**Sains dan Komputer (SAINTIKOM)**
Vol.18, No.1, Februari 2019, pp. 41~46
P-ISSN: 1978-6603 E-ISSN : 2615-3475 

# Penggunaan Chinese Reminder Theorem (CRT) pada Algoritma RSA

**Zaimah Panjaitan, Khairi Ibnutama, M.Gilang Suryanata**
STMIK Triguna Dharma

**Article Info**

***Article history:***
Received Jan 22th, 2018
Revised Jan 28th, 2018
Accepted Feb 07th, 2019

***Keyword:***
Encryption
Decryptio,
ChineseReminder Theorem
RSA Algorithm.

**ABSTRACT**

Chinese Reminder Theorem (CRT) merupakan suatu teori matematis yang berfungsi untuk membantu menyederhanakan eksponensiasi modular yang berukuran besar. Algoritma RSA merupakan algoritma asimetris yang banyak digunakan untuk keamanan data. Algoritma RSA memiliki kekurangan yaitu sulitnya melakukan proses dekripsi pada algoritma ini yang disebabkan nilai eksponensiasi pada proses dekripsi relatif sangat besar. Karena adanya kekuarangan ini, dilakukan penggunaan CRT saat dekripsi pesan dengan algoritma RSA untuk membantu menyederhanakan nilai eksponen yang berukuran besar. Uji program menunjukkan bahwa tanpa penggunaan CRT, dekripsi pesan pada algoritma RSA tidak dapat dilakukan. Program secara otomatis memotong karakter chiperteks sehingga hasil dekripsi tidak sama dengan pesan aslinya. Sebaliknya dengan CRT, nilai cipherteks yang besar dapat dikembalikan sesuai pesan aslinya.



## 1. PENDAHULUAN

CRT (Chinese Remainder Theorem) adalah suatu teori matematis yang digunakan untuk mengkonversi eksponensiasi modular yang berukuran besar menjadi eksponensiasi modular yang relatif lebih kecil. Teori ini kerap digunakan sebagai salah satu solusi dari permasalahan yang ditemukan saat operasi sebuah algoritma yang membutuhkan penyederhanaan eksponensiasi modular seperti pada algoritma RSA.

Algoritma RSA (Rivest Shamir Adleman) adalah suatu algortima kriptografi asimetris yang ditemukan oleh Rivest, A. Shamir, dan L. Adleman pada tahun 1997[1]. Algoritma ini menggunakan kunci berbeda untuk proses enkripsi dan dekripsinya. RSA didasarkan pada prinsip beberapa operasi matematika yang melakukan operasi dalam satu arah, tetapi inversnya lebih sulit tanpa bantuan teori lain[2]. Seperti algoritma kriptografi lainnya, algoritma RSA juga tidak luput dari kekurangan. Meskipun RSA dapat digunakan untuk mengenkripsi dan mendekripsi pesan, sangat lambat jika pesan tersebut panjang [3].

Karena terdapat kekurangan pada algoritma RSA, diperlukan suatu teori yang mampu membantu agar kinerjanya lebih baik. Kesulitan yang terjadi pada RSA yakni pada saat dekripsi pesan disebabkan eksponensiasi modular yang umumnya sangat besar. Oleh karena itu pada penelitian ini dipelajari penggunaan teori CRT pada saat dekripsi pesan dengan algoritma RSA.

Adapun tujuan dari penelitian ini adalah mempelajari penggunaan teori CRT pada saat dekrispi pesan dengan algoritma RSA.

## 2. LANDASAN TEORI

### 2.1 Chinese Remainder Theorem

CRT dapat mengkonversi bilangan yang besar dari kunci dengan panjang eksponensiasi modular berukuran besar menjadi kunci yang lebih pendek dengan eksponensiasi modular berukuran yang relatif lebih kecil [4]

CRT misalkan m = m1, m2....mn, dan setiap pasang mi,mj comprime (bilangan bulat positif sedemikian hingga PBB(mi,mj) = 1 untuk i≠j, maka sistem kongruen lanjar, $X = \left(\sum_{i-1}^{t} \left(\frac{n}{d_i}\right) y_i x_i\right) mod\ n.$

Di mana $y_i$ adalah suatu solusi $\left(\frac{n}{d_i}\right) y_i\ mod\ d_i = 1.$

Dengan catatan,

$$x \bmod d_i = \left( \left( \sum_{j=1}^{t} \left( \frac{n}{d_j} \right) y_i x_j \right) mod\ n \right) mod\ d_i = x_i.$$

.

## 2.2 Algoritma RSA

Algoritma RSA merupakan salah satu algoritma kunci asimetris. RSA (dari Rivest-Shamir-Adleman) adalah sebuah kriptografi kunci publik yang berdasarkan pada eksponensial terbatas pada modulo bilangan bulat N(ZN) di mana N adalah sebuah bilangan bulat gabungan dari dua faktor besar (yaitu semi-prime)[5].

Adapun tahapan dalam algoritma RSA adalah sebagai berikut :

a. Pembangkitan kunci. Tahapan ini memiliki beberapa langkah sebagai berikut :
   1. Memilih dua pilangan prima *p* dan *q*
   2. Menghitung $n = p \times q$
   3. Menghitung nilai $m = (p-1)(q-1)$
   4. Memilih *e* yang relatif prima terhadap *m.* Gcd(*e*,*m*) = 1
   5. Menghitung nilai *d*
      *e* x *d mod m* = 1
   6. Menentukan nilai kunci *public =(e,n)* dan kunci *private = (d,n)*
b. Enkripsi. Tahapan ini mengubah *plaintext*(M) menjadi *ciphertext* (C) menggunakan kunci *public* dengan rumus : $C=M^e \pmod n$
c. Dekripsi. Tahapan ini mengembalikan *ciphertext* (C) menjadi *plaintext*(M) menggunakan kunci *private* dengan rumus : $M =C^d \pmod n$

## 3. PEMBAHASAN DAN HASIL

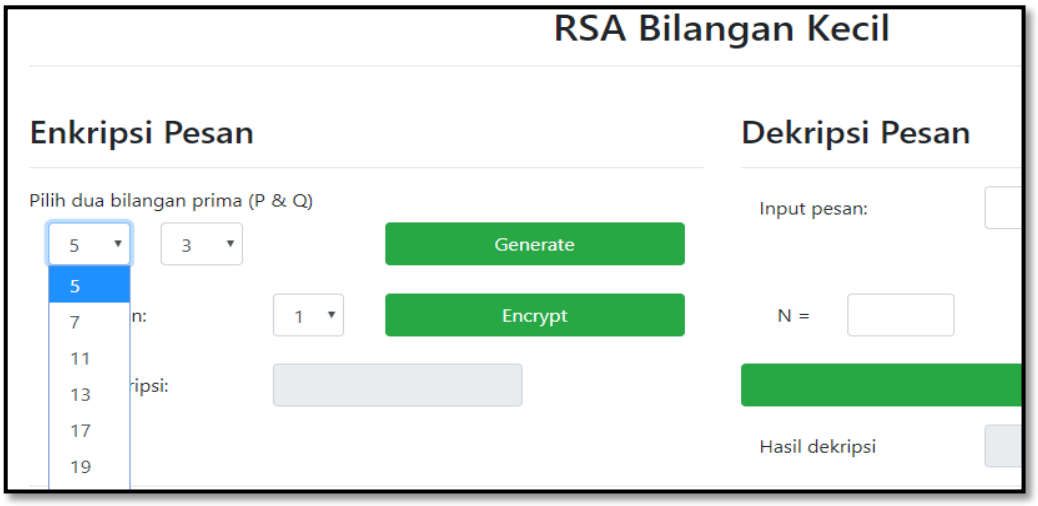
Pada dasarnya, proses enkripsi pesan dengan RSA tidak membutuhkan proses perhitungan dengan teori seperti CRT. Hal ini dikarenakan nilai pemangkatan eksponensial modulo yang diproses tidaklah besar. CRT hanya diperlukan untuk menurunkan nilai pemangkatan yang besar pada saat dekripsi pesan. Oleh karena itu, proses enkripsi disini dilakukan tanpa teori CRT. Adapun tahapan pada Algoritma RSA adalah sebagai berikut :

1. Pembangkit kunci RSA
   - Ambil nilai prima acak *p* =137 dan *q* =131
   - Hitung nilai $n = p \times q$, $n$=137 x 131 = 17947
   - Hitung nilai $m$=($p$-1)x($q$-1)
     $m$ =136 x 130= 17680
   - Memilih *e* yang relatif prima terhadap *m* dengan rumus euclidean gcd(*e*,*m*)=1. Didapat hasil *e* = 3
   - Menentukan nilai *d* dengan rumus :
     *e x d mod m* = 1, 3 x 11787 mod 17680 = 1
     maka nilai *d* = 11787
   - Menentukan nilai dP $=e^{-1} \pmod{(p-1)}$
     $=d \pmod{(p-1)}$, dP=91
   - Menentukan nilai dQ $=e^{-1} \pmod{(q-1)}$
     $=d \pmod{(q-1)}$, dQ=87
   - Menentukan nilai qInv $=q^{-1} \pmod p$
     qInv = 114
   - Menentukan kunci *public* dan kunci *private:*
     $K_{Public}$ = (*e*,*n*)=(3, 17947)
     $K_{Private}$ =(*dP*,*dQ*,*qInv*,*p*,*q*)
     =(91,87,114,137,131)
2. Enkripsi
   Contoh pesan yang dienkripsi adalah *M* = 1562
   $C = M^e \bmod n$, $C=1562^3 \bmod 17947 = 8825$
3. Dekripsi pesan dengan CRT
   Untuk mendekripsikan *ciphertext* di atas rumus dekripsi adalah $M = C^d \pmod{17947}$,

$M=8825^{11787}$ *mod* 17947. Namun jika kita perhatikan nilai eksponensiasi moduli terlalu besar, sangat sulit mencari nilai pemangkatan C, oleh karena itu disini dibutuhkan CRT. Jadi, proses dekripsinya sebagai berikut :

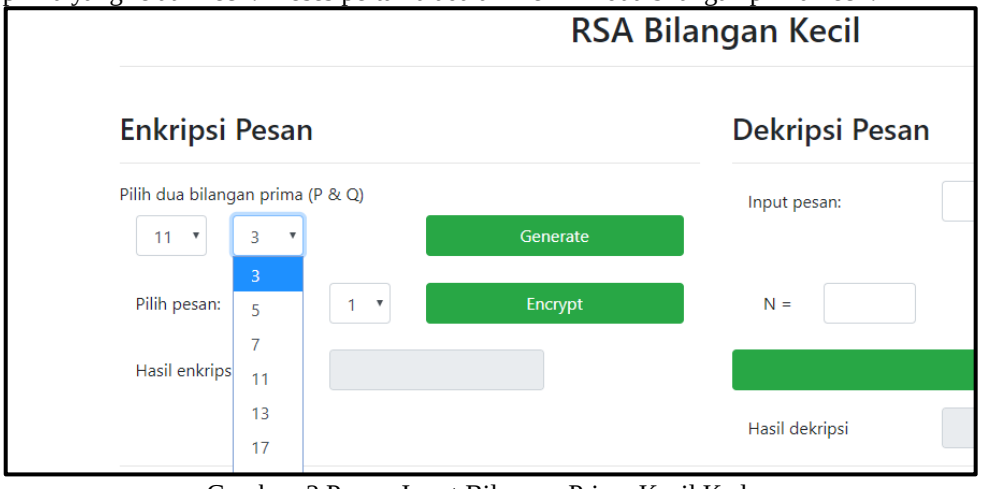
- Menentukan nilai $m_1$
  $m_1 = c^{dP} \ mod \ p$, $m_1 = 8825^{91} \ mod \ 137$
  untuk memangkatkan nilai besar ini dipakai CRT dengan membagi-bagi nilai pemangkatanya. Maka, $m_1 = 55$
- Menentukan nilai $m_2$
  $m_2 = c^{dQ} \ mod \ q$, $m_2 = 8825^{87} \ mod \ 131$
  untuk memangkatkan nilai besar ini dipakai CRT dengan membagi-bagi nilai pemangkatanya. Maka, $m_2 = 121$
- Menentukan nilai $h = qInv(m_1\text{-}m_2) \ mod \ p$
  $h = 114 \times 71 \ mod \ 137 = 11$
- Menghitung hasil dekripsi
  $M = m_2 + h.q$,
  $M = 121 + 11.\ 131 = 1562$

Telah dibahas bahwa dekripsi pesan dengan RSA membutuhkan CRT agar dapat berjalan dengan semestinya. Oleh karena itu, tanpa CRT dekripsi RSA tidak dapat berjalan sebagaimana mestinya seperti program pengujian berikut.




Gambar .1 Proses Input Bilangan Prima Kecil Pertama

Pertama sekali disediakan proses enkripsi RSA pada bilangan kecil. Dengan pesan masukan dan bilangan prima yang relatif kecil. Proses pertama adalah memilih dua bilangan prima kecil.

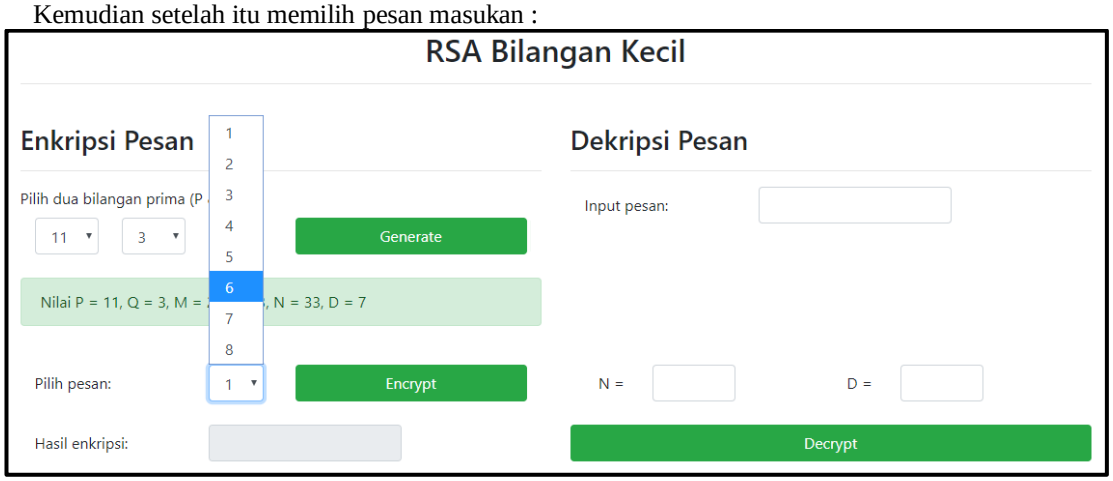



Gambar .2 Proses Input Bilangan Prima Kecil Kedua

Kemudian dilakukan *generate* kunci :

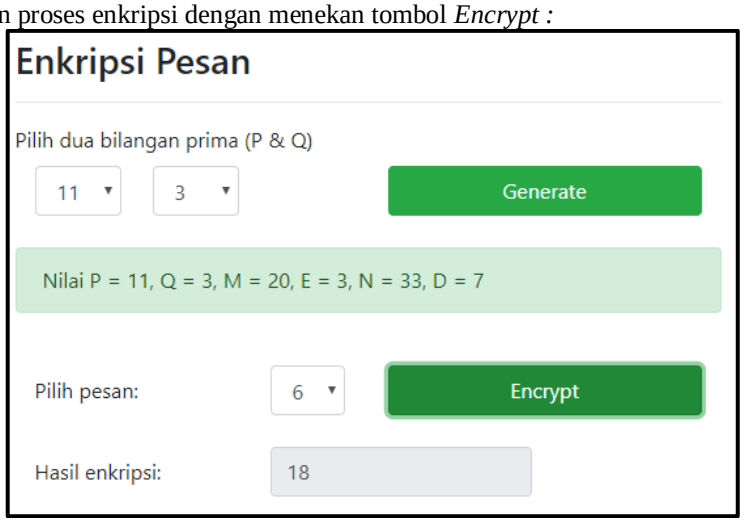

Gambar .3 *Generate* Kunci

Kemudian setelah itu memilih pesan masukan :




Gambar .4 Proses Memilih Pesan Masukan

Setelah itu dilakukan proses enkripsi dengan menekan tombol *Encrypt :*

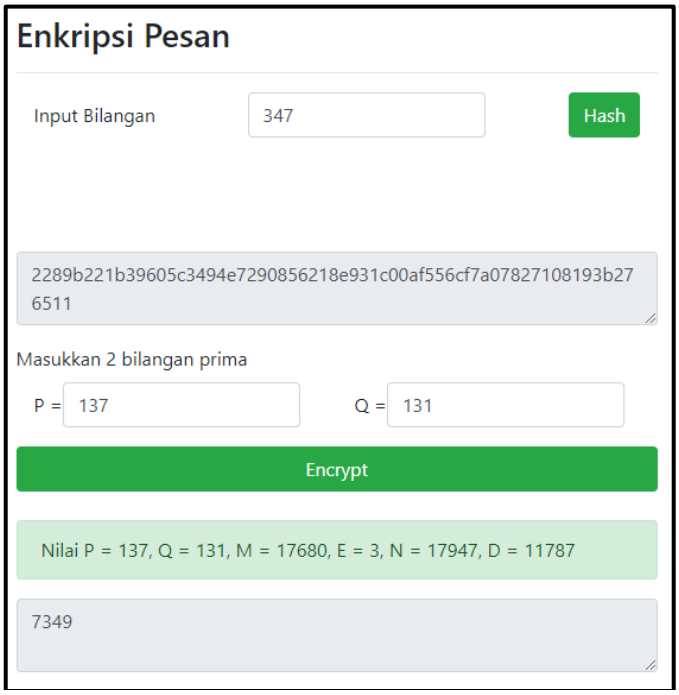



Gambar .5 Proses Enkripsi

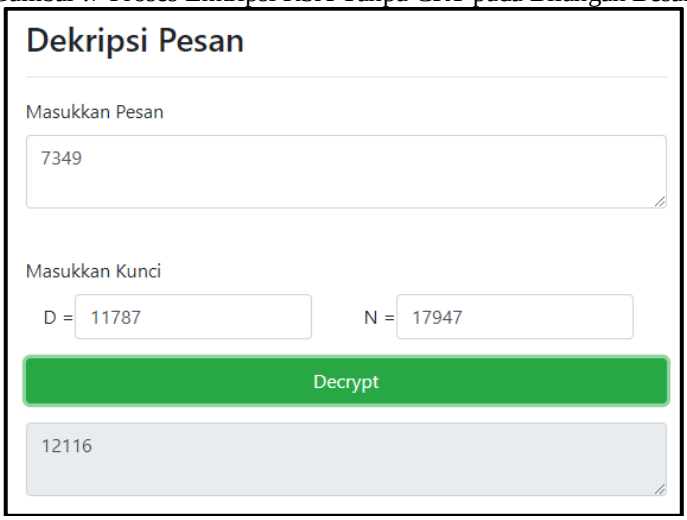
Setelah pesan dienkripsi maka untuk melihat proses dekripsinya perlu dimasukkan pesan masukan *ciphertext* kemudian kunci *private* berupa nilai D dan N dan hasil dekripsi setelah tombol *decrypt* ditekan adalah sebagai berikut :


Gambar .6 Proses Dekripsi

Terlihat bahwa pesan berhasil didekripsikan. Namun, proses tanpa CRT ini hanya dapat berlaku pada pesan dan prima pembangkit yang kecil. Jika diaplikasikan pada bilangan yang besar seperti *hash code*, maka hasilnya adalah sebagai berikut :

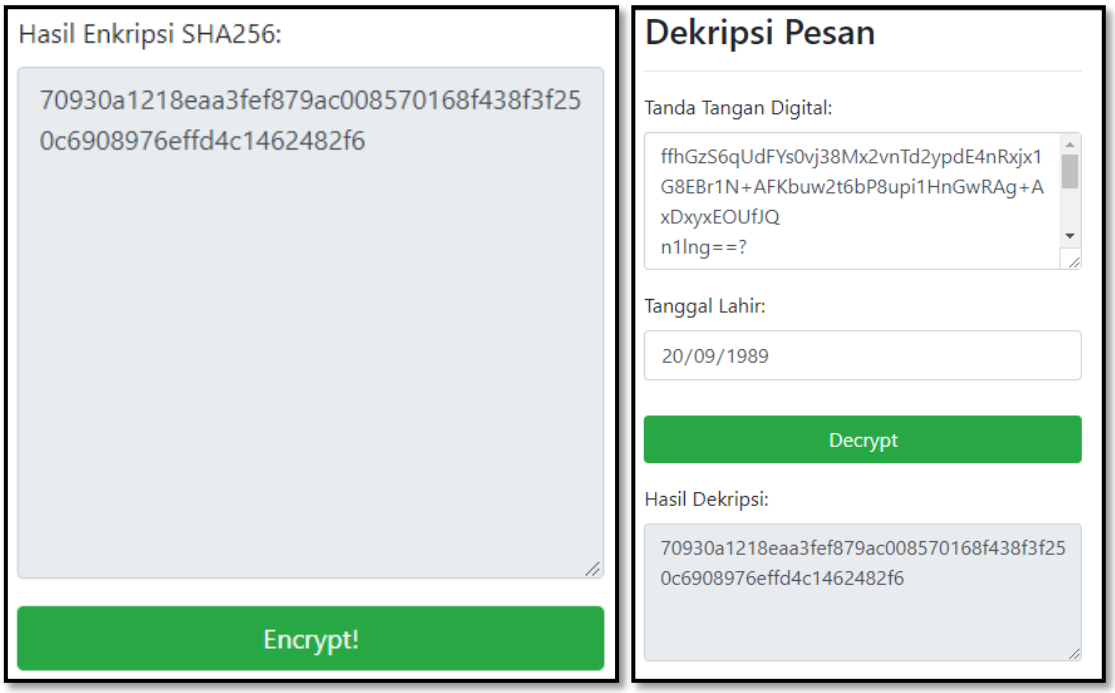



Gambar .7 Proses Enkripsi RSA Tanpa CRT pada Bilangan Besar




Gambar .8 Proses Dekripsi RSA Tanpa CRT pada Bilangan Besar

Dari hasil di atas terlihat bahwa pada saat bilangan yang dienkripsi besar dan prima juga besar, maka secara otomatis program akan memotong karakter sehingga pada saat didekripsikan, maka hasilnya sudah tidak sama dengan *ciphertext*. Hasil dekripsi merupakan pesan asli beruapa Hash code dengan panjang 128 bit. CRT terbukti mampu membantu dekripsi pesan sepanjang 128 bit.




Gambar .9 Proses Dekripsi RSA dengan CRT pada Bilangan Besar

## 4. KESIMPULAN

Berdasarkan penelitian di atas, dapat diambil kesimpulan sebagai berikut :
Proses enkripsi pesan dengan RSA tidak memerlukan CRT. CRT dipakai pada saat dekripsi karena nilai eksponensiasi modula yang besar terjadi pada saat dekripsi pesan., Dekripsi pesan dengan algoritma RSA memerlukan CRT. Tanpa CRT dekripsi pesan pada algoritma RSA tidak dapat dilakukan.

## REFERENSI

[1] Sadikin, M. A. dan Wardhani, R. W., (2016). 'Implementation Of RSA 2048-bit and AES 256-bit with Digital Signature for Secure Electronic Health Record Application'. 2016 International Seminar on Intelligent Technology and It's Application. IEEE. Hlm. 387-392. https://doi.org/ 10.1109/ISITIA.2016.7828691. ISBN : 9781509017096.

[2] Jaju, S. A. dan Chowhan, S. S., (2015). 'A Modified RSA Algorithm to Enhance Security for Digital Signature'. 2015 International Conference and Workshop on Computing and Communication. IEEE. https://doi.org/ 10.1109/IEMCON.2015.7344493. ISBN : 978-1-4799-6908-1.

[3] Arief, A. dan Saputra, R., (2016). 'Implementasi Kriptografi Kunci Publik dengan Algoritma RSA-CRT pada Aplikasi Instant Messaging'. Scientific Journal of Informatics. IEEE. Vol:3. Hlm: 46-54.

[4] Xiao, Z., et al, (2015). 'Research and Implementation of Four-prime RSA Digital Signature Algorithm'. 14th International Conference on Computer and Information Science. ISBN : 978-1-4799-8679-8. IEEE. Hlm : 545-549. https://doi.org/10.1109/ICIS.2015.7166652.

[5] Kiviharju, M., (2017). 'On the Fog of RSA Key Lengths Verifying Public Key Cryptography Strength Recommendations'. 2017 International Conference On Military Communication and Information System. ISBN : 978-1-5386-3858-3. IEEE. Hlm: 268-273.