National University of Computer &
Emerging Sciences
Karachi Campus

Project Report
Programming Fundamentals
Section: E

Group Members:
19K-1378 Saad Berry
19K-0168 Muhammad Abdullah
19K-1441 Muhammad Iqbal

<u>Asymmetric Cryptography</u>
<u>Public-Key Cryptosystems</u>

<u>(RSA)</u>

Project Report

- **Introduction**

RSA (**Rivest–Shamir–Adleman**) is one of the first public key crypto systems and is widely used for secure data transmission. In such a crypto system, the encryption key is public and it is different from the decryption key which is kept secret (private).

Our aim for this project was to provide an easy implementation of RSA and overcome the problems which earlier versions of RSA have faced e.g (RSA 1024, 2048 keys).

- **Background**

Ron Rivest, Adi Shamir, and Leonard Adleman at the Massachusetts Institute of Technology, made several attempts over the course of a year to create a one-way function that was hard to invert. Rivest and Shamir, as computer scientists, proposed many potential functions, while Adleman, as a mathematician, was responsible for finding their weaknesses. They tried many approaches including "knapsack-based" and "permutation polynomials".

For a time, they thought what they wanted to achieve was impossible due to contradictory requirements. In April 1977, they spent Passover at the house of a student and drank a good deal of Manischewitz wine before returning to their homes at around midnight. Rivest, unable to sleep, lay on the couch with a math textbook and started thinking about their one-way function. He spent the rest of the night formalizing his idea, and he had much of the paper ready by daybreak. The algorithm is now known as RSA – the initials of their surnames in same order as their paper.

- **Project Specifications**

The project uses different built-in C libraries to solve the mathematics behind RSA. Apart from built-in C libraries, we have used GMP (GNU Multiple Precision Arithmetic Library) . GMP is a free library for arbitrary precision arithmetic, operating on signed integers, rational numbers, and floating-point numbers. There is no practical limit to the precision except the ones implied by the available memory in the machine GMP runs on. GMP has a rich set of functions, and the functions have a regular interface.

- **Problem Analysis**

The goal behind building this project was to implement RSA-4096 (here 4096 refers to the bits in key), which is considered safe for encryptiondecrytion, unlike the RSA 1024 keys which are already crackable by shor's factoring algorithm or by brute force attack.

Table below gives the number of operations needed to factor n with Schroeppel's method, and the time required if each operation uses one microsecond, for various lengths of the number n (in decimal digits).

Digits	Number of operations	Time
50	1.4×10^{10}	3.9 hours
75	9.0×10^{12}	104 days
100	2.3×10^{15}	74 years
200	1.2×10^{23}	3.8×10^{9} years
300	1.5×10^{29}	4.9×10^{15} years
500	1.3×10^{39}	4.2×10^{25} years

- **Solution Design**

1. Generate two large random primes, p and q, of approximately equal size such that their product $n=pq$ is of the required bit length, e.g. 2048 bits.
2. Compute $n=pq$ and $\phi=(p-1)(q-1)$
3. Choose an integer e, $1<e<\phi$, such that $\gcd(e,\phi)=1$
4. Compute the secret exponent d, $1<d<\phi$, such that $ed\equiv 1 \bmod \phi$
5. The public key is (n,e) and the private key (d,p,q). Keep all the values d, p, q and ϕ secret. [Sometimes the private key is written as (n,d) because you need the value of n when using d. Other times we might write the key pair as $((N,e),d)$

n is known as the modulus.

e is known as the public exponent or encryption exponent or just the exponent.

d is known as the secret exponent or decryption exponent.

- **Implementation + Testing**

The implementation is done by using both the modules (RSA.h and rsabig.h).

1. The function isPrime checks if p and q are prime then continues to the next step.
2. The function calculatePHIN computes the value of phi-n function.
3. After this, the value of e is chosen randomly (must be between 1 and phi-n and coprime with phi-n and n).
4. Once the value of e is chosen, it is passed to decExponent function along with phi-n and n. The value of d is computed.
5. Once the value of d and e are computed, the user is prompted to enter some text which is encrypted, and the encrypted text is saved into a file.
6. Later, the encrypted text is read from the file and decrypted.

Note:

rsabig.h module works in the same way, but it takes big integers and generate large values of d, e, n and phi-n, which prevents it from brute-force attacks and shor's factoring algorithm.

Testing:

The primes p and q used while testing are in dummyPrimes.txt file. Later, change in the values of d and n while decrypting has produced random characters instead of actual message which means algorithm is working.

Note:

GMP library is a prerequisite for running the code.

- **Project Structure**

The Project is divided into four modules including main file.

1. RSA.h:
   This file consists of the functions which are used in the main to compute the keys. The code is written by Muhammad Abdullah.
2. RSA.c and fileopr.h:
   fileopr.h consist of the file operation functions which also format the text before exporting to files and importing as well. RSA.c consists of main function and all the headers. Fileopr.h and some part of RSA.c is written by Saad Berry.
3. rsabig.h and fileopr.h (Decoder function):
   rsabig.h module consists of key computation functions for RSA-4096. Rsabig.h and Decoder in fileopr.h is written by Muhammad Iqbal.

- **Results**

A secure cryptosystem that can be used in different end to end data transmission systems, ranging from messaging clients to file encryption, establishing secure connection over internet, browsing, protocols and so on.

- **Conclusion**

RSA is a method for implementing a public-key cryptosystem whose security rests in part on the difficulty of factoring large numbers. If the security of this method proves to be adequate, it permits secure communications to be established without the use of couriers to carry keys, and it also permits one to “sign” digitized documents.

The security of this system needs to be examined in more detail. In particular, the difficulty of factoring large numbers should be examined very closely. Once the method has withstood all attacks for a sufficient length of time it may be used with a reasonable amount of confidence.