Assymetric-Key Cryptography

RSA (Rivest, Shamir, Adleman)

Nama: Ramadhan Kalih Sewu

NPM: 1806148826

A. Jalannya Program

Menggunakan nilai yang disepakati oleh soal. Input yang dibutuhkan oleh algoritma RSA dalam membangun sebuah public dan private key hanyalah dua nilai eksponen yang merupakan bilangan prima (p dan q). Agar memastikan kecocokan dengan permintaan soal, kita sesuaikan nilai p, q, M, dan e.

Dengan ini, kita punya dua kunci bersifat khusus yang digunakan untuk:

- Public Key (Untuk enkripsi alias mengunci)
- Private Key (Untuk dekripsi alias membuka)

```
% Nilai p dan q kesepakatan soal
% Dalam dunia nyata, ini boleh di random
p = 8831;
q = 9769;

% Pesan yang ingin di enkripsi
M = 150;

% Generate Keys
[pub, priv, modulo] = GenerateRSA(p, q);

% Encryption RSA
C = FastExponent(M, pub, modulo);

% Decryption RSA
P = FastExponent(C, priv, modulo);

% Uji Kebenaran Hasil
display(P);
if (P == M) display('VALID!'); end
```

```
p= 8831
q= 9769
n= 86270039
w= 86251440
Public e = 1223, n= 86270039)
Private(d = 82584167, n= 86270039)
M= 150
C= 6755224
```

Setelah data di kunci menggunakan public key, satu-satunya cara untuk mengembalikannya kebentuk asli adalah dengan private key. Oleh karena itu, tidak masalah jika public key dan nilai modulo ini ter-ekspos melalui saluran yang tidak aman.

B. Pembuat Kunci RSA

Nilai e adalah greatest common divisor dari M (ø) yang bernilai sama dengan 1. Nilai tersebut sudah pasti lebih dari satu. Dalam algoritma RSA, kita dapat memilih salah-satu secara acak. Perlu diingat bahwa nilai e menjadi faktor dari rumus nilai d. Sehingga, perubahan nilai e akan merefleksikan kepada perubahan nilai d. Karena kita ingin menyocokkan dengan soal, maka kita atur paksa nilai e menjadi 1223. Jika tidak, fungsi saya akan mengambil alternatif nilai e terkecil yaitu 7.

```
function [public, private, modulo] = GenerateRSA(p, q)
    modulo = p * q;
    M = (p-1) * (q-1);
    % Cari nilai e dimana: gcd(e, M) = 1
    % Nilai e ada lebih dari satu, boleh dipilih secara bebas
    % Dalam fungsi ini, kita ambil yang paling kecil saja
    e = 1; out = 0;
    while(out ~= 1)
        e = e + 1;
        out = gcd(e, M);
    end
    % Line dibawah hanya berlaku untuk demonstrasi
    % dimana nilai p = 8831 & q = 9769;
    e = 1223;
    % Cari nilai d dimana: d * e = 1 mod M
    % atau dengan kata lain: (d * e) % M = 1
    d = 0; out = 0;
    while (out ~= 1)
        d = d + 1;
        out = mod(d * e, M);
    end
    public = e;
    private = d;
end
```

C. Output

Seluruh variabel sudah sesuai dengan deklarasi pada soal. Fungsi cipher dan decipher sudah bekerja sesuai dengan harapan. Dengan ini, implementasi RSA saya berhasil.

```
>> Source

P =

   150

VALID!
```

Name	Value
C	6755224
M	150
modulo	86270039
p	8831
P	150
priv	82584167
pub	1223
q	9769

Satu proses RSA ini hanya berlaku pada komunikasi satu arah. Yaitu dengan tujuan bahwa lawan bicara dapat mengirimkan pesan secara aman terhadap kita. Berikut simpulan skenario kegunaan RSA:

- Host A inisiasi dengan membuat kunci RSA
- Host A menyimpan private key & memberikan public key ke Host B
- Data yang dikirim dari Host B kepada Host A akan dikunci menggunakan public key yang diberikan oleh Host A.
- Host A membuka data dengan private key yang ia simpan

Agar terjadi komunikasi secara dua arah secara aman. Maka RSA harus di inisiasi oleh dua belah pihak.