The art of cryptography: security, reductions, and group cryptography

Prof. Dr. Joachim von zur Gathen, Konstantin Ziegler

11. Assignment

(Due: Sunday, 1 July 2012, 23^{59})

Exercise 11.1 (generating safe primes)**.** We continue with two sets of primes that you already encountered in Assignment 9

$$
\begin{aligned}
P_1 &= \{p \text{ prime}\colon p = 2\ell + 1 \text{ for some prime } \ell\},\\
P_2 &= \{p \text{ prime}\colon p = 2\ell_1\ell_2 + 1 \text{ for some primes } \ell_1, \ell_2\}.
\end{aligned}
$$

For a prime p from P_2, we have $n = (p-1)/2 = \ell_1\ell_2$ an RSA number. To prevent factorization by trial division or Fermat factorization many standards for RSA numbers require the primes to be sufficiently large on the one hand and also bounded away from $\sqrt{n}$. More precisely, we have two parameters α and δ, with $0 < \alpha < \delta < 1/2$ such that

$$n^{\alpha} \leq \ell_1 < n^{1/2-\delta}. \qquad (*)$$

(i) (2 points) Show that condition (*) can be expressed as

$$\ell_1^A < \ell_2 < \ell_1^B$$

with suitable values A and B depending on α and δ.

We let

$$P_2(A, B) = \{p \text{ prime}\colon p = 2\ell_1\ell_2 + 1 \text{ for some primes } \ell_1, \ell_2 \text{ with } \ell_1^A < \ell_2 < \ell_1^B\}$$

and take a look at the following algorithm that generates primes from $P_1 \cup P_2(A, B)$.

```
**Algorithms 1:** generating safe primes
  **Input**: positive $x$ and $1 < A < B < 3$
  **Output**: prime $p \in P_1 \cup P_2(A, B)$ with $x/\log^2 x \leq p \leq x$
  Repeat until a prime is returned
  Choose a random prime $\ell \in [x/(2\log^2 x) \ldots (x-1)/2]$
  If $p = 2\ell + 1$ is prime then return $p$
  Choose random prime $\ell_1 \in [(x/(2\log^2 x))^{1/(B+1)} \ldots (x/2)^{1/(A+1)},]$
  Choose random prime $\ell_2 \in [(x-1)/(2\ell_1 \log^2 x) \ldots (x-1)/(2\ell_1)]$
  If $p = 2\ell_1\ell_2 + 1$ is prime then return $p$
```

(ii) (3 points) To prove that the output satisfies the specification, it is sufficient to show that ℓ_2 satisfies the defining inequalities of $P_2(A, B)$. Prove it.

(iii) (6 points) Let $A = 1.04$ and $B = 2.62$ and run the above algorithm for increasing values of x. Compare the number of primes found from P_1 to the number of primes from $P_2(A, B)$ and compare the average number of loop executions to $\log^2 x$.

(iv) (+4 points) Consider only the generation of primes from $P_2(A, B)$ and split the interval from which ℓ_1 is chosen into ten equal pieces. Check for each piece how many (ℓ_1, ℓ_2) you generate with ℓ_1 in it.