AARHUS UNIVERSITY - COMPUTER SCIENCE DEPARTMENT

Cryptology - Handin 8

Peter Burgaard - 201209175

November 17, 2016

The "Repeated squaring"-algorithm for exponentiation is essential for all known public-key crypto-systems. It comes in several variants, scanning the exponent from the least or the most significant end. This exercise shows that scanning from the most significant end has some advantages in terms of the available optimizations:
Assume we want to compute $a^z \mod n$, where z is a k-bit number, and where z is $z_{k-1}z_{k-2}...z_0$ in binary notation (all z_i are 0 or 1).

The algorithm does the following:

1. $x := 1$
2. For $i = k-1,...,0$ do:
   a) $x := x^2 \mod n$
   b) $x := x \cdot a^{z_i} \mod n$ (note that this step is empty if $z_i = 0$)
3. return x

Let Z_i be the number which is binary notation is $z_{k_1}z_{k_2}...z_i$

SHOW THAT AFTER EACH EXECUTION OF STEP 2, WE HAVE $x = a^{Z_i} \mod n$ AND CONCLUDE THAT THE ALGORITHM RETURN $x = a^z \mod n$

In step 2, we see that $x = 1$ until the first 1-bit i encountered in a. At that point, $x = a$. From here on forward, x is squared in each iteration. Squaring in binary, is basicly shifting the whole binary number one position to the left, and adding zero at the end. Now we are done with step a.

If the next number encountered in Z_i is 0 we can skip step b, and we are done with this iteration. If not we have to multiply x with another a. We see from previous steps that the exponent in binary ends with zero, since it have been squared, and the multiplication of a just flips the last bit of the exponent to 1.

We se that we have went through and iteration, and adjusted x accordingly.

IF THE BITS IN z ARE RANDOMLY CHOSEN, WHAT IS THE EXPECTED NUMBER OF MULTIPLICATIONS MOD N DONE IN THE ALGORITHM?

We always have to a multi-mod-n for all k iteration in step a. When the number of 1's are randomly chosen then $P(z_i = 1) = \frac{1}{2}$. Since we only have to do step b if $z_i = 1$, we see that the number of multi-mod-n's are $k + \frac{k}{2}$.

THE ALGORITHM CAN BE OPTIMIZED: ASSUME THAT WE FIRST COMPUTE THE VALUE $a^2 \mod n$ AND $a^3 \mod n$, THIS CAN BE DONE USING 2 MULTIPLICATIONS. SO WE NOW KNOW $a^n \mod v$ FOR ANY VALUE OF v THAT CAN BE WRITTEN WITH 2 BITS. DESCRIBE A VARIANT OF THE ALGORITHM ABOVE WHICH READS OFF 2 BITS OF THE EXPONENT IN ONE STEP. ARGUE THAT IT IS CORRECT AND GIVE THE EXPECTED NUMBER OF MULTIPLICATIONS YOU NEED FOR A RANDOM EXPONENT.

1. $x := 1$
2. For $i = k-1, k-3, .., 0$ do
   a) $x := x^4 \mod n$
   b) if $z_i z_{i-1} = 11$ then
      i. $x := x \cdot a^3$
   c) if $z_i z_{i-1} = 10$ then
      i. $x := x \cdot a^2$
   d) if $z_i z_{i-1} = 01$ then
      i. $x := x \cdot a$
   e) if $z_i z_{i-1} = 00$ then
      i. $x := x$

Proof. The loop in step 2 now runs over 2 bits at a time, so we modify the previous algorithm, and add additional cases. In step 2a we set $x := x^4 \mod n$, since this is equivalent to shifting the binary x two bits, which is what we want since we are looking two bits forward in z. Now follows a case for each outcome of two bits in z, where x is multiplied by the base 10 version of the binary number from z. By the same concept of the original algorithm, this returns the wanted result. □

Since we only set $x := x^4 \mod n$ half the times of the original algorithm, we perfom $\frac{k}{2}$ multi-mod-n's here. The chance of entering a case with a multi-mod-n is $P(z_i z_{i-1} \neq 00) = \frac{3}{4}$, and since the loop iterates $\frac{k}{2}$ times, this step has $\frac{3k}{8}$. Last place is in the precomputations of $a^2 \mod n$ and $a^3 \mod n$ which gives us two additonal multi-mod-n's, which gives a total of $2 + \frac{k}{2} + \frac{3k}{8}$ multi-mod-n's.