**RSA Implementation**

Chi-Heng Hung CHH162

Ting-Wei Shen TIS50

Yu-Hsien Lee YUL198

**RSA Introduction**

Find two prime p and q

Compute n and Φ(n), where n=p\*q and Φ(n)=(p-1) (q-1)

Find prime number b

Compute a, where a = b-1 mod Φ(n)

Public Key: (n, b)

Private Key: (p, q, a)

Encryption: e(X) = Xb mod n

Decryption: d(Y) = Ya mod n

**Implementation**

**Computation**

Using Euclid’s Algorithm to find GCD of two numbers

Find GCD of X and Y

Using Euclid’s Totient function to find Φ(n)

Find Φ(n) = (p-1) (q-1) Because p & q are prime

Using Extended Euclid's Algorithm to find Multiplicative Inverse

Compute r and s, where r \* X + s \* Y = GCD(X, Y)

**Efficiency**

Square and multiply algorithm

Compute xb mod n

Express the exponent b as a binary number

b = bk-1 bk-2 bk-3 bk-4…b2 b1 b0

b = bk-1 2k-1 +bk-2 2k-2 + bk-3 2k-3 +…+b121 + b0

We can write it as

b = (((…(bk-12 +bk-2)2+bk-3)2+…+b1)2+b0

Algorithm

When bi =1, you square AND you multiply

When bi =0, you only square

Chinese Remainder Theorem

Compute ya mod n = X

Find yp and yq

yp = y mod p

yq = y mod q

Find ap and aq

ap = a mod (p-1)

aq = a mod (q-1)

Find Xp and Xq

Xp = ypap mod p

Xq = yqaq mod q

Find Up and Uq

Up = q-1 mod p

Uq = p-1 mod q

X = Xp q Up + Xq P Uq

**Efficiency Comparison**

一張含有 螢幕擷取畫面, 地圖 的圖片

描述是以高可信度產生When the key length exceeds around 800 bits long, the CRT algorithm starts to out perform than the one's without CRT algorithm.

**Usage**

run the RSA.jar file:

java -jar RSA.jar

User will be ask to input key length and message to input with the following command:

Please enter the length:

Please enter the message:

The key length should only be numbers, the message can be any characters.

Note that if the message has a longer length than the key, the encrypted and decrypted result will be garbled.

User can choose to use Chinese Remainder Theorem for the encryption and decryption or not with the following command:

Use Chinese Remainder Theorem? (T/F):

The Input should be only T, t, F or f, any other inputs will be considered as true.

The application will show the Public Key, Private Key, Ciphertext and decrypt result with the following command:

Public Key:

Private Key:

ciphertext:

decrypt result:

The Public Key, Private Key, Ciphertext will be encoded with Base64, and the decrypt result will be shown in normal text form.

The application will also show the time spent to encrypt and decrypt with the following command:

Time Spent:

The time will be shown in the unit of nanoseconds.