

Mathematics and Cryptography

Course 3

Jean-Sébastien Coron

University of Luxembourg

September 28, 2009

- Modular exponentiation
 - The square and multiply algorithm

Modular exponentiation

- We want to compute $c = a^b \pmod n$.
 - Example: RSA
 - $c = m^e \pmod N$ where m is the message, e the public exponent, and N the modulus.
- Naive method:
 - Multiplying a in total b times by itself modulo n
 - Very slow: if b is 100 bits, roughly 2^{100} multiplications !

Square and multiply algorithm

- Let $b = (b_{\ell-1} \dots b_0)_2$ the binary representation of b

- $b = \sum_{i=0}^{\ell-1} b_i \cdot 2^i$

- Square and multiply algorithm :

- Input : a , b and n

- Output : $a^b \pmod n$

- $c \leftarrow 1$

- for $i = \ell - 1$ down to 0 do

- $c \leftarrow c^2 \pmod n$

- if $b_i = 1$ then $c \leftarrow c \cdot a \pmod n$

- Output c

- Let B_i be the integer with binary representation $(b_{\ell-1} \dots b_i)_2$
 - $B_i = \sum_{j=i}^{\ell-1} b_j \cdot 2^{j-i}$
 - $B_{i-1} = 2 \cdot B_i + b_{i-1}$
- Claim : let c_i be the value of c at the end of step i :

$$c_i = a^{B_i} \pmod n$$

- Claim is true for $i = \ell - 1$
 - $B_{\ell-1} = b_{\ell-1}$
 - $c_{\ell-1} = 1$ if $b_{\ell-1} = 0$ and $c_{\ell-1} = a$ if $b_{\ell-1} = 1$
 - $c_{\ell-1} = a^{b_{\ell-1}} = a^{B_{\ell-1}} \pmod n$

- Assume that claim is true for i .

- Then $c_i = a^{B_i} \pmod n$
- $c_{i-1} = (c_i)^2 \pmod n$ if $b_{i-1} = 0$
- $c_{i-1} = (c_i)^2 \cdot a \pmod n$ if $b_{i-1} = 1$

$$c_{i-1} = (c_i)^2 \cdot a^{b_{i-1}} \pmod n$$

$$c_{i-1} = (a^{B_i})^2 \cdot a^{b_{i-1}} \pmod n$$

$$c_{i-1} = a^{2 \cdot B_i + b_{i-1}} = a^{B_{i-1}} \pmod n$$

- The output value c is $c = c_0$
 - $c_0 = a^{B_0} \pmod n$ and $B_0 = b$ gives

$$c = a^b \pmod n$$