Problem Set 8

Due on Monday, April 30, 2012.

Instructions This is an optional extra credit problem. The grade on it will replace the grade on your lowest problem set (unless of course the grade on this problem is lower still, in which case it will be ignored).

Work the problem below, prepare your answer in electronic form, and submit your solution using the submit script on the Zoo. Remember to specify "8" for the problem number argument to `submit`.

1: Problem Description

The goal of this assignment is to implement the EC ElGamal cryptosystem described in lecture 13 using the elliptic curve routines you implemented for problem set 7.

2: Assignment

You should implement three commands:

```
ecgenkey params pub prv
ecencrypt params pub ptext ctext
ecdecrypt params prv ctext ptext
```

All three routines take as first parameter an extended "domain parameters" file. `ecgenkey` generates a random ElGamal public key pair and writes the public part to file `pub` and the private part to file `prv`. `ecencrypt` uses the public key to encrypt the plaintext file `ptext`, writing the ciphertext to the file `ctext`. `ecdecrypt` uses the private key to decrypt the ciphertext file `ctext`, writing the plaintext to the file `ptext`.

Each byte of the plaintext file is treated as a separate message to be encrypted by EC ElGamal.[1] To encrypt a byte m, one first uses the Koblitz encoding method to find a point $X = P_m$. Then one encrypts X with EC ElGamal. To decrypt, one first recovers X and then uses Koblitz decoding method to recover m.

3: Data representation

The `params` file begins with the three parameters describing an elliptic curve as in problem set 7. Following in the same file are additional parameters needed for EC ElGamal:

- A base point G on the elliptic curve, represented by a pair of decimal arbitrary precision integers. This number should be used for Bob's number α in EC ElGamal.

[1] In real life, the message would be a long random bit string to be used as a session key for a symmetric cryptosystem. We satisfy ourselves with encrypting single bytes for simplicity, even while recognizing that this causes an enormous expansion in the size of the plaintext.

- An encoding parameter k for use by the Koblitz's encoding method. k determines the probability that the method will fail to find the point P_m to encode a message m. Note that Koblitz's method fails if the elliptic curve equation cannot be solved for any of the numbers x in the sequence $\{mk+1, mk+2, \ldots mk+(k-1)\}$.

The `pub` key file consists of the two points α and β describing the public encryption key. (Thus, the file has four big integers in all.)

The `prv` key file consists of Bob's secret integer (which is denoted by a on slide 40 of lecture 13, not to be confused with the coefficient a in the elliptic curve equation).

The plaintext file `ptext` can be any file. The ciphertext file `ctext` consists of a sequence of point pairs (Y_1, Y_2) that encrypt the point X chosen to represent a given plaintext byte. Each point is described by two integers, so each plaintext byte is encrypted by four whitespace-separated big decimal integers.

4: Deliverables

You should submit the following items:

(a) A `makefile`, all source code and header files needed to build your project.

(b) Test data files and the output from your code when run on them.

(c) A brief human-readable document with information about your code such as known bugs, procedures for building and running it, and anything else that might help the grader.

Please be aware that the submit script can only handle files, not whole directory trees.