Cryptography, winter 2015/16

MICHAEL NÜSKEN, SIMON SCHNEIDER

11. Exercise sheet
Hand in solutions until Saturday, 30 January 2016, 12:00

Exercise 11.1 (An example of Pollard's ϱ method). (10+4 points)

(i) Complete the table below, which represents a run of Pollard's ϱ algorithm for $N = 100181$ and the initial value $x_0 = 399$, up to $i = 6$. [6]

i	x_i rem N	x_i rem 17	x_{2i} rem N	x_{2i} rem 17	$\gcd(x_i - x_{2i}, N)$
0	399	8	399	8	100181
1	...	...	...	...	...

(ii) The smallest prime divisor of N is 17. Describe the idea of the algorithm by looking at x_i rem 17 and x_{2i} rem 17 and in particular, why we stopped at $i = 6$. [4]

(iii) Complete the factorization of N using Pollard's ϱ algorithm. [+4]

Exercise 11.2 (Discrete logarithm). (0+12 points)

Choose one of the presented discrete logarithm algorithms and determine the discrete logarithm of 3 with basis $g = 2$ in the group $\mathbb{Z}_p^\times$ where p is the smallest 42-bit prime. *Hint*: $2^{41} \leq p < 2^{42}$! [+12]

Exercise 11.3 (Key sizes). (5 points)

On the website `http://www.keylength.com/` you find various methods for extrapolating the security level for public key sizes. Select two of them and throughly compare the two estimates. Decide which one you would follow. Why are all recommendations time-dependent? [5]

Exercise 11.4 (Security estimate). (5 points)

The best known factoring algorithms achieve the following (heuristic, expected) running times:

method	year	time for κ-bit integers
trial division	$-\infty$	$\mathcal{O}^{\sim}\left(2^{\frac{\kappa}{2}}\right)$
Pollard's $p-1$ method	1974	$\mathcal{O}^{\sim}\left(2^{\frac{\kappa}{4}}\right)$
Pollard's ϱ method	1975	$\mathcal{O}^{\sim}\left(2^{\frac{\kappa}{4}}\right)$
Pollard's and Strassen's method	1976	$\mathcal{O}^{\sim}\left(2^{\frac{\kappa}{4}}\right)$
Morrison's and Brillhart's continued fractions	1975	$2^{\mathcal{O}(1)\kappa^{\frac{1}{2}}\log_2^{\frac{1}{2}}\kappa}$
Dixon's random squares	1981	$2^{(\sqrt{2}+o(1))\kappa^{\frac{1}{2}}\log_2^{\frac{1}{2}}\kappa}$
Lenstra's elliptic curves method	1987	$2^{(1+o(1))\kappa^{\frac{1}{2}}\log_2^{\frac{1}{2}}\kappa}$
quadratic sieve		$2^{(1+o(1))\kappa^{\frac{1}{2}}\log_2^{\frac{1}{2}}\kappa}$
general number field sieve	1990	$2^{((64/9)^{\frac{1}{3}}+o(1))\kappa^{\frac{1}{3}}\log_2^{\frac{2}{3}}\kappa}$

It is not correct to think of $o(1)$ as zero, but for the following rough estimates just do it, instead add a $\mathcal{O}(1)$ factor. Factoring the 768-bit integer RSA-768 needed about 1500 2.2 GHz CPU years (ie. 1500 years on a single 2.2 GHz AMD CPU) using the general number field sieve. Estimate the time that would be needed to factor an κ-bit RSA number assuming the above estimates are accurate with $o(1) = 0$ (which is wrong in practice!)

1 (i) for $\kappa = 1024$,

1 (ii) for $\kappa = 2048$ (2015 recommendation),

1 (iii) for $\kappa = 3072$.

2 (iv) Now assume that the attacker has 1000 times as many computers and 1000 times as much time as in the factoring record. Which κ should I choose to be just safe from this attacker?

Remark: The statistics for discrete logarithm algorithms are somewhat similar as long as we consider groups $\mathbb{Z}_p^\times$. For elliptic curves (usually) only generic algorithms are available with running time $2^{\frac{\kappa}{2}}$.