function [N,d,e] = RSA\_Gen(p,q)

% N= p \* q both p and q is the prime factor of the N

N=p\*q;

disp("N="+N);

% z is the relatively prime

z=(p-1)\*(q-1);

% find the suitable e which meets with requirement of the e.

for e=N-2:-1:2

% e and z is Coprime

if(gcd(e,z)==1)

break;

end

end

disp("e="+e);

for d=1:1:z

% d meets the requirement ed % z ==1

if(mod(e\*d,z)==1)

break;

end

end

disp("d="+d);

end

%c=m^e mod N

% c=RSA\_Enc(1235,54,2537);

% c=RSA\_Enc(56,53,10);

% c= m^e mod N we need find a quick way to calculate c

%1. change the decimalism of e to binary of e

%2. if the lowest bit of the e is 1 record the number after multiply and mod N

%3. if the lowest bit of the e is 0 just calculate the lastnum (sqrt and mod N)

function [c]= RSA\_Encryption(N,e,m) %Encrypt the public key (c)

i=1; % The node point to the situation of the n\_num

bin=length(dec2bin(e)); %gGet the size of the n\_num

n\_num = ones(1, bin); % Record the mod number (lastnum)

lastnum=sym(m); % The current base number

% Get the first binary of the bin\_num

% If it is 1 then store the lastnum to the n\_num

% n\_num is record the mod number

% lastnum is record the last time mod number

while e % Until the e is 0

if(bitget(e, 1)==1) % Get the lowest bit of the e

n\_num(i)=lastnum; % record the n\_num(i) is current base number

end

lastnum=mod(lastnum\*lastnum,N); % Sqrt the base number and mod to N

i= i+1; % Move the node to next

e=bitshift(e, -1); % Shift right one bit of the e

end

c=mod(prod(n\_num),N);% Multiple continuously of the n\_num(base number) and then mod to the N

disp(c);

end

% m = c^d mod N. we need to make the calculation faster and simplify the

% process.

function [m]=RSA\_Dec(N,d,c)

i=1;% The node point to the situation of the n\_num

bin=length(dec2bin(d));

n\_num = ones(1, bin); % Record the mod number (lastnum) initialize the array

lastnum=sym(c);% The current base number

while d % Until the d is 0

if(bitget(d, 1)==1) % Get the lowest bit of the d

n\_num(i)=lastnum; % record the n\_num(i) is current base number

end

lastnum=mod(lastnum\*lastnum,N); % Sqrt the base number and mod to N

i= i +1; % Move the node to next

d=bitshift(d, -1); % Shift right one bit of the d

end

m=mod(prod(n\_num),N);% Multiple continuously of the n\_num(base number) and then mod to the N

disp(m);

end

%% m= c^d mod N

% we know the N and c but don't know d

% so we need to calculate d , d can be calculated as mod(e\*d,z)==1

% z is equal to (p-1)\*(q-1) and N = p\*q

% so we should calculated P and q through N and thus we can know

% z and then calculate the d

% in conclusion we need calculate p

function [m]=RSA\_Hack (N,e,c)

format long;

% c:ciphertext m:plaintext N:N=p\*q e:e=gcd(e,z) d=mod(e\*d,z)=1

% e:公钥 d：私钥 m:明文 c:密文 N:质数相乘

% c= m^e mod N m = c^d mod N.

fa=factor(N);% set the Prime number array

p=fa(1);% ergodic the first Prime number

q=N/p; % get the q

z=floor((p-1)\*(q-1)); % get the z

for i=1:1:length(fa)%ergodic to find the secret key d

[g,u,~]=gcd(e,z); % e\*u + z\*~ = g . g is the common factor of the e and z

% if common factor is 1

% it means e\*u + z\*~ = 1 and then e\*u - 1 = -z\*~

% Because mod(e\*d,z)=1 means e\*d-1 %z =0

% so (e\*u -1) mod z = -z\*~ mod z = 0

% so d = u mod z to prevent the d is < 0

if(g==1)

d=mod(u,z);

break;

else % g ~= 1 so choose next prime factor to calculate gcd

p=fa(i);

q=N/p;

z=(p-1)\*(q-1);

end

end

m=RSA\_Dec(N,d,c); % calculate the m = c^d mod N.

disp(m);

end