. The algorithm used in PARI/GP is not in the state of the art : the most efficient algorithm has been published by Bostan-Morain-Salvy-Schost. Implement BMSS would improve the speed of SEA.

Thank you for your attention,

any questions ?