# Penerepan Teori Bilangan pada Enkripsi RSA untuk Manipulasi Bit Pesan Text dan Gambar

*Pratama Nugraha Damanik 13513001*
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
*tamadamanik@students.itb.ac.id*

***Abstrak* — Makalah ini membahas tentang pengaplikasian teori bilangan pada salah satu metode enkripsi pesan, yaitu Enkripsi RSA untuk memanipulasi pesan text dan pesan gambar. Saat ini, hampir semua penyampaian pesan dilakukan melalui jaringan jaringan nirkabel, termasuk pesan yang berupa gambar, sehingga untuk menjaga keamanan pesan, diperlukan suatu metode penyamaran pesan. Penyamaran pesan ini bertujuan agar pesan yang disampaikan dari satu pihak ke pihak lain hanya dapat diketahu kedua pihak yang bersangkutan. Apabila pesan tersebut sampai kepada pihak yang bukan merupakan pihak penerima, maka pesan tersebut tidak akan mempunyai makna. Pada proses ini, kunci umum (kunci untuk mengenkripsi pesan) dapat diketahui semua orang, tetapi kunci privat (kunci untke mengdekripsi pesan) hanya diketahui oleh pihak yang berwenang untuk membuka pesan tersebut.**

***Kata Kunci*—,RSA , Enkripsi, Dekripsi, Key**

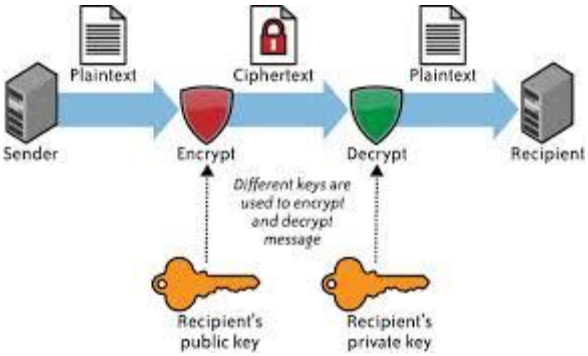
## I. PENDAHULUAN

Sekarang ini, jaringan sudah menjadi hal yang tak terpisahkan dengan komunikasi. Sebagai media komunikasi, jaringan sangat rentan terhadap penyerengan seperti penyadapan, pencurian, atau manipulasi informasi. Oleh karena itu, proses pengiriman data / pesan melalui suatu jaringan harus dijamin keamanan, keutuhan pesan tersebut untuk menghindari kemungkinan buruk yang bisa saja terjadi.

Pesan gambar merupakan salah satu bentuk pesan yang banyak dikirim melalui jaringan. Pesan gambar ini bisa berupa data penting, seperti foto tanda tangan, foto data rahasia, ataupun gambar yang seharusnya tidak diketahui oleh umum. Oleh karena itu diperlukan suatu metode penyandian pesan agar pesan tersebut tersamarkan sehingga hanya orang yang memang berhak untuk melihatnya yang bisa melihat gambar tersebut,

Teknik penyandian pesan atau yang biasa disebut kriptografi sebenarnya sudah dikenal sejak lama bahkan sejak jama Julius Caesar. Pada saat itu penyandian pesan dilakukan dengan cara menggeser huruf sebanyak beberapa karakter tergantung si penyandi pesan. Dan agar pesan tersebut dapat terbaca karakter tiap pesan digeser mundur sebanyak saat penyandian pesan.

Dalam penyandian pesan dikenal istilah enkripsi dan dekripsi. Enkripsi adalah suatu proses mengubah sebuah teks pesan murni (*plain text)* menjadi sebuah pesan yang tidak mempunya arti atau menyamarkan arti pesan sebenarnya. Hasil enkripsi ini disebut (*chipertext).* Lalu untuk mengubah pesan yang telah di enkripsi tadi dilakukan proses dekripsi. Proses dekripsi ini mengubah chipertext tadi menjadi plaintext sehingga pesan tersebut dapat dibaca.




**Proses enkripsi dan dekripsi RSA**
*Sumber : http://iocvo.wordpress.com*

Seiring dengan semakin umumnya penggunaan jaringan sebagai media transfer pesan, maka banyak tercipta algoritma penyandian pesan. Beberapa contoh algoritma penyandian adalah, MD2 ( untuk GSM ), IDEA ( *International Data Encryption Algorithm )*. RC2, ECC, DSA, RSA, A5, dll . Saat ini, yang umum dipakai adalah DES dan RSA.

Algoritma RSA sendiri dibuat pada tahun 1978 oleh 3 peneliti dari MIT yaitu Ron Rivest, Adi Shamir dan Leonard Adleman. RSA memiliki banyak kelebihan salah satunya sulit untuk dibobol. Satu satunya cara untuk membobol RSA adalah mencoba segala kemungkinan kombinasi kunci enkripsi dan dekripsinya. Namun kelemahan dari RSA ini adalah, proses dekripsinya memakan waktu yang cukup lama dibandingkan dengan metode lain.

**Penemu Algoritma RSA**
*Sumber : www.google.com*

## II. DASAR TEORI

2.1. Algoritma Euclid

Algoritma Euclid adalah algoritma yang digunakan untuk mencari bilangan pembagi bersama terbesar dari 2 bilangan. Algoritma ini dilaksanakan secara bertahap, dimana hasil yang didapat dari tahap sebelumnya akan digunakan di tahap berikutnya. Misalmya ada 2 bilangan a dan b, anggap a dibagi b bersisa $r_1$ . Jika $r_1$ tidak sama dengan nol, maka langkah selanjutnya adalah mencari sisa dari b dibagi $r_1$ (anggap b < a). Misalnya sisanya $r_2$ , jika $r_2$ tidak sama dengan nol maka dicari lagi sisa $r_1$ bagi $r_2$ dan bersisa $r_3$. Jika $r_3$ tidak sama dengan nol, dicari lagi sisa $r_2$ bagi $r_3$ dan selanjutnya hingga ditemukan sisanya sama dengan nol. Misalnya sisa dari $r_{n-1}$ dibagi $r_n$ adalah nol, maka FPB dua bilangan a dan b tadi adalah $r_n$ . Secara prosedur, algoritma Euclid dapat dituliskan seperti gambar dibawah :

$$
\begin{aligned}
&a = q_0 b + r_1 \\
&b = q_1 r_1 + r_2 \\
&r_1 = q_2 r_2 + r_3 \\
&r_2 = q_3 r_3 + r_4 \\
&\ldots \\
&r_{n-1} = q_n r_n + 0 \\
&\text{dengan } q \text{ sebagai } quotient \text{ (hasil bagi)} \\
&\text{Jika } b \text{ kita anggap sebagai } r_0 \text{ (sisa ke 0)} \\
&\text{maka persamaan akan menjadi} \\
&a = q_0 r_0 + r_1 \\
&r_0 = q_1 r_1 + r_2 \\
&r_1 = q_2 r_2 + r_3 \\
&r_2 = q_3 r_3 + r_4 \\
&\ldots \\
&r_{n-1} = q_n r_n + 0
\end{aligned}
$$

**Algoritma Euclid secara procedural**
*Sumber : http://hallofnotes.blogspot.com*

2.2. Relatif Prima

Dua buah bilangan a dan b dikatakan relative prima jika FPB dari kedua bilangan adalah 1. Untuk mencari FPB dari a dan b dapat dicari dengan algoritma Euclid. Apabila sisa pembagi sebelum 0 adalah 1, maka kedua bilangan tersebut relative prima.

2.3. Modulo, Kongruen dan Modulo Invers

Misal a dan b bilangan bulat (b > 0) dimana a dibagi b menyisakan r, *b* disebut modulo. Ditulis dengan "a mod b = r" . Contohnya adalah "5 mod 3 = 2" .
Kongruen adalah jika ada bilangan a dan b, jika kedua bilangan tersebut di-modulo-kan dengan suatu bilangan c (c >0), maka akan menyisakan sisa yang sama. Ditulis $a \equiv b \ (mod\ c)$ . Contohnya , 23 mod 3 = 2 dan 5 mod 3 = 2, dapat ditulis $23 \equiv 5 \pmod 3$.
Jika 2 bilangan a dan b relative prima dan b > 1, maka balikan (*invers*) dari a (mod b) ada . Balikan dari a (mod b) adalah bilangan bulat x, sehingga :

$xa \equiv 1 \pmod b$ *atau* $a^{-1} \pmod m = x$

2.4. Teorema Fermat

Jika p adalah bilangan prima dan a merupakan bilangan bulat yang tidak habis dibagi p, yaitu PBB(a,p) = 1, maka

$a^{p-1} \equiv 1 \pmod p$

## III. Enkripsi RSA untuk Manipulasi Bit Pesan Text dan Gambar

3.1. Proses Enkripsi dan Dekripsi RSA

Dalam proses penyandian pesan dengan RSA, pertama kali yang harus dibuat adalah kunci public dan kunci privat. Caranya adalah sebagai berikut :

1. Pilih dua bilangan prima p dan q , dengan p tidak sama dengan q . Untuk mempertinggi keamanan bilangan p dan q harus cukup besar (minimal 100 digit).
2. Hitung N = pq . Bilangan N disebut parameter sekuriti.
3. Hitung m = (p-1)(q-1)
4. Pilih satu bilangan bulat yang akan digunajan sebagai kunci public *e* . e berada diantara 1 dan m dan relative prima terhadap m. Langkah ini bisa menggunakan algoritma euclidan
5. Lalu cari kunci privat *d* sehingga : $de \equiv 1 \pmod m$

Bilagan p dan q adalah bilangan yang rahasia, karena terdapat N dimana N merupakan hasil kali p dan q. Bentuk ini memperbolehkan deskripsi secara tepat dan signing menggunakan Chinese Reminder Theorem (CRT), tetapi ini membuat menjadi lebih tidak aman, karena bentuk ini memperbolehkan *slide channel attacks. Slide channel attacks* adalah sebuah serangan yang berdasarkan informasi yang dikumpulkan dari implementasi fisik (kelemahan secara fisik) dari sebuah system kriptografi, disbanding dengan

kelemahan teoritis dari algoritmanya sendiri. Contohnya, factor factor kurun waktu dari informasi, konsumsi tenaga, bahkan suara yang ditimbulkan dapat membantu mempermudah informasi yang bisa didapat untuk menjebol system tersebut.

Setelah didapat kunci pubilk dan kunci privat, lalu dilakukan enkripsi pesan. Enkripsi pesan dapat dilakukan dengan memecah pesan menjadi blok yang lebih kecil sehingga dihasilkan sebuah angka *n* . Lalu untuk mengubah angka tersebut menjadi chipper text, dapat dibuat menjadi :

$$c = n^e \mod N$$

Dimana c adalah chippertext yang dihasilkan, n adalah integer, e adalah kunci public, dan N adalah hasil kali p dan q sebelumnya. Jadi pesan yang dikirim adalah pesan yang telah diubah dalam bentuk chipertext.

Setelah didapat pesan yang berupa chipertext, si penerima pesan dapat mengubah pesan tersebut menjadi bentuk aslinya. Cara mengubah pesan adalah sebagai berikut :

$$n = c^d \mod N$$

dimana n adalah teks pesan asli, c adalah chipertext, d adalah kunci privat dan N adalah hasil kali p dan q .

3.2. Enkripsi RSA pada Bit Pesan Text

Metode ini bekerja dengan cara memanipulasi bit tiap karakter pada pesan text. Pertama setiap karakter tiap pesan teks diubah ke bentuk heksadesimal. Setelah didapatkan bentuk heksadesimal, heksadesimal tersebut diubah ke bentuk biner. Sehingga didapatkan Biner 2 Byte .

Lalu seluruh plaintext biner akan dibagi menjadi sejumlah *y* blok dengan tiap blok mengandung *x* bit . Misalnya sebuah plaintext biner mengandung 63, maka dapat dibuat menjadi 9 blok dengan tiap bloknya menjadi 7 bit.

Lalu nilai biner tiap blok tadi di representasikan menjadi integer, missal *i*. Nilai *i* inilah yang akan di enkrispsi. Misalnya setelah dienkripsi menghasilkan *j* .Lalu *j* akan direpresentasikan menjadi bentuk biner kembali. Setelah itu, seluruh biner tiap blok digabung kembali dan dipecah menjadi 8 bit. Setelah itu tiap 8 bit direpresentasikan menjadi nilai heksadesimal.

Setelah itu nilai heksadesimal tersebut direpresentasikan menjadi karakter. Untuk proses dekripsi menggunakan system pemecahan yang sama dengan pemecahan saat enkripsi. Berikut contoh manipulasi bit pesan text dengan RSA :

1. Misalkan dipilih p = 7 dan q = 11 . Sehingga didapat n = 77 dan m = 60 .
2. Cari public key yang relative prima terhadap 60. Didapat yaitu 17.
3. Lalu cari kunci privat yang membuat de mod m = 1. Didapat d = 53 . Bukti : 17*53 = 901, 901 mod 60 = 1.
4. Misalkan pesan yang dikirim “GUNUNG”
5. Setiap karakter dalam ‘GUNUNG’ diubah menjadi heksadesimal dan biner, yaitu :

| | | |
|---|---|---|
| G | 47 | 01000111 |
| U | 55 | 01010101 |
| N | 4E | 01001110 |
| U | 55 | 01010101 |
| N | 4E | 01001110 |
| G | 47 | 01000111 |

Didapat biner text :
010001 110101 010101 001110 010101 010100 111001 000111

Lalu biner text dibagi menjadi 8 blok dengan 6 bit per blok :

| Block | Decimal | Biner |
|---|---|---|
| 1 | 17 | 010001 |
| 2 | 53 | 110101 |
| 3 | 21 | 010101 |
| 4 | 12 | 001110 |
| 5 | 21 | 010101 |
| 6 | 20 | 010100 |
| 7 | 57 | 111001 |
| 8 | 7 | 000111 |

Lalu tiap bilangan decimal tersebut dienkripsi dengan menggunakan enkripsi RSA :

1. $C1 = 17^{17} \mod 77 = 19$
2. $C2 = 53^{17} \mod 77 = 37$
3. $C3 = 21^{17} \mod 77 = 21$
4. $C4 = 12^{17} \mod 77 = 45$
5. $C5 = 21^{17} \mod 77 = 21$
6. $C5 = 20^{17} \mod 77 = 48$
7. $C5 = 57^{17} \mod 77 = 29$
8. $C5 = 7^{17} \mod 77 = 28$

| Block | Dec | Enk | Biner |
|---|---|---|---|
| 1 | 17 | 19 | 010011 |
| 2 | 53 | 37 | 100101 |
| 3 | 21 | 21 | 010101 |
| 4 | 12 | 45 | 001010 |
| 5 | 21 | 21 | 010101 |
| 6 | 20 | 48 | 110000 |
| 7 | 57 | 29 | 011101 |
| 8 | 7 | 28 | 011100 |

Digabung menjadi :
01001110 01010101 01001010 01010111 00000111 01011100

Setelah itu dipisah menjadi 8 bit tiap blok :

| Biner | Hex | Karakter |
|---|---|---|
| 01001110 | 43 | C |
| 01010101 | 55 | U |
| 01001010 | 4A | J |
| 01010111 | 47 | G |
| 00000111 | 07 | *BEL* |
| 01011100 | 5C | \ |
| | | |

Maka pesan menjadi :
CUJG*BEL*\

Lalu untuk mendekripsi pesan dilakukan cara yang sama :

CUJG*BEL*\ dibuat ke representasi biner dan dipecah dakm bentuk blok blok

| | | |
|---|---|---|
| C | 43 | 01001110 |
| U | 55 | 01010101 |
| J | 4A | 01001010 |
| G | 47 | 01010111 |
| *BEL* | 07 | 00000111 |
| \ | 5C | 01011100 |

Representasi biner :
01001110 01010101 01001010 01010111 00000111 01011100

Diubah ke bentuk blok blok menjadi 8 blok dengan tiap blok 6 bit :

| | | |
|---|---|---|
| 1 | 010011 | 19 |
| 2 | 100101 | 37 |
| 3 | 010101 | 21 |
| 4 | 001010 | 45 |
| 5 | 010101 | 21 |
| 6 | 110000 | 48 |
| 7 | 011101 | 29 |
| 8 | 011100 | 28 |

Lalu setiap decimal blok di dekripsi menggunakan kunci privat :

1. $19^{53}$ mod 77 = 17 - 010001
2. $37^{53}$ mod 77 = 53 - 110101
3. $21^{53}$ mod 77 = 21 - 010101
4. $45^{53}$ mod 77 = 12 - 001010
5. $21^{53}$ mod 77 = 21 - 010101
6. $48^{53}$ mod 77 = 20 - 010100
7. $29^{53}$ mod 77 = 57 - 111001
8. $28^{53}$ mod 77 = 7 - 000111

Jika digabung maka akan menjadi :

01000111 01010101 01001110 01010101 01001110 01000111

Jika tiap 8 bit dari biner text diatas diubah menjadi heksa decimal, maka akan menjadi :

01000111 47
01010101 55
01001110 4E
01010101 55
01001110 4E
01000111 47

Lalu langkah terakhir adalah mengubak heksadesimal di atas menjadi karakter , yaitu :

47 = G
55 = U
4E = N
55 = U
4E = N
47 = G

3.3. Enkripsi RSA pada Pesan Gambar

Pada pesan gambar, proses enkripsi dengan RSA hampir sama dengan sebelumnya. Hanya saja, pada gamabar, unsur yang dideskripsi berbeda.

Seperti yang diketahui, sebuah gambar pasti mempunyai ukuran pixel tertentu. Lalu di setiap pixel tersebut pasti ada wamengandung kode warna tersendiri. Pada enkripsi RSA untuk pesan gambar, yang di enkripsi adalah kode warna tersebut. Sebelum memulai enkripsi, langkah langkah yang dilakukan kurang lebih sama seperti sebelumnya, yaitu kta mencari kunci public dan kunci privat.

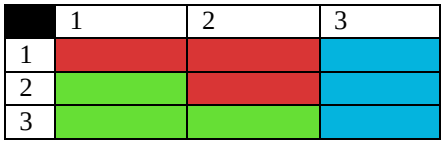
Setelah kunci public dan kunci privat didapat, proses enkripsi pun dimulai. Kode warna tiap pixel pada gambar di enkripsi dengan rumus enkripsi RSA. Setelah di enkripsi, kode warna untuk tiap pixel tersebut akan berubah menjadi 1 bilangan baru, misalnya *x*. Setelah itu, bilangan baru *x* tadi direpresentasikan dalam biner.

Pada saat proses enkripsi, nilai modulo N sangat mempengaruhi proses enkripsi, yaitu :

1. Nilai modulo harus lebih besar dari 264, karena jumlah kode warna dalam bitmap adlah 255, dan agar dapat direpresentasikan dalam 4 byte, maka jumlah modulo harus lebih besar dari 264

Jika modulo N lebih besar dari 264, pasti ada hasil modulo yang lebih besar dari 255. Untuk mengatasinya, semua hasil modulo direpresentasikan dalam bentuk 4 byte . Lalu 4 byte tersebut dipecah menjadi 2 blok dimana masing

masing blok adalah 2 byte. Setelah itu, bilangan decimal hasil representasi bilangan biner bilangan 2 byte adalah kode warna hasil enkripsi. Jadi untuk modulo N lebih besar dari 264, maka setiap 1 pixel pada gambar asli, jika dienkripsi akan menghasilkan 2 pixel. Jadi hasil gambar enkripsi akan memiliki lebar yang 2x dari lebar sebelumnya, tetapi memiliki tinggi yang sama. Setelah proses enkripsi selesai, maka akan didapat gambar hasil enkripsi yang lebarnya 2x gambar semula, Untuk proses dekripsi, setiap 2 pixel mewakili 1 pixel apda gambar asli. Dua pixel tersebut, kita dekripsi kode warnanya sehingga akan membentuk kode baru. Lalu kedua bilangan hasil enkripsi tadi di representasikan ke biner. Selanjutnya, kedua representasi biner tadi digabung menjadi 1 representasi biner dan dihitung nilai desimalnya. Hasil nilai decimal itulah kode warna gambar sebenarnya.




**Contoh pesan gambar dengan jumlah pixel 9**
Merah : 217,53,53
Hijau : 102,223,53
Biru : 4,180,222

Gambar di atas adalah contoh pesan gambar berukuran 3x3 Untuk mengenkripsi kita harus memilih kucni public,kunci privat, dan modulo N.

1. P dan q dipilih 61 dan 53, sehingga dihasilkan N= 3233
2. M didapat 3120
3. Kunci public relative prima dengan 3120 yaitu 17
4. Lalu kunci privat 2753

Setelah itu kode warna di tiap pixel kita ambil, sehingga :

| No | Posisi | Kode Warna |
|---|---|---|
| 1 | 1,1 | 217,53,53 |
| 2 | 1,2 | 217,53,53 |
| 3 | 1,3 | 4,180,222 |
| 4 | 2,1 | 102,223,53 |
| 5 | 2,2 | 217,53,53 |
| 6 | 2,3 | 4,180,222 |
| 7 | 3,1 | 102,223,53 |
| 8 | 3,2 | 102,223,53 |
| 9 | 3,3 | 4,180,222 |

**Tabel kode warna untuk gambar sebelumnya**

Lalu setiap kode warna dienkripsi menjadi :

1. $217^{17} \bmod 3233 = 2132$
2. $53^{17} \bmod 3233 = 1802$
3. $4^{17} \bmod 3233 = 1387$
4. $180^{17} \bmod 3233 = 2937$
5. $222^{17} \bmod 3233 = 248$
6. $102^{17} \bmod 3233 = 1752$
7. $223^{17} \bmod 3233 = 93$

Lalu setiap kode warna direpresentasikan dalam biner :
2132 : 00001000 01010100
1802 : 00000111 00001010
1387 : 00000101 01101011
2937 : 00001011 01111001
248 : 00000000 11111000
1752 : 00000110 11011000
93 : 00000000 01011101

Selanjutnya tiap representasi biner tersebut diubah ke bentuk decimal :

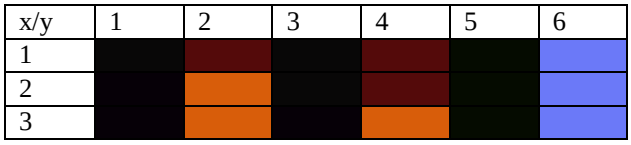
| | **Dec Enkripsi** | **Dec. Enkripsi 1** | **Dec. Enkripsi 2** |
|---|---|---|---|
| **217** | 2132 | 8 | 84 |
| **53** | 1802 | 7 | 10 |
| **4** | 1387 | 5 | 107 |
| **180** | 2937 | 11 | 121 |
| **222** | 248 | 0 | 248 |
| **102** | 1752 | 6 | 216 |
| **223** | 93 | 0 | 93 |

Sehingga, untuk menampung kode warna tersebut, ukuran gambar akan diperlebar menjadi 2x semula, sehingga kode warna untuk tiap pixel berubah menjadi :

| No | Posisi | Kode Warna |
|---|---|---|
| 1 | 1,1 | 8,7,7 |
| 2 | 1,2 | 84,10,10 |
| 3 | 1,3 | 8,7,7 |
| 4 | 1,4 | 84,10,10 |
| 5 | 1,5 | 5,11,0 |
| 6 | 1,6 | 107,121,248 |
| 7 | 2,1 | 6,0,7 |
| 8 | 2,2 | 216,93,10 |
| 9 | 2,3 | 8,7,7 |
| 10 | 2,4 | 84,10,10 |
| 11 | 2,5 | 5,11,0 |
| 12 | 2,6 | 107,121,248 |
| 13 | 3,1 | 6,0,7 |
| 14 | 3.2 | 216,93,10 |
| 15 | 3,3 | 6,0,7 |
| 16 | 3,4 | 216,93,10 |
| 17 | 3,5 | 5,11,0 |
| 18 | 3,6 | 107,121,248 |

**Tabel kode warna tiap pixel setelah dienkripsi**

Maka gambar akan berubah menjadi :




**Gambar setelah dienkripsi**

Dapat dilihat, setelah dienkripsi, bentuk gambar jauh berbeda dengan aslinya. Apabila jumlah pixel banyak dan warna lebih komplek, tentu bentuk gambar tidak akan

terlihat lagi.

Untuk mendekripsi gambar ulang, maka kode warna tiap pixel gambar diambil, menjadi :

| No | Posisi | Kode Warna |
|---|---|---|
| 1 | 1,1 | 8,7,7 |
| 2 | 1,2 | 84,10,10 |
| 3 | 1,3 | 8,7,7 |
| 4 | 1,4 | 84,10,10 |
| 5 | 1,5 | 5,11,0 |
| 6 | 1,6 | 107,121,248 |
| 7 | 2,1 | 6,0,7 |
| 8 | 2,2 | 216,93,10 |
| 9 | 2,3 | 8,7,7 |
| 10 | 2,4 | 84,10,10 |
| 11 | 2,5 | 5,11,0 |
| 12 | 2,6 | 107,121,248 |
| 13 | 3,1 | 6,0,7 |
| 14 | 3.2 | 216,93,10 |
| 15 | 3,3 | 6,0,7 |
| 16 | 3,4 | 216,93,10 |
| 17 | 3,5 | 5,11,0 |
| 18 | 3,6 | 107,121,248 |

**Kode warna tiap pixel untuk gambar terenkripsi**

Seperti yang diketahui, 2x1 pixel di gambar terenkripsi mewakili 1x1 di gambar aslinya. Sehingga representasi biner untuk 2 pixel digabung menjadi 1 representasi biner, sehingga :

1. 8 : 0000100 | 84 : 01010100
   Digabung menjadi : 0000100001010100 = 2132
2. 7 : 00000111 | 10 : 00001010
   Digabung menjadi : 0000011100001010 = 1802

…. . . .

Begitu seterusnya, hingga nanti didapat 7 buah bilangan baru, yaitu :

| | **Hasil Gabungan** | **Dec. Enkripsi 1** | **Dec. Enkripsi 2** |
|---|---|---|---|
| **1** | 2132 | 8 | 84 |
| **2** | 1802 | 7 | 10 |
| **3** | 1387 | 5 | 107 |
| **4** | 2937 | 11 | 121 |
| **5** | 248 | 0 | 248 |
| **6** | 1752 | 6 | 216 |
| **7** | 93 | 0 | 93 |

Selanjutnya bilangan hasil gabungan tersebut didekripsi dengan kunci privat, yaitu :

1. $2132^{2753} \bmod 3233 = 217$
2. $1802^{2753} \bmod 3233 = 53$
3. $1387^{2753} \bmod 3233 = 4$
4. $2937^{2753} \bmod 3233 = 180$
5. $248^{2753} \bmod 3233 = 222$
6. $1752^{2753} \bmod 3233 = 102$
7. $93^{2753} \bmod 3233 = 223$

Maka, kode warna tiap pixel berubah menjadi :

| No | Posisi | Kode Warna |
|---|---|---|
| 1 | 1,1 | 217,53,53 |
| 2 | 1,2 | 217,53,53 |
| 3 | 1,3 | 4,180,222 |
| 4 | 2,1 | 102,223,53 |
| 5 | 2,2 | 217,53,53 |
| 6 | 2,3 | 4,180,222 |
| 7 | 3,1 | 102,223,53 |
| 8 | 3,2 | 102,223,53 |
| 9 | 3,3 | 4,180,222 |

**Tabel kode warna gambar asli**

Dapat dilihat bahwa ukuran gambar kembali ke semula, ini karena tiap 2 pixel gambar enkripsi mewakili 1 pixel gambar asli. Misal di posisi 1,1 gambar enkripsi kode warnanya 8,7,7 dan di 1,2 kode warnanya 84,10,10 . Representasi biner 8 dan 84 digabung menjadi 2132. Setelah itu 2132 di dekripsi dengan kunci privat menjadi 217. Lalu untuk semua kode warna dilakukan cara yang sama sehingga didapat table diatas.

Setelah di dekripsi maka gambar akan menjadi :

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |

**Gambar Asli**

3.4. Kelemahan dan Kelebihan RSA untuk diterapkan pada manipulasi bit dan gambar

Kelebihan RSA :

- Problem dalam faktorisasi bilangannya menjadi 2 faktor prima sangat banyak. Dan untuk membuat enkripsi RSA dianjurkan p dan q lebih dari 100 digit, sehingga akan sangat susah mencari faktornya
- Pada manipulasi pesan bit, pesan bit menjadi sangat jauh berbeda dari pesan aslinya, sehingga apabila sampai ke pihak yang bukan penerima, keamanannya akan terjamin
- Pada manipulasi pesan gambar, gambar jauh berubah dari bentuk asli, dan apabila warna gambar lebih kompleks, maka gambar tidak akan terlihat lagi bentuk aslinya

Kelemahan RSA :

- Proses dekripsi RSA memakan waktu yang sangat lama dibanding system enkripsi lainnya
- Apabila nilai n terlalu kecil, akan mudah untuk memfaktorisasinya menjadi 2 buah bilangan prim. Sehingga kunci public dan kunci privat akan lebih mudah didapatkan
- Pada manipulasi bit, apabila hasil enkripsi adalah karakter yang tidak terlihat maka akan menjadi

masalah. Pada contoh diatas, ditemukan karakter *BEL*, sehingga untuk merepresentasikan karakter tidak terlihat, pengirim dan penerima pesan harus sepakat untuk menggunakan ciri tertentu

- Pada manipulasi gambar, hasil gambar menjadi lebih besar 2x dari sebelumnya. Akibatnya memori yang dimakan untuk menyimpan akan 2x lebih besar dan proses dekripsi menjadi lebih lama dari proses enkripsi

## IV. KESIMPULAN

Kesimpulan yang dapat diambil dari penggunaan RSA untuk manipulasi bit dan gambar adalah :

- RSA adalah metode penyandian yang sangat kuat untuk mengatasi masalah keamanan pengiriman data, baik berupa teks maupun gambar pada suatu jarungan komunikasi
- Porses enkripsi lebih mudah daripada proses dekripsi.
- Teknik pembobolan yang bisa digunakan pada RSA hanya brute force attack, yaitu memfaktorkan N menjadi 2 buah bilangan prima dan mencari kunci privat dari kunci public yang bersesuaian. Maka untuk membuat N, dianjurkan factor pengali p dan q sangat besar
- Pada manipulasi bit dan gambar, RSA sangat bermanfaat untuk menyamarkan bentuk asli pesan
- Kelemahan pada manipulasi bit adalah apabila bit yang dihasilkan menunjuk ke karakter yang tidak terlihat
- Kelemahan pada manipulasi gambar adalah gambar yang dihasilkan lebih besar

## V. DAFTAR PUSTAKA


1. Slide kuliah Matematika Diskrit IF2120 “Rinaldi M/IF2120 Matematika Diskrit – Teori Bilangan [2014]”
2. http://fototiptrik.blogspot.com/2012/05/apa-itu-pixel.html (tanggal akses 7 Desember 2014)
3. http://sikomku.wordpress.com/2011/03/26/konversi-bilangan-ascii-ke-bilangan-heksadesimal/ (tanggal akses 6 Desember 2014)
4. http://budi.insan.co.id/courses/ec5010/ (tanggal akses 6 Desember 2014)
5. http://journal.amikom.ac.id/index.php/KIDA/article/viewFile/4400/2108 (tanggal akses 6 Desember 2014)
6. http://mathsisgood.wordpress.com/2012/08/01/algoritma-euclid/ (tanggal akses 6 Desember 2014)
7. http://www.academia.edu/4473248/ (tanggal akses 6 Desember 2014)


## PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 8 Desember 2014

Pratama Nugraha Damanik – 13513001