Cryptanalysis – Project
Master 2–DI–ENS de Lyon 2019-2020

Guillaume Hanrot and Damien Stehlé

CRYPTANALYSIS – PROJECT.

The homework is to be handed back before 15/01/2020, 23:59. In electronic format to `guillaume.hanrot@ens-lyon.fr` *and* `damien.stehle@ens-lyon.fr`. *The clarity of the code and the overall presentation will be taken into account in the evaluation.*

The goal of this project is to implement some algorithms described in class, or some applications of these algorithms. The recommended language for this project is Sage/Python, though for programming lovers with plenty of time C+GMP might offer a more interesting challenge (you might need external libraries for some components).

1 Exercise 1: A few factoring / discrete log algorithms

1.1 Pollard ρ for factoring

Program Floyd's variant of Pollard ρ method (using epact) for factoring with $f(x) = x^2 + 1 \bmod N$.
Application : $N = 60331193824455101058028269521753$,
$N = 2764749333879647734604195328573859286669681$.

1.2 Pohlig-Hellman algorithm for DL

Implement Pohlig-Hellman algorithm for computing discrete log when the group order factorization is computable.

Example: $G = (\mathbb{Z}/p\mathbb{Z})^*$, $p = 138278216702273536 01$, $g = 3$, $h = 107809091741645010 09$.

1.3 $p+1$ algorithm for factoring

Implement the $p+1$ method (recall that to get a starting point (a, b) in $T_2(\mathbb{Z}/N\mathbb{Z})$, you have to choose a, b, and define $D = (1 - a^2)/b^2 \bmod N$. You may have to repeat a few times to get a factor (only half the D work).

Examples :
$N = 958535448642502991114 09$ (take $B = 2100$).
$N = 74648282401223830866161949113577350333338506436676205\ 995761855483$
$573844956741857881725 3229$.

(restrict the primes of the product defining $\mathcal{B}$ to be among the first 1000 ones, and take $B = 40000000000$).

1.4 Adleman's / Dixon's algorithm

Implement either Adleman's algorithm (the plain one with relations found by factoring g^a for random a) for discrete log, or Dixon's algorithm for factoring. You can actually do both at little extra cost: most of the machinery is common.

Report on your final choices – do not pick too large a factor basis since sage's linear algebra over $\mathbb{Z}/n\mathbb{Z}$ seems pretty lame.

You may gain significantly in terms of efficiency by implementing early abort strategy and/or large prime variation, but this is purely optional. You may also play with sieving ideas (this is easier for factoring).

Examples for DL : $G = (\mathbb{Z}/p\mathbb{Z})^*$, $p = 10000000259$, $g = 2$, $h = 7038304916$
$G = (\mathbb{Z}/p\mathbb{Z})^*$, $p = 1000000000005719$, $g = 11$, $h = 492328621286001$
$G = (\mathbb{Z}/p\mathbb{Z})^*$, $p = 100000000000000000039$, $g = 3$, $h = 56088846212909947255$ (feasible with a crude implementation of Adelman, but rather long).

Examples for factoring : $N = 8591966237$, $N = 2251802665812493$, $N = 73786976659910426999$.

2 Exercise 2: An application of Coppersmith's method

In 1998, Takagi proposed to use a modulus N of the form $p \cdot q^r$ with p, q prime and $r \geq 2$, rather than $p \cdot q$, in order to accelerate RSA decryption.

1- Explain why Takagi's idea helps accelerating RSA decryption.

2- Assume that $|\log_2 p - \log_2 q| \leq O(1)$. Using Coppersmith's method, show that one can factor pq^r in polynomial time when $r = \Omega(\log p)$.

3- Let N be as follows:
264281542590159029356089748891272231955653390811996944900260592349786398330829961448354899779696461685634524435425222926912905962158417545166424592733185577977759316121768584362425272409014052639676778985766113063317512207456970318590307683838925081157146302884609814664803645187090171504941635684854738644013476632024605221403960192334245934526904355324600265360679539001374081987621790146657201096025378770233470383970102781325591629292365139475970224294858059230403654763416152756378472684204644360313130892675643439899592893603255926530117231997356661350759298228510599507660377993360507002371881622197065049932987.
The integer N is of the form $p \cdot q^3$, with $|q - \overline{q}| \leq 2^{320}$ and $\overline{q}$ as follows:
30185903299161069037459501612758221021867921684629726838215078020335523872946301938053575955400221735067890965825658069724406927120922045769729509411690034.
Find p and q.