MPRI – Cours 2.12.2

F. Morain

$L[1/2]$ methods for DLP

2019/10/15

The slides are available on http://www.lix.polytechnique.fr/Labo/Francois.Morain/MPRI/2019

I. Index-calculus: a general framework

II. Coppersmith/Odlyzko/Schroeppel

I. Index-calculus: a general framework

(Western and Miller; Pollard, Adleman, Merkle, etc.)

Input: G in which primes exist ($\mathbb{F}_q^*$, hyperelliptic curves of large genus, etc.), $G = \langle g \rangle$, $n = \#G$, $h \in G$.

Output: x s.t. $h = g^x$.

Step 1: find the logarithms of the primes in $\mathcal{B} = \{p_1, p_2, \ldots, p_k\}$.

Step 2: look for b s.t. hg^b factors over $\mathcal{B}$:

$$hg^b = \prod_{j=1}^{k} p_j^{\alpha_j} \Leftrightarrow x + b \equiv \sum_{j=1}^{k} \alpha_j \log_g p_j \bmod n.$$

How do we find the $\log_g p_j$?

Choose random integers b_i for which

$$g^{b_i} = \prod_{j=1}^{k} p_j^{\alpha_{i,j}} \Leftrightarrow b_i \equiv \sum_{i=1}^{k} \alpha_{i,j} \log_g p_j \bmod n.$$

When enough relations have been gathered, solve the system.

Cost: $O(L_{\#G}[1/2, c])$ where c depends on G and/or $\#G$.

$$L_N[\alpha, c] = \exp((c + o(1))(\log N)^{\alpha}(\log\log N)^{1-\alpha}).$$

Adaptive and non-adaptive versions?

More adaptive:

$$g^{b_0} \prod_{i=1}^{r} h_i^{b_i} = \prod_{j=1}^{k} p_j^{\alpha_{i,j}}$$

and use linear algebra on the logs.
Depends on the value of r, etc.

Non-adaptive: perform Step 1 only and bet on $O((\log q)^c)$ queries?

Two strategies:

- non-adaptive versions;
- adaptive: precomputation phase only + improved Step 2.

The case of $\mathbb{F}_p$: Adleman's algorithm

This is the case $G = \mathbb{F}_p$ in the preceding:

Step 1: find the logarithms of the primes in $\mathcal{B} = \{p_1, p_2, \ldots, p_k\}$.

Step 2: look for b s.t. hg^b factors over $\mathcal{B}$:

$$hg^b = \prod_{j=1}^{k} p_j^{\alpha_j} \bmod p \Leftrightarrow x + b \equiv \sum_{j=1}^{k} \alpha_j \log_g p_j \bmod (p-1).$$

At the heart of the analyses of both steps is the probability

$$\frac{\psi(p, p_k)}{p}$$

Analysis

Prop. STEP1 costs $O(L(p)^{2+o(1)})$; STEP2 costs $O(L(p)^{3/2+o(1)})$.
Proof.
Proba(g^{b_i} is p_k-smooth)$= \dfrac{\psi(p,p_k)}{p}$ $\Rightarrow$ we need $k\frac{p}{\psi(p,p_k)}$ relations.
Trial division $\Rightarrow$ testing p_k-smoothness costs k divisions.
Linear algebra costs $O(k^r)$ with $2 \leq r \leq 3$ (see later).
Total cost:

$$O\left(k \cdot k\frac{p}{\psi(p,p_k)}\right) + O(k^r).$$

$k = L(p)^b \Rightarrow p_k \approx k \log k = O(L(p)^{b+o(1)})$

$$\Rightarrow O(L^{2b}L^{1/(2b)}) + O(L^{rb}) = O(L^{\max(2b+1/(2b), rb)}).$$

$2b + 1/(2b)$ minimal for $b = 1/2$ and has value $2 \geq rb$.

STEP2: $O(k\frac{p}{\psi(p,p_k)}) = O(L^{b+1/(2b)}) = O(L^{3/2})$. $\square$

II. Coppersmith/Odlyzko/Schroeppel (COS)

$$H = \lfloor\sqrt{p}\rfloor + 1, \quad J = H^2 - p$$

$$\mathcal{B} = \{q | q \text{ prime } < q_{\max}\} \cup \{H + c, 0 < c < c_{\max}\} \quad \ni g$$

Let $H + c_1$, $H + c_2$ in $\mathcal{B}$:

$$(H + c_1)(H + c_2) \equiv F(c_1, c_2) = J + (c_1 + c_2)H + c_1c_2 \bmod p$$

or

$$\log_g(H + c_1) + \log_g(H + c_2) \equiv \sum_i f_i \log q_i \bmod (p-1).$$

Goal: find enough equations to get $\log$ of elements in $\mathcal{B}$.

COS: algorithm

If we fix c_1, then

$$F(c_1, c_2) \equiv 0 \Longleftrightarrow c_2 \equiv -(J + c_1H)(H + c_1)^{-1} \bmod q.$$

$\Rightarrow$ sieve!

For given $0 < c_1 < c_{\max}$:

1. Set $T[c_2] = []$ for $0 < c_2 < c_{\max}$;
2. **for all** $q \in \mathcal{B}$ and $q^e \leq X$ **do**
   2.2 $c_2 = -(J + c_1H)(H + c_1)^{-1} \bmod q^e$;
   2.3 **while** $c_2 \leq c_{\max}$ **do**
      2.3.1 append$(T[c_2], q)$; {*trick!*}
      2.3.2 $c_2 := c_2 + q^e$;

Postsieve: find all c_2 s.t. $c_2 = \prod_{q \in T[c_2]} q$ and store $(c_1, c_2, T[c_2])$.

COS: analysis

Step 1:
With $k = L(p)^\beta$, $q_{\max} = L^\beta$, $c_{\max} = L^{\beta+\varepsilon}$, $|F(c_1, c_2)| = O(p^{1/2})$.

$$O\left(k \cdot k^0 \cdot \frac{p^{1/2}}{\psi(p^{1/2}, p_k)}\right) + O(k^r).$$

$$O(L^\beta L^{1/(4\beta)}) + O(L^{r\beta}) = O(L^{\max(\beta+1/(4\beta), r\beta)}).$$

Minimum for $\beta = 1/2$ (giving L^1); linear algebra dominates anyway, so cost in $O(L^{r/2})$, better than L^2.

Step 2: if we use the plain version, we still have $L^{3/2}$, which we can improve.

COS: smoothing

Step 1: express $g^w h$ as a product of small and medium primes $u < L[2]$ in time $L[1/2]$ (using ECM)

$$g^w h \equiv \left(\prod_i q_i^{e_i}\right) \times \left(\prod_i u_i^{f_i}\right);$$

Step 2: find $\log u$ for $u \in \{u_i\}$:

- find a $L[1/2]$-smooth $y = \prod_i q_i^{m_i}$ in an interval of length $L[1/2]$ around $\sqrt{p}/u$;
- find v in an interval of length $L[1/2]$ around $\sqrt{p}$ s.t. $vyu - p = \prod_i q_i^{n_i}$ is $L[1/2]$-smooth.
- compute $\log u \equiv \sum_i (n_i - m_i) \log q_i - \log(H + c)$.

Step 3: combine to find

$$\log h = -w + \sum_i e_i \log q_i + \sum_i f_i \log u_i.$$

Overall complexity is $O(L[1/2])$.

Improved smoothing

Classical technique: write $g^i h = u/v$ where $u, v = O(\sqrt{p})$ using Euclid's algorithm; use medium primes + q-descent; use (1 or 2) large primes.

Joux/Lercier: see this as reducing the lattice

$$\begin{pmatrix} z & p \\ 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} A_1 & A_2 \\ B_1 & B_2 \end{pmatrix}$$

and remark that

$$g^i h \equiv z \equiv \frac{A_1}{B_1} \equiv \frac{A_2}{B_2} \equiv \frac{k_1 A_1 + k_2 A_2}{k_1 B_1 + k_2 B_2}$$

$\Rightarrow$ sieve on (k_1, k_2) and use 2 large primes.

See later specific NFS smoothing.