Cryptology Exercise Week 8

Zijun Yu 202203581

Octobor 2023

Repeated Squaring

Question 1

We prove by induction that the algorithm is correct. First, at the end of the first iteration where $i = k-1$, we have

$$x = (1^2 \text{ mod } n) \cdot a^{z_{k-1}} \text{ mod } n = a^{z_{k-1}} = a^{Z_i}$$

Then suppose at the begining of some n'th iteration, $x = a^{Z_i} = a^{z_{k-1}z_{k-2}\ldots z_i}$ holds. At the end of this iteration, we have

$$\begin{aligned} x &= (a^{z_{k-1}z_{k-2}\ldots z_i})^2 \cdot a^{z_{i-1}} \text{ mod } n \\ &= a^{z_{k-1}z_{k-2}\ldots z_i 0} \cdot a^{z_{i-1}} \text{ mod } n \\ &= a^{z_{k-1}z_{k-2}\ldots z_i z_{i-1}} \text{ mod } n \\ &= a^{Z_{i-1}} \text{ mod } n \end{aligned}$$

Hence by induction, we have $x = a^{Z_0} = a^z$ at the end of the algorithm.

Question 2

In each iteration, we have to compute x^2 mod n. On average, half of the k bits are 1, so in half of the iterations, we have to compute $x \cdot a^{z_i}$ mod n. Therefore, the expected number of multiplications is

$$(1 + \frac{1}{2}) \cdot k = k + \frac{1}{2}k$$

Question 3

1. $x := 1$
2. For $i = k-1, k-3, ..., 0$, do
   (a) $x := x^2$ mod n
   (b) $x := x^2$ mod n
   (c) $x := x \cdot a^{z_i z_{i-1}}$ mod n ($a^{z_i z_{i-1}}$ is either a or the a^2 or a^3 we have computed beforehand, this step is empty if $z_i z_{i-1} = 00$)
3. Return x

At the last iteration, if we have less than 2 bits left, we do the two steps from the original algorithm.

The two square operations will append two 0's to the exponent of $a^{z_k z_{k-1} \cdots z_i}$, so it is trivial to see that the algorithm is correct by the same induction proof as in Question 1.

In each iteration, the possibility that we can skip $x \cdot a^{z_i z_{i-1}}$ is 1/4. Assume that 2 devides k, we then have exactly $k/2$ iterations. So the expected number of multiplications is

$$(2 + \frac{3}{4}) \cdot \frac{k}{2} + 2 = \frac{11}{8}k + 2$$

As long as k is reasonly big, which it is, the new algorithm is faster.

Question 4

If we scan n bits at once, the expected number of multiplications, assuming that n divides k, is

$$\begin{aligned} &(n + \frac{2^n - 1}{2^n}) \cdot \frac{k}{n} + 2^n - 2 \\ =&k + \frac{2^n - 1}{n2^n} k + 2^n - 2 \end{aligned}$$

Where $2^n - 2$ comes from computing a^2, a^3, ..., a^{2^n-1}. With a fixed k, the number first goes small as n increases, but then goes up again as n increases. And with k increasing, the optimal choice of n is also increasing. For example, if $k = 100$, the optimal n is 3, and if $k = 1000$, the optimal n is 5.