The discrete logarithm problem in some finite fields

For who ?

This is a project for a student interested in cryptography. It deals with number fields, finite fields, polynomials rings and (maybe most of all) resultants of polynomials with integer coefficients.

What's the problem ?

Cryptography relies on hard problems. Some protocols consider the discrete logarithm problem that is as follows :

Discrete logarithm problem. *Let G by a multiplicative group generated by g, of order n, and h be a arbitrary element in G. We solve the discrete logarithm problem (DLP) in G if we are able to find an integer $x \in \mathbb{Z}/n\mathbb{Z}$ such that :*

$$g^x = h.$$

Pairing protocols (that help for instance three or more participants to share a common secret key) relies on the security of both DLP in finite fields and DLP in elliptic curves. More precisely, a pairing is a map e :

$$e : E/\mathbb{F}_p \times E/\mathbb{F}_p \mapsto \mathbb{F}_{p^n}$$

A protocol using e is secure as soon as the DLP in the group of the rational points of the elliptic curve $E/\mathbb{F}_p$ is hard **and** the DLP in the finite field $\mathbb{F}_{p^n}$ is hard too. I spare you the details but we can show that the best choice (for a defender, someone that is using the protocol) is to choose a finite field such that the characteristic p can be written as :

$$p = L_{p^n}(1/3, c).$$

where $L_Q(a, c) = \exp(c + o(1)(\log Q)^a(\log\log Q)^{1-a})$.

The question is : in that precise configuration, which of the current algorithms that solve the DLP in a finite field would be the best ? Answering this question is essential to understand the security of this kind of protocols, and to give precise advices on the size of the finite field that should be taken in real life. For this reason, answering this question would constitute a solid basis for a publication to a conference on cryptography.

How to solve it ?

This question has not been answered yet since several algorithms seem to be competitive in this setting – and some of them are quite recent. Thus, a precise asymptotic complexity comparison needs to be done. This project will lead to consider :

- the Function Field Sieve (FFS)
- Frobenius Representation Algorithms (FRA)
- the Number Field Sieve (NFS) together with its variants.

All these algorithms have a complexity in $L_{p^n}(1/3, \cdot)$ so we need to compare the second constant. In particular, the complexity of NFS in this setting in hard to express. To give a simple and constant complexity in this setting, we would be pleased to find a new upper bound on the resultant of two polynomials. We recall that the resultant of two univariate polynomials over a field is commonly defined as the determinant of their Sylvester matrix. Some bound already exist but the are not appropriate to our study.