Design Document Assignment 6

Abhinav Prasanna

November 22, 2021

Description

The goal of the assignment is to create RSA key tokens to encrypt and decrypt to teach students about cryptography. We must submit 6 different files encrypt, decrypt, keygen, numtheory, randstate, and rsa.

Decrypt

Decrypt works testing harness for our files to test the decryption of the RSA Files.

Pseudocode

OPTIONS Command Line options
getopt(with args and OPTIONS)
fopen()
if(fopen() fails)
print error
rsareadpriv(file)
print(verbose)
rsadecryptfile(user)
delete variables

Encrypt

Pseudocode

OPTIONS Command Line options
getopt(with args and OPTIONS)
fopen()
if(fopen() fails)

print error
rsareadpub(file)
print(verbose)
mpzsetstr(user,USERNAME)
rsaencryptfile(user)
delete variables

numtheory

Numtheory works as a c file that contains all the neccesary math theory functions.

Pseudocode

GCD(a,b)
while b!=0
t¡-b
b¡-a mod b
a¡-t
return a

modinverse()
mpzt temp;
mpzt modval1
mpzt modval2
mpzset(temp, o)
while (mpzcmp(mpzmodq(modval1, o, a), zeroval) ! = 0)
mpzadd(temp, temp, n)
if (mpzcmp(mpzmodq(modval2, temp, a), zeroval) == 0)
mpzmodq(i, mpzcdiv(temp, temp, a), n)
else
mpzset(i, temp)
powmod(i,a,n)
while(i mod n does not equal 0)
i = i+n
if(i mod n equals 0)
i = i/a mod n
break

makeprime(a)
mpznextprime(a)

isprime(a)
write n1 = 2 to the power of s times r such that r is odd
or i ←1 to k choose random a 2,3,...,n2
y = POWERMOD(a,r,n)
if y!= 1 and y!= n1
j ←1
while j s1 and y!= n1
y ←POWER-MOD(y,2,n)
if y == 1
return FALSE
j ←j+1
if y!= n1
return FALSE
return TRUE

randstate

Randstate is a C file that generates random seeds using GMP methods.

Pseudocode

void ranstateinit(seed
gmprandsinitmt(state)
gmprandseedui(state,seed)
gmprandclear(state)

RSA

RSA is a C file that generates the neccesary RSA functions for our lab. We will generate the functions rsa make public key, make private key, read public key, read private key, write public key, write private key, encrypt file, encrypt function, decrypt file, decrypt function, sign and verify. RSA will utilize all of the neccesary numtheory functions in the file.

Pseudocode

void rsamakepub(p,q,n,e,nbits,iters)
mpzinit(vars)
while(mpzsizeinbase(2.n) ¡ nbits)
set size between nbits/4 and (3*nbits/4)
qbits = nbits - size
makeprime(p) makeprime(q) mpzmul(n = p * q)

temp = p - 1
temp2 = q - 1
finalval = temp1 * temp2
random val from 0 to nbits-1
gcd of exponent and final val
while gcd !=0
generate random num between 0 to nbits-1
gcd of exponent and final val
return e

void rsawritepub(vars)
gmpfprintf(pbfile,values,vars)

void rsareadpub(vars)
scan(pvfile,values,vars)

void rsamakepriv(d,e,p,q)
p = p -1
q = q -1
d = e to -1 mod (p-1) (q-1)

void rsawritepriv(vars)
print(pvfile,values,vars)

void rsareadpriv(vars)
printf(pvfile,values,vars)

void rsaencrypt(c,m,e,n)
c = m to e mod n

void rsadecrypt(m,c,d,n)
m = c to d mod n

void rsadecryptfile(files,vars)
mpzinit(vars)
k = mpzsizeinbase(n,2) - 1 / 8
initialize array

scanf(file,value,c)
while(cant read file)
scanf(file,value,c)
rsadecrypt(vars)
mpzexport(array from m)
fwrite(array on outfile)

free(array)
mpzclears

void rsaencrypttfile(files,vars)
mpzinit(vars)
k = mpzsizeinbase(n,2) - 1 / 8
initialize array
while(cant read file)
readf(file,value,c)
mpzimport(m from array)
rsaencrypt(vars)
fprintf(outfile,values,c)

free(array)
mpzclears

void rsasign(vars)
s = $m^d mod n$

$voidrsaverify(m.s.e.n)t = stoemodnifm = treturntrueelsereturnfalse$

keygen

Generates RSA Key for a given user.

Pseudocode

int main(void) OPTIONS Command Line options
getopt(with args and OPTIONS)
fopen()
if(fopen() fails)
print error
fchmod()
fileno()
randstateinit(seed)

rsamakepub()
rsamakeprivate()
printf(Enter username)
scanf(Name)
getenv()
mpzt user
mpzsetstr(user,Name,62)
rsasign(user)
rsawritepub(user)
rsawritepriv(user)
if verbose : printf(username,signature,P,Q,N,Exponent,D)
rsaverify(user)
randstateclear()
delete variables

encrypt

Encrypts inputted RSA Key or RSA public key. Input file can default to standard in if errored. Output file can default to standard out if errored. RSA Public Key can default to rsa.pub if errorred.

Pseudocode

int main(void) OPTIONS Command Line options
getopt(with args and OPTIONS)
fopen()
if(fopen() fails)
print error
mpzinitialize(vars)
rsareadpub(vars)
mpzsetstr(uservalue = username)
if verbose = true: print(username,signature,n,exponent)
if(!rsaverify(vars)):exit()
rsaencryptfile(vars)
fclose(files)
mpzclear(vars)

decrypt

Decrypts inputted RSA Key or RSA private key. Input file can default to standard in if errored. Output file can default to standard out if errored. RSA Private Key can default to rsa.priv if errorred.

Pseudocode

int main(void)
OPTIONS Command Line options
getopt(with args and OPTIONS)
fopen()
if(fopen() fails)
print error
mpzinit(vars)
rsareadpriv(vars)
if verbose = true: print(n,private key e)
rsadecryptfile(vars)
fclose(files)
mpzclears(vars)