RSA Assignment

Description

RSA (Rivest–Shamir–Adleman) is a public-key cryptosystem that is widely used for secure data transmission.

In a public-key cryptosystem, the encryption key is public and distinct from the decryption key, which is kept secret (private). An RSA user creates and publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers are kept secret. Messages can be encrypted by anyone, via the public key, but can only be decoded by someone who knows the prime numbers.

The security of RSA relies on the practical difficulty of factoring the product of two large prime numbers, the "factoring problem". There are no published methods to defeat the system if a large enough key is used.

This project aims to give you a sense how RSA works, and how it is hard to break it.

You are required to:

- Build a program that is able to **encrypt/decrypt** text using the *RSA* algorithm.
- Implement another program that tries to break RSA and get the correct private key.
- Analyze different key sizes (number of bits of n) and how it affects the speed of encryption/decryption and the time of algorithm breaking.

Alphabet

Here is the following alphabet conversion:

- 0-9 should return their value 0→0, 5→5, 9→9
- a should map to 10, b to 11, c to 12 etc. up to z→35
- Space should map to 36

We will use only 37 characters for our messages (26 lower case letters, 10 numbers and the space). If you find other characters in the plain text message (input message) such as punctuation, white space, etc., simply treat the characters as spaces and assign them the value of 36.

Character Conversion

Before you can implement RSA, you must encode an input message as a number. We will use the following scheme to convert plaintext (input) messages to numbers:

Convert any extra characters to spaces as specified above.

Group the plaintext into sets of five characters per group. If the last grouping does not have exactly five characters, then append some space to the end of the plaintext message to fill out the last grouping. Each group must have five characters.

Convert each group into a separate number: If the grouping is [c4,c3,c2,c1,c0] then the number is

$$\sum_{i=0}^{4} c_i . 37^i$$

Here is an example. Assume our plaintext grouping is [hi s7]. First we translate the characters into numbers:

c4 = ’h’ = 17
c3 = ’i’ = 18
c2 = ’ ’ = 36
c1 = ’s’ = 28
c0 = ’7’ = 7
Then we compute the plaintext number:
17 · 37^4 + 18 · 37^3 + 36 · 37^2 + 28 · 37^1 + 7 = 32,822,818

A decoding operation should be performed in the same way except that the process is reversed. A number is converted to its character grouping by using mod and div.

Deliverables

- A program that does encryption/decryption.
    - Should be able to run two instances of this program and start chatting.
        - Note: you can use sockets.
    - Take care of what should be public and what should be private.
- A program that does the attack using plain-ciphertext pairs.
- A report (pdf) containing the analysis results and your conclusions.
- README file explaining how to run your application.
- A video running and explaining *briefly* your code and conclusions. (should be less than **5 mins**)

You are free to use any language but make sure you can handle large numbers (hundreds of digits).
All students should submit their own-written code without the help of any external source. Copied projects from each other or from the internet will not be accepted.
Due Date: 28 March at 11:59 PM