Institut for Matematik og Datalogi
Syddansk Universitet

October 13, 2006
JFB

Cryptology – E06 – Week 7

Lecture, October 12

We finished all but section 5.9 of chapter 5 and covered probabilistic encryption from chapter 8. This included definition 8.2, the Quadratic Residuosity Assumption (Problem 8.1 on page 338 in the textbook), and the Goldwasser-Micali Public-key Cryptosystem (Cryptosystem 8.1 on page 346 in the textbook).

Lecture, October 26

We will finish chapter 5 and start on chapter 6.

Lecture, November 2

We will finish chapter 6 and cover the McEliece Cryptosystem (copied from an earlier edition of the textbook).

Problem session October 30

1. Do problem 5.26. It is easy to program with Maple. You will probably use a loop including something like `while (igcd(x2-x1,262063)=1) do ... end do;`.

2. Do problem 5.27. Note that it is sufficient to start with 507, skip everything between 517 and 528, and continue to 531, so you can do this interactively in Maple.

3. Do problem 5.28a.

4. Do problem 5.29. Note that for part c, $d = 9$ works.

5. Do problem 5.34.

6. Assume that $p \equiv q \equiv 3 \pmod 4$ and $n = pq$. Assume that r is chosen at random from the uniform distribution over $\mathbb{Z}_n^*$. Show that $r^2 \pmod n$ is a random quadratic residue modulo n from the uniform distribution. Show that $-r^2 \pmod n$ is a random quadratic nonresidue with Jacobi symbol $+1$, also from the uniform distribution over these values.

Assignment due Monday, November 20, 14:15

Note that this is part of your exam project, so it must be approved in order for you to take the exam in January, and you may not work with others not in your group. If it is late, it will not be accepted (though it could become the assignment you redo). You may work in groups of two or three.

Write a program which implements the Goldwasser–Micale Public-key Cryptosystem, which is described, starting on page 345 in the textbook. Use it to encrypt and decrypt a string of bits. Use the Miller–Rabin algorithm for primality testing finding random primes congruent to 3 modulo 4 to create your modulus.

To deal with the long numbers necessary, you may use Java, there is a class in java.math called BigInteger which should be efficient and easy to use. There is documentation available on IMADA's system at

http://www.imada.sdu.dk/Technical/Manpages/jdk1.3/docs/api/.

This cannot be accessed outside of IMADA. You may use most of the standard methods provided there, though not the routines for generating or testing random primes, for computing Jacobi or Legendre symbols, or for modular exponentiation. (You may test your own for efficiency and correctness by comparing your results to the ones given by the methods in the package.)

Please turn in your program and some output. You should write a brief report, explaining how your program should be used, and how long your program takes with different length moduli. What confidence level did you choose for primality checking?

You should send me your program and any extra files via e-mail (they can just be attachments in `pine`). But, in addition to the e-mail, I would like printed copies of everything.

If you prefer another language to Java, please come talk with me by Thursday, October 26.