Ujian Akhir Semester

Kriptografi

Nama: Ramadhan Kalih Sewu

NPM: 1806148826

1. Temukan (Crack) kunci publik RSA untuk Mendekripsi kunci AES

   Untuk mempercepat proses pencarian kunci publik, kita ambil potongan kode dari fungsi RSA yang sebelumnya telah dibuat untuk membangun kunci RSA. Kita tahu syarat public dan private key dari RSA adalah sesuai rumus berikut:

   $$d\ x\ e \equiv 1\ mod\ M$$

   Dengan: d adalah kunci private, e adalah kunci publik.

   Sehingga, dengan nilai d, p, dan q yang diketahui, kita dapat mengira-ngira nilai kunci publik.

```
%% Crack Public Key
p = 6959; q = 13417;
% cari nilai e dengan brute force
% dimana d * e kongruen dengan 1 mod M
modulo = p * q;
M = (p-1) * (q-1);
d = 12512191; out = 0; e = 0;
while (out ~= 1)
    e = e + 1;
    out = mod(d * e, M);
end
fprintf('Found Public Key: %d', e);
```

Selanjutnya, kita ingin mendekripsi sebuah bilangan dengan menggunakan RSA melalui kunci yang baru kita dapat sebelumnya. Nantinya, plaintext atau hasil decipher ini akan di gunakan sebagai kunci publik AES. Langkah detil telah diberikan pada komentar di kode dibawah:

```
%% RSA Decryption
% dekripsi RSA menggunakan fast exponent
% karena data dienkripsi menggunakan private key,
% maka public key kita gunakan untuk dekripsi
encryptedKeyAES = hex2dec('038D03A7');
originalKeyAES = FastExponent(encryptedKeyAES, e, modulo);
% padding, 8 digit hexadecimal untuk merepresentasi 1 kolom key AES
originalKeyAES = num2str(originalKeyAES,'%08x');
% pecah 2 karakter (digit hexadecimal) untuk membentuk 4 cell
col = cellstr(reshape(originalKeyAES,2,[])');
% format ke bentuk matrix 4x4 sebagai kunci AES
fKeyAES = cell(4);
fKeyAES(:,:) = {'00'};
fKeyAES(:,1) = col
% ubah ke format decimal untuk diproses AES
fKeyAES = reshape(hex2dec(fKeyAES), size(fKeyAES));
% Penyesuaian key, baris ke 3 ditambah 1
fKeyAES(3,1) = fKeyAES(3,1) + 1;
```

Jangan lupa penyesuaian kunci AES pada baris 3 kolom 1 dimana nilainya di-increment 1. Berikut kunci AES yang seharusnya:

```
Valid
Found Public Key: 1231
fKeyAES =

     0     0     0     0
    62     0     0     0
   139     0     0     0
   177     0     0     0
```

Public Key RSA (e: 1231, n: 93368903);

Plain Teks AES: 4098737 $(003e8ab1)_{16}$

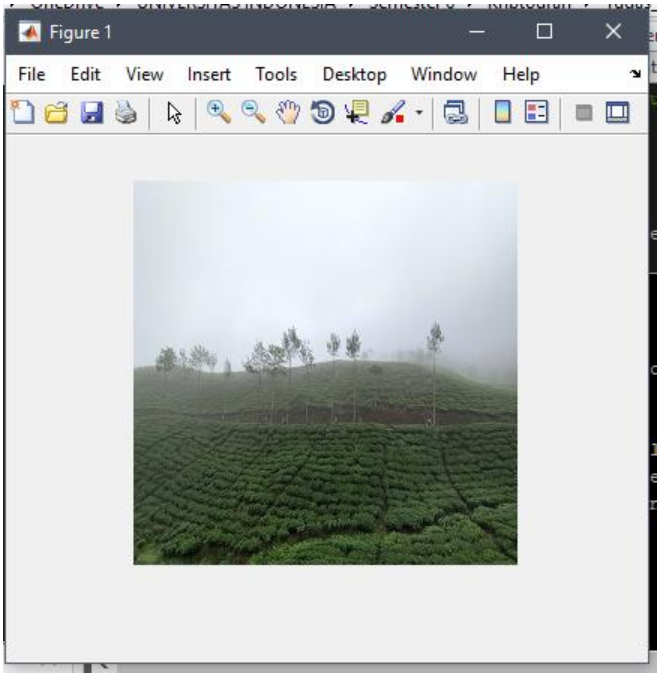
AES Setelah Penyesuaian: 4098993 $(003e8bb1)_{16}$

Dengan dipecah tiap byte sehingga 00 = 0, 3e = 62, 8b = 139, b1 = 177.

2. Decipher gambar menggunakan AES 4x4 matriks

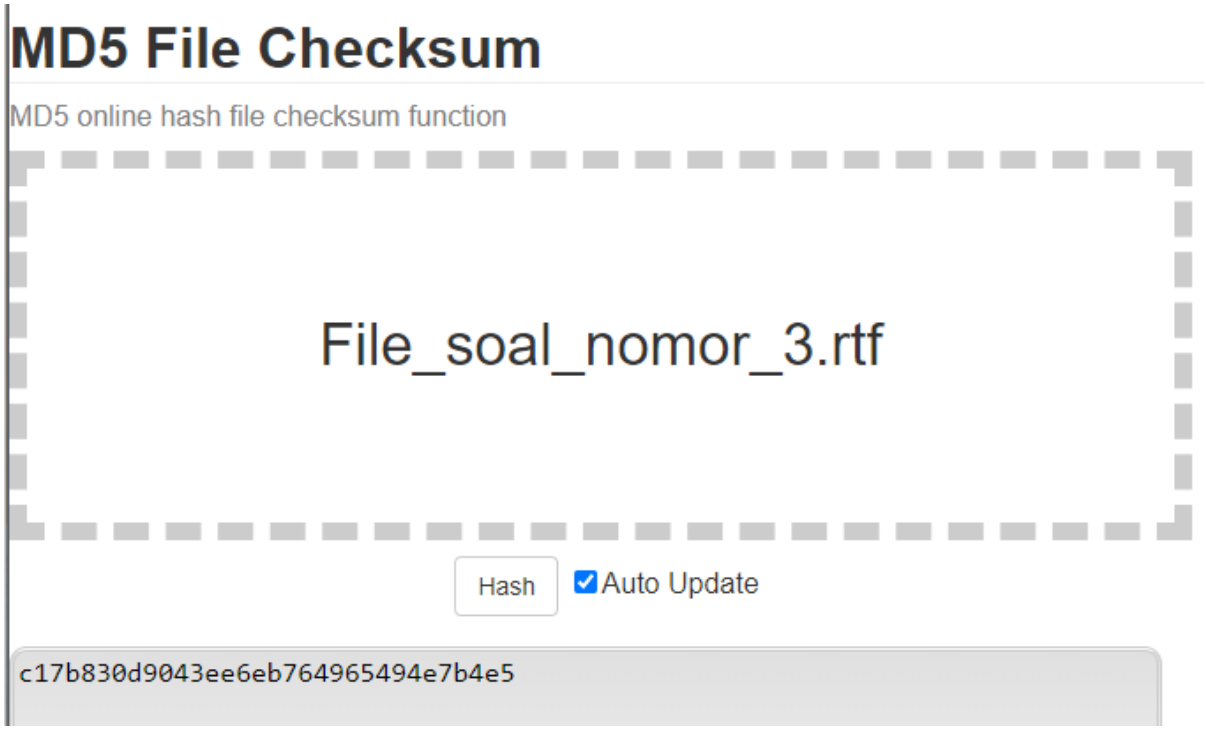
   Iterasi setiap kolom dan row sebanyak 4 piksel terhadap setiap channel. Kita dekripsi menggunakan kunci yang didapatkan pada soal no.1

```
%% Decipher Gambar menggunakan AES
img = imread('cipherImage.png');
for channel=1:3
    fprintf('Processing Channel: %d\n', channel);
    for row=1:4:256
        for col=1:4:256
            state = img(row:row+3,col:col+3,channel);
            res = decryptAES(double(state), fKeyAES);
            img(row:row+3,col:col+3,channel) = res;
        end
    end
end
imshow(img);
imwrite(img, 'result.png');
```



3. Integrity, Authenticity, dan Non-repudiation

   Kita cek menggunakan online tool untuk mencari checksum dari sebuah file menggunakan algoritma hash MD5. Nilai checksum ini layaknya ‘fingerprint’ untuk file tersebut. Apabila terdapat perubahan pada file walaupun hanya sedikit, nilai akan berubah secara signifikan. Ini merupakan langkah yang kita gunakan untuk memastikan Integritas suatu file atau dalam kata lain memastikan keaslian suatu file tidak berubah saat sampai ditangan kita.



Dengan beberapa nilai yang diketahui, kita dapat langsung mengimplementasikan ini kedalam program untuk lebih lanjut melakukan cek validasi.

```
%% Integrity, Authenticity, and Non-repudiation File Check
% nilai hash yang diketahui
hash = 'c17b830d9043ee6eb764965494bb326a';
checksum = 'c17b830d9043ee6eb764965494e7b4e5';
% ubah ke bentuk decimal untuk bagian 7 hexa yang di enkripsi
encryptedHashPart = hash(end-6:end);
encryptedHashVal = hex2dec(encryptedHashPart);
% dekripsi menggunakan RSA
res = FastExponent(encryptedHashVal, 1223, 93184991);
% ubah hasil ke bentuk hexa
res = num2str(res,'%x');
% selipkan kembali nilai hash pada 7 digit terakhir
hash(end-6:end) = res;
% apabila hasil hash sama dengan checksum, kita yakin bahwa
% file itu berasal dari orang yang sesuai
fprintf('%s : %s\n', hash, checksum);
if (hash == checksum) fprintf('Valid\n'); end
```

Dalam kasus ini, file tidak di-enkripsi dalam pengirimannya sehingga tidak memberikan sebuah aspek keamanan ‘Confidentiality’ atau kerahasiaan. Walaupun begitu, kita dapat gunakan algoritma RSA ini untuk melengkapi sebuah aspek Integritity, Authenticity, dan Non-repudiation. Kita tahu bahwa saat dikirimnya file, disertai juga sebuah checksum MD5 yang telah dienkripsi pada 7 digit terakhir bilangan hexadecimal. Enkripsi ini dibantu dengan sebuah algoritma RSA. Berikut langkah yang perlu kita lakukan:

- Kita dekripsi bagian yang terenkripsi menggunakan pasangan kunci public key dengan modulo yang diberikan oleh pihak yang berwenang (Iwan) melalui FastExponent.
- Hasil FastExponent akan disesuaikan kedalam bentuk hexadecimal dan menyelipkannya kembali ke bagian yang sebelumnya ter-enkripsi.
- Sekarang kita punya nilai hash yang ter-dekripsi (original), bandingkan dengan nilai checksum file.

```
>> Source
c17b830d9043ee6eb764965494e7b4e5 : c17b830d9043ee6eb764965494e7b4e5
Valid
```

Kita lihat bahwa hasilnya dekripsi Hash yang diberikan sudah sesuai dengan checksum file. Algoritma RSA ini membuat suatu enkripsi yang dilakukan oleh kunci private hanya dapat di dekripsi menggunakan pasangan kunci publiknya. Karena file tersebut telah ‘ditanda-tangani’ melalui nilai hash checksum dengan menggunakan kunci private, satu-satunya cara memvalidasi file tersebut adalah melalui kunci publik

Dengan kasus ini kita dapat yakin bahwa file berasal dari orang berwenang (Iwan) karena hanya dia seorang yang mengetahui pasangan kunci private-nya. Ini memberikan aspek keamanan Authenticity dimana kita bisa mengetahui kebenaran asal file. Selain itu memberikan aspek keamanan Non-repudiation dimana kedua belah pihak tidak bisa membantah bahwa terjadi sebuah pengiriman file dari orang yang dimaksud. Tentunya dengan menanda-tangani nilai hash checksum melalui RSA menambah aspek keamanan Integrity dimana kita bisa yakin tidak ada data yang berubah.