ASSIGNMENT 5 DESIGN DOCUMENT

Chris Moon

February 2023

1 Goal

Implement the SS method of file encryption. Write an function that generates public and private key pairs, as well as an encryptor function and decryptor function that uses said pair to encrypt and decrypt files.

In addition, write a short math library to handle the mathematics behind the SS encryption method.

2 Pseudocode

```
RANDOMIZER MODULE
    -used for the later math library
    -include gmp.h (don't forget to download gmp and pkg-config first)
    -includes functions to initialize and clear a random state variable, which is defined as an extern variable of type gmp randstate
    -use gmp randinit mt(state); and gmp randseed ui(state, seed); to set the state
    -use gmp randclear(state) to clear the state
MATH LIBRARY AND GMP
    -GNU multiprecision (GMP) library needed to support precise math on large numbers
    -GMP works like this: included gmp.h
    -also have to include the randomizer module header
    -Use and mpz type variable like this:
    -mpz var;
    -mpz init(var);
    -then use gmp functions, like mpz set() or mpz add() to set a value and do arithmetic operations on the variable
    -the math library needs a function for gcd, mod inverse, modular exponentiation, a prime tester, and a prime number generator
GCD
    -takes a and b, and sets g to the answer
    -while b isnt 0
    -b = mod a, b
    -a = the old value of b (use a temp variable)
MOD INVERSE
    -takes a and n, and sets o to the value
    -set up variables
    -r = n and r' = a
    -t = 0 and t' = 1
    -while r' is not 0:
    -set a value, q, to r/r'
    -swap the values of r and r' with r' and r-qr' respectively
    -swap the values of t and t' with t' and t-qt' respectively
    -outside the loop, check if r > 1
```

-if so, there is no answer, so return 0 (set o to 0).
-if t < 0, then add n to t and return t (set o to t).

<u>POW MOD</u>

-takes a, d, and n, and sets o to the answer
-set a value v to 1
-set a value p to a
-while d is positive:
-check if d is odd:
-if d is odd, set v to vp mod n
-after the if statement, p is set to pp mod n
-d = d/2
-at the end of the while loop, return v

<u>MAKE PRIME</u>

-check if the number is 2, since two is the only even prime
-if its not, and its even, return false
-solve for s and r, so that n-1 = 2 to the s, times r
-do this by looping until r is odd:
-r = r/2
-s = s+1
-after finding r and s, loop from 1 to the given number of iterations
-choose a random number from 2 - n-2 using urandomm
-set a variable y to power mod(a, r, n)
-if y isnt 1 and y isnt n-1:
-then y = pow mod(y,2,n)
-inside the same if: if y is 1, return false
-j = j+1
-outside the while, if y isnt n-1, return false
-default case is return true;

<u>MAKE PRIME</u>

-make a prime number using a random number generator
-while the number is composite, (check this using the is prime function from above), add 1 to the number and check again
-return the final number

<u>SS LIBRARY</u>

-includes functions to create a public key, private key, and write said keys to files
-also include functions to read keys from files, and to encrypt and decrypt files using said keys
-making the public key:
-need to allocate the total bits given between p and q, such that 2p + q = bits
-get a value p using random()
-since q = total bits - p
-if p is even, divide it by two, if not, take one bit from q, add it to p, then divide p by q
-this prevents loss of bits due to truncating division
-make a prime number using p and q bits
-n = p prime * p prime * nprime
-making private key:
-private key consists of d and pq
-d = mod inverse of n and lcm(p-1, q-1) where p and q are the primes created for the public key
-pq = p*q
-encrypting a file:
-basically, create a block array
-then run through the file using fread to append to the block

-use mpz import to get values from the block and encrypt them 1 by 1
-print the encrypted values to a file
-decrypting a file:
-again create a block
-use fscanf to read each hexstring per line
-decrypt each one and export to the block
-write the block to the file