Computer Science and Engineering, UCSD
CSE 107: Introduction to Modern Cryptography
Problem Set 5 Solutions

Winter 09
Instructor: Mihir Bellare
March 4, 2009

Problem Set 5 Solutions

Problem 1. [**40 points**] Consider the following computational problem:

INPUT: N, a, b, x, y where $N \geq 1$ is an integer, $a, b \in \mathbf{Z}_N^*$ and x, y are integers with $0 \leq x, y < N$
OUTPUT: $a^x b^y \bmod N$

Let $k = |N|$. The naive algorithm for this first computes $a^x \bmod N$, then computes $b^y \bmod N$, and multiplies them modulo N. This has a worst case cost of $4k+1$ multiplications modulo N. Design an alternative, faster algorithm for this problem that uses at most $2k+1$ multiplications modulo N.

Let us first explain the claim about the naive algorithm. On inputs N, a, b, x, y it would do the following:

$A \leftarrow$ MOD-EXP(a, x, N)
$B \leftarrow$ MOD-EXP(b, y, N)
$z \leftarrow$ MOD-MULT(A, B, N)
Return z

The algorithm MOD-EXP was presented in class. Section 9.2.6 of the chapter on Computational Number Theory presents a more general algoorithm EXP_G for exponentiation in an arbitrary group. MOD-EXP is simply the special case of the latter in which the group G is $\mathbf{Z}_N^*$, and its properties are listed in the table of Figure 9.1. Each iteration of the **for** loop of that algorithm uses two modular multiplications, the first to obtain $w = y^2 \bmod N$ from y and the second to obtain $w \cdot a^{b_i} \bmod N$. Thus, MOD-EXP uses $2k$ modular multiplications in all. So the above naive algorithm uses $4k+1$ modular multiplications.

The faster algorithm extends the ideas of the modular exponentiation algorithm of Section 9.2.6 in the chapter on Computational Number Theory. It works as follows:

Alg FASTEXP(N, a, b, x, y)
Let $x_{k-1} \ldots x_1 x_0$ be the binary representation of x
Let $y_{k-1} \ldots y_1 y_0$ be the binary representation of y
$c \leftarrow ab \bmod N$
$z \leftarrow 1$
for $i = k-1$ downto 0 **do**
  if $x_i = 1$ and $y_i = 1$ **then** $z \leftarrow z^2 \cdot c \bmod N$
  if $x_i = 1$ and $y_i = 0$ **then** $z \leftarrow z^2 \cdot a \bmod N$
  if $x_i = 0$ and $y_i = 1$ **then** $z \leftarrow z^2 \cdot b \bmod N$
  if $x_i = 0$ and $y_i = 0$ **then** $z \leftarrow z^2 \bmod N$

return z

Since $0 \leq x, y < N$ and N is k-bits long, we know that x and y are also at most k bits long. Therefore, the number of iterations for the loop is at most k. Since each loop incurs at most two modular multiplications, the total number of multiplications in the for loop is $2k$. Adding the one multiplication done on the 4th line of the code to get c, we have that the total number of multiplications for FASTEXP is $2k+1$ as desired.
