The Discrete Logarithm Problem in Finite Fields of Small Characteristic

Jens Zumbrägel[*]

Graduate Lectures in Mathematics
TU Dresden · May 2014

1 Introduction to the DLP

Let $(G, \cdot)$ be a cyclic group of order n, and let $\alpha \in G$ be a generator.

Definition 1.1. The *Discrete Logarithm Problem (DLP)* in the group G is the following computational problem:

- given the group $(G, \cdot)$, the generator $\alpha \in G$, and $\beta \in G$,
- find $x \in \mathbb{Z}_n$ such that $\alpha^x = \beta$.

Notation $\log_\alpha \beta := x$.

Remark 1.2. Consider the group isomorphism

$$\varphi : \mathbb{Z}_n \to G, \quad x \mapsto \alpha^x.$$

This map is efficiently computable (via the Square-and-Multiply method), but difficult in general to invert. Such a map is called a *one-way function*.

Example 1.3. Let $G = \mathbb{Z}_{13}^*$, the group of invertible elements modulo 13, and let $\alpha = 2$.

x	0	1	2	3	4	5	6	7	8	9	10	11
α^x	1	2	4	8	3	6	12	11	9	5	10	7

One sees that $\log_\alpha 5 = 9$. Check this using the Square-and-Multiply method: $2^9 = 2^8 \cdot 2 = ((2^2)^2)^2 \cdot 2 = (4^2)^2 \cdot 2 = 3^2 \cdot 2 = 5$.

Which groups are used for the DLP?

- $\mathbb{F}_{q^n}^*$, where $q = p^n$, the multiplicative group of a finite field,
- $E(\mathbb{F}_q)$, the $\mathbb{F}_q$-rational points on an elliptic curve,
- $J(\mathbb{F}_q)$, the Jacobian of a (hyperelliptic) curve.

[*] Institute of Algebra, TU Dresden, 01062 Dresden, Germany, E-mail: jens.zumbragel@ucd.ie. Supported by the Irish Research Council, grant number ELEVATEPD/2013/82.

Application

The difficulty of the DLP is used for secure Internet communication. Here are examples for three basic cryptographic protocols [7, 8].

1. **Key agreement**:

$$\begin{array}{ccccc} \text{Alice} & & \text{public} & & \text{Bob} \\ \hline & & (G,\cdot),\, \alpha & & \\ a \in \mathbb{Z}_n & \to & \alpha^a & & \\ & & \alpha^b & \leftarrow & b \in \mathbb{Z}_n \\ k_A = (\alpha^b)^a & & & & k_B = (\alpha^a)^b \end{array}$$

   Alice and Bob have shared a common secret key $k_A = k_B$.

2. **Encryption**: Alice → Bob

   Bob chooses $b \in \mathbb{Z}_n$ and releases his *public key* $\beta := \alpha^b$.
   Alice has secret message $m \in G$, chooses $a \in \mathbb{Z}_n$ and sends ciphertext

$$(\gamma, \delta) \ = \ (\alpha^a,\, m \oplus \beta^a).$$

   Bob decrypts by computing $\delta \ominus \gamma^b = \delta \ominus \alpha^{ab} = \delta \ominus \beta^a = m$.
   Here, $\oplus$ is some arbitrary group operation (say, XOR), and $\ominus$ its inverse.

3. **Signature**: Bob → Alice

   Bob chooses $b \in \mathbb{Z}_n$ and releases his *public key* $\beta := \alpha^b$.
   Suppose there is a computable "hash" map $G \to \mathbb{Z}_n$, $r \mapsto \bar{r}$.
   To sign a message message $m \in \mathbb{Z}_n$, Bob chooses $k \in \mathbb{Z}_n^*$ and sends

$$(r, s) \ = \ (\alpha^k,\, k^{-1}(m - b\bar{r})) \ \in \ G \times \mathbb{Z}_n.$$

   Alice verifies the signature by checking $\alpha^m = \beta^{\bar{r}} r^s$.
   This works, since $\beta^{\bar{r}} r^s = \alpha^{b\bar{r}} \alpha^{m - b\bar{r}} = \alpha^m$.

DLP algorithms

Definition 1.4. Notation for running time:

$$L_n(\alpha, c) \ := \ \exp\big((c + o(1))(\log n)^{\alpha}(\log\log n)^{1-\alpha}\big),$$

where $\alpha \in [0, 1]$ and $c > 0$.

The running time is usually given in terms of the input size, which for a DLP algorithm in a group of order n means $\log n$. We distinguish between:

- *polynomial time*: $O((\log n)^c) = \exp\big((c + o(1))(\log\log n)\big)$, i.e., $\alpha = 0$,
- *exponential time*: $O(n^c) = \exp\big((c + o(1))(\log n)\big)$, i.e., $\alpha = 1$,
- *subexponential time*: $0 < \alpha < 1$.

Example 1.5. Multiplication and exponentiation in $\mathbb{Z}_p$ or $\mathbb{F}_q$ are polynomial time algorithms. On the other hand, *generic* algorithms for the DLP, which do not exploit a particular group representation and work in any group, are exponential time. One example of a generic algorithm is the following.

Algorithm 1.6. Baby-Step-Giant-Step method to compute $\log_\alpha \beta$. Let $m := \lceil \sqrt{n} \rceil$.

1. compute α^i, for all $0 \leqslant i < m$ (baby steps)
2. compute $\beta\alpha^{-mj}$, for all $0 \leqslant j < m$ (giant steps)

If a "collision" $\beta\alpha^{-mj} = \alpha^i$ is found, we have $\beta = \alpha^{mj+i}$, and hence $\log_\alpha \beta = mj + i$. The running time is $O(\sqrt{n})$, i.e., $L_n(1, \frac{1}{2})$.

Index Calculus Method

The index calculus method (ICM) yields a subexponential time algorithm for certain particular groups. The method is first described abstractly for a general cyclic group G.

We choose a subset $S \subseteq G$ called *factor base*. Consider the homomorphism

$$\varphi : \mathbb{Z}_n^S \to G, \quad (e_s)_{s\in S} \mapsto \prod_{s\in S} s^{e_s}.$$

Assume that $\langle S \rangle = G$, i.e., that φ is surjective. This method has three phases.

1. **Relation Generation**: Find elements in $\ker \varphi$, called *relations*. We thereby construct a subset $\mathcal{R} \subseteq \ker \varphi$.
2. **Linear Algebra**: Find a nonzero element $(x_s)_{s\in S} \in \mathcal{R}^\perp$, i.e., $\sum_{s\in S} x_s e_s = 0$ for all $(e_s)_{s\in S} \in \mathcal{R}$.
3. **Descent Tree**: Given $\beta \in G$ find $(e_s)_{s\in S}$ such that $\beta = \prod_{s\in S} s^{e_s}$. Then $\log_\alpha \beta = \sum_{s\in S} e_s \log_\alpha s$ has been found.
   Start with β as root and descend down, i.e., rewrite elements as a product of "smaller" elements, until only factor base elements remain.

When do the relations collected in the first step determine the factor base logarithms in the second step?

Lemma 1.7. *If* $(x_s)_{s\in S} \in \mathcal{R}^\perp$ *and* $\operatorname{span} \mathcal{R} = \ker \varphi$ *then* $\exists \lambda \in \mathbb{Z}_n \; \forall s \in S : \; x_s = \lambda \log_\alpha s$.

Proof. It holds $\mathcal{R}^\perp = (\operatorname{span} \mathcal{R})^\perp = (\ker \varphi)^\perp \cong \mathbb{Z}_n^S / \ker \varphi \cong \operatorname{Im} \varphi \cong \mathbb{Z}_n$. On the other hand, we have $(\log_\alpha s)_{s\in S} \in \mathcal{R}^\perp$. Indeed, since $\log_\alpha : G \to \mathbb{Z}_n$ is a group homomorphism, $\sum_{s\in S} e_s \log_\alpha s = \log_\alpha(\prod_{s\in S} s^{e_s}) = 0$ for all $(e_s)_{s\in S} \in \mathcal{R} \subseteq \ker \varphi$. □

How to devise a concrete algorithm out of this framework? The following algorithm works for $\mathbb{Z}_p = \mathbb{Z}/(p)$ and $\mathbb{F}_{p^n} = \mathbb{F}_p[x]/(f(x))$, where f is an irreducible polynomial of degree n. Both fields are represented as a residue class ring R/I of a unique factorisation domain R. The index calculus method crucially depends on the ability to obtain factorisations of random elements into "small" factors in R, using trial division.

Algorithm 1.8. Basic Index Calculus Method to compute $\log_\alpha \beta$.
Choose a number t and let the factor base $S = \{p_1, \dots, p_t\}$ consist of the t smallest primes, or monic irreducible polynomials, respectively. For simplicity, we assume $\alpha \in S$.

1. For random k try to write $\alpha^k = \prod p_i^{e_i}$, using trial division. If successful, a relation is found.
2. If more than t relations are found, solve the corresponding linear system to obtain $\log_\alpha p_i$ for all $p_i \in S$.
3. For random k try to write $\beta\alpha^k = \prod p_i^{f_i}$ and conclude $\log_\alpha \beta = -k + \sum f_i \log_\alpha p_i$.

The running time has been analysed by Adleman [1]. By using an optimal choice for the factor base size t one obtains the subexponential running time $L(\frac{1}{2}, 1)$.

Example 1.9. Let $G = \mathbb{Z}_{101}^*$, so that $n = 100$, and let $\alpha = 2$. As factor base we choose $S = \{2, 3, 5, 7\}$.

1. Suppose we take $k = 7$, $k = 9$, and $k = 33$. We find the relations $2^7 \equiv 3^3$, $2^9 \equiv 7$, and $2^{33} \equiv 5 \cdot 7$.
2. Writing $\log$ for $\log_2$, we obtain the linear equations $7 = 3\log 3$, $9 = \log 7$, and $33 = \log 5 + \log 7$. Computing in $\mathbb{Z}_{100}$ we conclude $\log 5 = 24$ and $\log 3 = 7/3 = 69$.
3. Suppose the target is $\beta = 42$, and we take $k = 14$. Then $\beta \cdot 2^{14} \equiv 3 \cdot 5$, hence $\log \beta = 69 + 24 - 14 = 79$.

2 Factoring Problem and Further Developments of the ICM

Algorithms for the DLP have often progressed in parallel with factoring algorithms.

Definition 2.1. The *Integer Factorisation Problem (IFP)* is the following problem:

- given a composite integer $n = pq$, where p, q are different primes, $\log p \approx \log q$,
- find the prime factors p, q of n.

Most cryptographic protocols used in practice are based either on the DLP or on the IFP. The most famous one is the RSA cryptosystem [19].

- **RSA encryption**: Alice → Bob

  Choose a composite n as above, let $\varphi(n) := |\mathbb{Z}_n^*| = (p-1)(q-1)$, choose $e \in \mathbb{Z}_{\varphi(n)}^*$ and compute $d := e^{-1} \in \mathbb{Z}_{\varphi(n)}^*$. Bob's public key is (n, e), his private key is d.
  Alice encrypts a message $m \in \mathbb{Z}_n^*$ by sending $c = m^e$.
  Bob decrypts the ciphertext by $c^d = m^{ed} = m$, since $ed \equiv 1$ modulo $|\mathbb{Z}_n^*| = \varphi(n)$.
  (The procedure also works for messages $m \in \mathbb{Z}_n$.)

- **Signature**: Bob → Alice

  With the same setup as before, Bob signs a message $m \in \mathbb{Z}_n^*$ by computing $s = m^d$.
  Alice verifies the signature by computing $s^e = m$.

Factoring algorithms

Usually, factoring algorithms are based on the *congruent squares* attack:

$$\text{given } n, \quad \text{generate numbers } x, y \text{ such that } x^2 \equiv y^2 \mod n.$$

Then n divides $(x-y)(x+y)$, and, assuming that x, y are random and independent, with probability $\frac{1}{2}$ we have $x \neq \pm y$, in which case $\gcd(n, x-y) \in \{p, q\}$.

A number m is called *B-smooth* if all its prime factors are $\leqslant B$.

Algorithm 2.2. Index Calculus Method to find a congruent square $x^2 \equiv y^2 \mod n$.
Let $S = \{p \text{ prime} \mid p \leqslant B\}$ be the factor base.

1. Find elements x_i such that $z_i := x_i^2 \mod n$ is B-smooth; let $z_i = \prod p^{e_{p,i}}$ and $i \in I$.
2. Solve a linear system over $\mathbb{F}_2$ to find a subset $J \subseteq I$ such that $\sum_{i \in J} e_{p,i}$ is even for all $p \in S$, so that $\prod_{i \in J} z_i$ is a square.

Let $x := \prod_{i \in J} x_i$ and $z := \prod_{i \in J} z_i$, then $x^2 \equiv z \mod n$ and z is a square.

If the x_i are chosen randomly in $\{1, \ldots, n\}$, then the z_i are expected to be in the order of n, so less likely to be smooth.

The idea of the **Quadratic Sieve** [18] is instead to use $x_i \approx \sqrt{n} + i$ for small i, so that $z_i \approx 2\sqrt{n}i + i^2 \approx \sqrt{n}$.

For the complexity analysis of index calculus methods the following result is crucial.

Theorem 2.3 (Canfield, Erdös, Pomerance [6]). *Let P be the probability that a number in $\{1, \ldots, M\}$ is B-smooth. Then*

$$P = u^{-(1+o(1))u} \quad \textit{where } u = \frac{\log M}{\log B}.$$

In the following we use the notation $L_n(\alpha)$ as a short hand for $L_n(\alpha, c)$ for some $c > 0$.

Corollary 2.4. *If $B = L_M(\frac{1}{2})$ the expected #trials to obtain a B-smooth number is $L_M(\frac{1}{2})$.*

Proof. Using the formula for L_M we compute, ignoring $o(1)$ terms,

$$u = \frac{\log M}{\log B} = \frac{1}{c} \frac{(\log M)^{1/2}}{(\log\log M)^{1/2}}$$

and hence

$$\begin{aligned} u^u &= \exp(u \log u) \\ &= \exp\big(\tfrac{1}{c} \frac{(\log M)^{1/2}}{(\log\log M)^{1/2}} \cdot (\log \tfrac{1}{c} + \tfrac{1}{2} \log\log M - \tfrac{1}{2} \log\log\log M)\big) \\ &= \exp\big((\tfrac{1}{2c} + o(1))(\log M)^{1/2}(\log\log M)^{1/2}\big) = L_M(\tfrac{1}{2}, \tfrac{1}{2c}) \end{aligned}$$ □

For the Quadratic Sieve we apply these results with $M \approx \sqrt{n}$ and we assume that the z_i are uniformly distributed in $\{1, \ldots, M\}$. This yields a heuristic running time of $L(\frac{1}{2}, 1)$.

Number Field Sieve

The **Number Field Sieve** (NFS) [16] is a method to obtain the much lower order $M = L_n(\frac{2}{3})$. It is the fastest integer factorisation algorithm currently known.

The idea is to write n in m-adic representation

$$n = m^d + c_{d-1}m^{d-1} + \cdots + c_1 m + c_0,$$

with $m = \lfloor n^{1/d} \rfloor$. Then $n = f(m)$, where $f(t) := t^d + c_{d-1}t^{d-1} + \cdots + c_1 t + c_0 \in \mathbb{Z}[t]$, and we assume that $f(t)$ is irreducible.

Let $\alpha \in \mathbb{C}$ be a root of f and consider the homomorphism

$$\varphi : \mathbb{Z}[\alpha] \to \mathbb{Z}_n, \quad g(\alpha) \mapsto g(m),$$

which is well-defined, since $n \mid f(m)$. In diagram form:

$$\begin{tikzcd} & \mathbb{Z}[t] \arrow[dl, "\mathrm{ev}_\alpha"'] \arrow[dr, "\overline{\mathrm{ev}}_m"] & \\ \mathbb{Z}[\alpha] \arrow[rr, "\varphi"'] & & \mathbb{Z}_n \end{tikzcd}$$

Now we look for $g_i(t) \in \mathbb{Z}[t]$ such that $\prod g_i(\alpha) = \beta^2$ is a square in $\mathbb{Z}[\alpha]$ and $\prod g_i(m) = y^2$ is a square in $\mathbb{Z}$. Then, letting $x := \varphi(\beta)$, we have found a congruent square

$$x^2 = \varphi(\beta^2) = \varphi\big(\prod g_i(\alpha)\big) = \prod g_i(m) = y^2.$$

A problem we have to deal with is factorisation in $\mathbb{Z}[\alpha]$. One uses the norm map $N : \mathbb{Q}(\alpha) \to \mathbb{Q}$, which satisfies $N(\mathbb{Z}[\alpha]) \subseteq \mathbb{Z}$, in order to obtain "smooth" elements $g_i(\alpha)$.

In particular, we choose $g_i(t) := a_i t + b_i$ with $|a_i|, |b_i| \leqslant C$. We let

$$C = L_n(\tfrac{1}{3}) \quad \text{and} \quad d \approx \Big(\frac{\log n}{\log\log n}\Big)^{1/3},$$

so that $m \approx n^{1/d} = L_n(\frac{1}{3})$. Then we have $N(g_i(\alpha)) \approx C^d n^{1/d} = L_n(\frac{2}{3})$, and this gives rise to an $L_n(\frac{1}{3})$ factoring algorithm (actually, $L_n(\frac{1}{3}, 1.923)$).

DLP algorithms

The idea of the Number Field Sieve has been applied to obtain $L(\frac{1}{3})$ algorithms for the DLP in various finite fields.

- We can use the Number Field Sieve for the DLP in a prime field $\mathbb{F}_p$ [12].

  The idea is to write $p = m^d + c_{d-1}m^{d-1} + \cdots + c_1 m + c_0 = f(m)$. Again, let $\alpha \in \mathbb{C}$ be a root of f and consider the homomorphism $\varphi : \mathbb{Z}[\alpha] \to \mathbb{Z}_p$. We look for $g_i(t) := a_i t + b_i$ such that both $a_i\alpha + b_i$ and $a_i m + b_i$ are "smooth".

- **Function Field Sieve** (FFS) [2, 3] for the DLP in $\mathbb{F}_{p^n}$, where p is small.

  The FFS is the analog of the NFS, with the ring $\mathbb{Z}$ replaced by $\mathbb{F}_p[x]$.
  Let $\mathbb{F}_{p^n} = \mathbb{F}_p[x]/(P(x))$, and write $P(x) = m(x)^d + c_{d-1}(x)m(x)^{d-1} + \cdots + c_1(x)m(x) + c_0 = f(m(x))$, where $f(t) = f(x,t) = \sum c_i(x)t^i$ is a bivariate polynomial, which defines an algebraic curve C and its function field $\mathbb{F}_p(C) = \operatorname{Quot}\big(\mathbb{F}_p[x,t]/(f(x,t))\big)$ (if f is irreducible).

3 Recent developments of the Function Field Sieve

There have been dramatic developments of the FFS, which led cryptologists to say that the DLP in small characteristic is *dead*. These advancements are based on an improved and simplified version of the FFS. We start with a recap, comparing the relation generation for the basic and the advanced versions of the index calculus method.

- **Basic version** of the ICM:
  Let $G = \mathbb{Z}_p^* = \langle\alpha\rangle$, $n = p-1$, let $S := \{p \text{ prime}, p \leqslant B\}$, assume $\alpha \in S$.
  Choose $k \in \{1, \ldots, n\}$ uniformly at random and look for $\alpha^k \bmod p$ to be B-smooth.
  - l.h.s. = α^k : trivially B-smooth by construction,
  - r.h.s. = $\alpha^k \bmod p$: uniformly distributed in $\{1, \ldots, p-1\}$, is B-smooth with probability $\approx u^{-u}$, where $u = \log p / \log B$, by Theorem 2.3.

  Similarly, for groups $G = \mathbb{F}_{p^n}^*$ of finite fields of small characteristic.

- **Advanced versions** (NFS, FFS) of the ICM:
  Use a "construction" of elements of l.h.s. and r.h.s., so that both sides have to be smooth, but the "size" on either side is much smaller and we have a higher smoothness probability.

  For the NFS in $\mathbb{Z}_p$: Choose $f \in \mathbb{Z}[t]$ monic irreducible of degree d such that $p = f(m)$ for some $m \approx p^{1/d}$, let $\alpha \in \mathbb{C}$ be a root of f, and consider the diagram:

$$\begin{tikzcd} & \mathbb{Z}[t] \arrow[dl, "\mathrm{ev}_\alpha"'] \arrow[dr, "\mathrm{ev}_m"] & \\ \mathbb{Z}[\alpha] \arrow[dr, "\varphi"'] & \circlearrowleft & \mathbb{Z} \arrow[dl, "\pi"] \\ & \mathbb{Z}_p & \end{tikzcd} \qquad \begin{tikzcd} & at+b \arrow[dl, mapsto] \arrow[dr, mapsto] & \\ a\alpha+b & & am+b \end{tikzcd}$$

  Here, $\varphi(g(\alpha)) = g(m) \bmod p$ for $g \in \mathbb{Z}[t]$, which is well-defined since $p \mid f(m)$.
  For relation generation, choose $a, b \in \mathbb{Z}$ small and consider the images of $at + b$. If $a\alpha + b = \prod \gamma_i$ and $am + b = \prod c_i$, where the $\varphi(\gamma_i)$ and $c_i \bmod p$ are factor base elements, then a relation $\prod \varphi(\gamma_i) = \varphi(a\alpha + b) \equiv am + b = \prod c_i \bmod p$ is found.

  For the FFS in $\mathbb{F}_{p^n}[x] = \mathbb{F}_p[x]/(P(x))$: Replace $\mathbb{Z}$ by the polynomial ring $\mathbb{F}_p[x]$, and $\mathbb{Z}[\alpha] \cong \mathbb{Z}[t]/(f(x))$ by $\mathbb{F}_p[x,t]/I$, where $I = (f(x,t))$, so that:

$$\begin{tikzcd} & \mathbb{F}_p[x,t] \arrow[dl] \arrow[dr, "\mathrm{ev}_{m(x)}"] & \\ \mathbb{F}_p[x,t]/I \arrow[dr] & \circlearrowleft & \mathbb{F}_p[x] \arrow[dl] \\ & \mathbb{F}_{p^n} & \end{tikzcd}$$

Then $\mathrm{Quot}(\mathbb{F}_p[x,t]/I) = \mathbb{F}_p(C)$ is the *function field* of the curve C defined by f, and $\mathrm{Quot}(\mathbb{F}_p[x]) = \mathbb{F}_p(x)$ is the rational function field. A full description and implementation of the original Function Field Sieve is rather involved and technical.

The Joux-Lercier FFS

This is a simplified, more efficient and wider applicable version [15] of the Function Field Sieve. Let $g_1(t)$ and $g_2(x)$ be polynomials and consider the following "symmetrised" version of the previous diagram:

$$\begin{tikzcd}
& \mathbb{F}_p[x,t] \arrow[dl, "x\mapsto g_1(t)"'] \arrow[dr, "t\mapsto g_2(x)"] & \\
\mathbb{F}_p[t] \arrow[dr, "\varphi"'] & \circlearrowleft & \mathbb{F}_p[x] \arrow[dl, "\psi"] \\
& \mathbb{F}_{p^n} &
\end{tikzcd}$$

For commutativity, the homomorphisms φ and ψ have to satisfy $\varphi(t) = \psi(g_2(x)) = g_2(\psi(x))$ and $\psi(x) = \varphi(g_1(t)) = g_1(\varphi(t))$. Letting $\tau := \varphi(t)$ and $\xi := \psi(x)$ this means

$$\tau = g_2(\xi) \quad \text{and} \quad \xi = g_1(\tau).$$

Now, if we define $\mathbb{F}_{p^n}$ as $\mathbb{F}_p[x]/(P(x))$, where $P(x) \mid g_1(g_2(x)) - x$, this works with $\xi := [x]$ and $\tau := g_2(\xi)$, since $g_1(\tau) = g_1(g_2(\xi)) = \xi$.

The diagram for relation generation is

$$\begin{tikzcd}
& xt + at + bx + c \arrow[dl, mapsto] \arrow[dr, mapsto] & \\
g_1(t)t + at + bg_1(t) + c & & xg_2(x) + ag_2(x) + bx + c\,,
\end{tikzcd}$$

which yields the equation $g_1(\tau)\tau + a\tau + bg_1(\tau) + c = \xi g_2(\xi) + ag_2(\xi) + b\xi + c$ in $\mathbb{F}_{p^n}$, for $a, b, c \in \mathbb{F}_p$. If the polynomials on both sides are smooth a relation has been found.

The running time analysis of FFS algorithms is based on the following polynomial analog of Theorem 2.3.

Theorem 3.1 (Odlyzko, Lovorn [17]). *The probability for a polynomial in $\mathbb{F}_p[x]$ of degree m to be b-smooth (all prime factors have degree $\leqslant b$) is*

$$u^{-(1+o(1))u} \quad \textit{where } u = \frac{m}{b}.$$

Now, since $P(x) \mid g_1(g_2(x)) - x$ we must have $n = \deg P(x) \leqslant \deg g_1 \deg g_2$, and thus a natural requirement for g_1, g_2 is to choose $\deg g_1 \approx \deg g_2 \approx \sqrt{n}$. Observe that the degrees of the polynomials on both sides is then about $\sqrt{n}$, which is much smaller than n. The resulting running time is $L(\frac{1}{3}, 1.526)$.

We remark that the base field $\mathbb{F}_p$ can be replaced by any finite field $\mathbb{F}_q = \mathbb{F}_{p^m}$, which gives an algorithm for the DLP in $\mathbb{F}_{q^n} = \mathbb{F}_{p^{mn}}$. In 2012 the Joux-Lercier algorithm was applied for a record DLP computation in the finite field $\mathbb{F}_{3^{6\cdot 97}}$ of size 923 bits [13].

An extremely efficient variation

There are choices of g_1, g_2 in the Joux-Lercier algorithm that turn out to be much more efficient than the ones of balanced degree. This is independent recent work by Göloğlu, Granger, McGuire, Zumbrägel [9], and by Joux [14]. The new algorithms attack DLPs in finite fields of the form $\mathbb{F}_{q^{kn}}$, where $q = p^\ell$, so that the extension degree ℓkn is composite. Since any finite field $\mathbb{F}_{p^m}$ of small characteristic can be embedded into such a field, these methods do actually affect the DLP in *all* finite fields of small characteristic.

Higher splitting probabilities. Consider a field of the form $\mathbb{F}_{q^{kn}}$, where $k \geqslant 2$, and let

$$g_2(x) := x^q,$$

so that the r.h.s. polynomial is of the form

$$f_{a,b,c}(x) := x^{q+1} + ax^q + bx + c\,.$$

The key observation is that polynomials of this form enjoy a much higher probability of splitting completely into linear factors. Indeed, letting

$$Z := \{(a,b,c) \in \mathbb{F}_{q^k}^3 \mid f_{a,b,c}(x) \text{ splits completely}\},$$

then one can show that $|Z| \approx q^{3k-3}$ and the set Z can be effectively parametrised.

Idea of proof. Start with the equation $X^q - X = \prod_{\nu \in \mathbb{F}_q}(X - \nu)$ and deduce the identity

$$X^q Y - XY^q = Y \prod_{\nu \in \mathbb{F}_q} (X - \nu Y).$$

If we substitute X by $\alpha x + \beta$ and Y by $\gamma x + \delta$, where $\alpha, \beta, \gamma, \delta \in \mathbb{F}_{q^k}$, we obtain

$$\begin{aligned} X^q Y - XY^q &= (\alpha x + \beta)^q (\gamma x + \delta) - (\alpha x + \beta)(\gamma x + \delta)^q \\ &= (\alpha^q x^q + \beta^q)(\gamma x + \delta) - (\alpha x + \beta)(\gamma^q x^q + \delta^q) \\ &= (\alpha^q \gamma - \alpha \gamma^q) x^{q+1} + (\alpha^q \delta - \beta \gamma^q) x^q + (\beta^q \gamma - \alpha \delta^q) x + (\beta^q \delta - \beta \delta^q) \end{aligned}$$

and hence (up to a scalar factor) a polynomial of the form $f_{a,b,c}$. On the other hand, since

$$Y \prod_{\nu \in \mathbb{F}_q} (X - \nu Y) = (\gamma x + \delta) \prod_{\nu \in \mathbb{F}_q} \big((\alpha - \nu\gamma)x + (\beta - \nu\delta)\big)$$

this polynomial splits completely into linear factors, and for $k \geqslant 2$ this method produces many distinct splitting polynomials. $\square$

Now we may choose $\deg g_1$ very small, so that the l.h.s. polynomial $g_1(t)t + at + bg_1(t) + c$ splits also with high probability. With the appropriate parameter choices this yields a *polynomial time* $L(0)$ algorithm for Phases 1 and 2 of the Index Calculus Method. This means that the first parts of the ICM have become very easy, and the focus lies now on Phase 3, i.e., building up the descent tree, which has previously been the easiest step.

Further progress and final remarks

- Barbulescu, Gaudry, Joux, Thomé have found a way [5] to make also Phase 3 efficient by a new *Descent Method*:

  There is a polynomial time algorithm to express elements $Q(x) \in \mathbb{F}_{q^k}[x]$ of degree $2d$, modulo $P(x)$, as product of degree d elements – for all d. The idea is to apply for each $Q(x)$ an Index Calculus Method (now replace X by $\alpha Q(x) + \beta$ and Y by $\gamma Q(x) + \delta$), which runs as before in polynomial time.

  This results in an overall *quasi-polynomial time* $L(o(1))$ algorithm for the DLP in small characteristic finite fields, which is a breakthrough result and a big leap from the previous $L(\frac{1}{3})$ algorithms. However, the new algorithm is not (yet) practical.

- The new methods have been applied to obtain much larger DLP records, the latest one in the finite field $\mathbb{F}_{2^{18 \cdot 513}}$ of size 9234 bits [11]. Some facts:
  - The finite field was defined as

    $$\mathbb{F}_{2^{9234}} = \mathbb{F}_{2^{18 \cdot 513}} = \mathbb{F}_{2^{18}}[x]/(x^{513} - c),$$

    for $c \in \mathbb{F}_{2^{18}}^*$ a primitive element. Thus $\mathbb{F}_{2^{9234}}$ is a *Kummer extension*, which exists since $513 \mid 2^{18} - 1$.
  - The factor base was $S := \{p \in \mathbb{F}_{2^{18}}[x] \text{ irreducible}, \deg p \leqslant 2\}$, of size $\approx 2^{35}$.
  - Using a *factor base preserving* automorphism group of size 1026, the number of variables was reduced, and 256 systems in 2^{17} variables each to be solved.
  - The descent was done in several stages, but without using the quasi-polynomial descent method of [5]. Instead, we used a *Gröbner basis method* (solving bilinear quadratic systems) due to Joux [14], which is a very efficient method in practice for descending small degree elements.
  - The running time was about $250\,000$ core hours for the linear algebra phase and about $150\,000$ core hours for building up the descent tree.

- The DLP in finite fields of small characteristic (usually 2 or 3) is of great importance in the context of *pairing-based cryptography*. Hence the natural question arises whether the new algorithmic progress weakens or indeed breaks any of the parameters proposed in the literature. The difficulty of the following DLPs in $\mathbb{F}_{q^k}$, arising from genus 1 or 2 supersingular curves over $\mathbb{F}_q$, was previously believed to be at the industry-standard 128-bit security level:

char.	genus 1		genus 2	
$p = 2$	$k = 4$	$q^k = 2^{4 \cdot 1223}$	$k = 12$	$q^k = 2^{12 \cdot 367}$
$p = 3$	$k = 6$	$q^k = 3^{6 \cdot 509}$		

  - Adj, Menezes, Oliveria, Rodríguez-Henríquez [4] showed that the DLP in $\mathbb{F}_{q^k}$
    - for $q^k = 3^{6 \cdot 509}$ is in 2^{74} operations computable.
  - Granger, Kleinjung, Zumbrägel [10] showed that the DLP in $\mathbb{F}_{q^k}$
    - for $q^k = 2^{4 \cdot 1223}$ is in 2^{59} operations computable,
    - for $q^k = 2^{12 \cdot 369}$ is in 2^{48} operations practically broken.

  Thus small characteristic pairings should now be considered completely insecure.

References

[1] L. M. Adleman, "A subexponential algorithm for the discrete logarithm problem with applications to cryptography," Proc. Foundations of Computer Science, pp. 55–60, IEEE (1979)

[2] L. M. Adleman, "The function field sieve," in: Algorithmic Number Theory—ANTS-I 1994, LNCS 877, pp. 108–121, Springer (1994)

[3] L. M. Adleman, M.-D. A. Huang, "Function Field Sieve Method for Discrete Logarithms over Finite Fields," *Information and Computation* **151**, pp. 5–16 (1999)

[4] G. Adj, A. Menezes, T. Oliveira and F. Rodríguez-Henríquez, "Weakness of $\mathbb{F}_{3^{6\cdot 509}}$ for discrete logarithm cryptography," in: Pairing-based Cryptography—Pairing 2013, LNCS 8365, pp. 20–44, Springer (2013), http://eprint.iacr.org/2013/446

[5] R. Barbulescu, P. Gaudry, A. Joux, E. Thomé, "A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic," in: Advances in Cryptology—EUROCRYPT 2014, LNCS 8441, pp. 1–16, Springer (2014), http://eprint.iacr.org/2013/400

[6] E. R. Canfield, P. Erdš, C. Pomerance, "On a problem of Oppenheim concerning 'factorisatio numerorum'," *J. Number Theory* **17**, no. 1, pp. 1–28 (1983)

[7] D. Hellman, M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inform. Theory* **22**, no. 6, pp. 644–654 (1976)

[8] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in: Advances in Cryptology—CRYPTO '84, LNCS 196, pp. 10–18, Springer (1985)

[9] F. Göloğglu, R. Granger, G. McGuire, J. Zumbrägel, "On the Function Field Sieve and the Impact of Higher Splitting Probabilities," in: Advances in Cryptology—CRYPTO 2013, LNCS 8043, pp. 109–128, Springer (2013), http://eprint.iacr.org/2013/074

[10] R. Granger, T. Kleinjung, J. Zumbrägel, "Breaking '128-bit Secure' Binary Supersingular Curves," to appear in: Advances in Cryptology—CRYPTO 2014, LNCS, Springer (2014), http://eprint.iacr.org/2014/119

[11] R. Granger, T. Kleinjung, J. Zumbrägel, "Discrete Logarithms in GF(2^9234)," NMBRTHRY Mailing List, 31 Jan 2014

[12] D. M. Gordon, "Discrete Logarithms in GF(p) using the Number Field Sieve," *SIAM J. Disc. Math.* **6**, no. 1, pp. 124–138 (1993)

[13] T. Hayashi, T. Shimoyama, N. Shinohara, T. Takagi, "Breaking Pairing-Based Cryptosystems Using η_T Pairing over $GF(3^{97})$," in: Advances in Cryptology–ASIACRYPT 2012, LNCS 7658, pp. 43–60, Springer (2012)

[14] A. Joux, "A new index calculus algorithm with complexity L(1/4+o(1)) in small characteristic," in: Selected Areas in Cryptography—SAC 2013, LNCS 8282, pp. 355–379, Springer (2014), http://eprint.iacr.org/2013/095

[15] A. Joux, R. Lercier, "The Function Field Sieve in the Medium Prime Case," in: Advances in Cryptology—EUROCRYPT 2006, LNCS 4004, pp. 254–270, Springer (2006)

[16] A. K. Lenstra, H. W. Lenstra, Jr. (eds), *The development of the number field sieve*, Lecture Notes in Mathematics 1554, Springer (1993)

[17] A. M. Odlyzko, "Discrete logarithms and smooth polynomials," in: *Finite Fields: Theory, Applications, and Algorithms*, Contemporary Mathematics 168, pp. 269–278, AMS (1994)

[18] C. Pomerance, "The quadratic sieve factoring algorithm," in: Advances in Cryptology–EUROCRYPT '84, LNCS 209, pp. 169–182, Springer (1985)

[19] R. L. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM* **21**, no. 2, pp. 120–126 (1978)