BAB IV  
HASIL DAN PEMBAHASAN

Bab ini membahas tentang algoritma RSA dalam melakukan enkripsi dan dekripsi pada File Teks.

## Proses Enkripsi Dan Dekripsi File Teks

RSA melibatkan kunci publik dan kunci privat. Kunci publik boleh diketahui oleh semua orang sehingga kunci publik tersebut mempunyai sifat yang tidak rahasia. Maksudnya adalah pada proses enkripsi kunci publik e dan n yang merupakan kunci publik perlu dibuat untuk mengubah pesan (*plaintext*) menjadi tidak bisa dibaca (*ciphertext*) oleh semua orang. Sedangkan kunci privat hanya diketahui oleh si pembuat kunci dan orang tertentu saja sehingga kunci privat tersebut mempunyai sifat yang rahasia. Maksudnya adalah bahwa saat akan melakukan proses dekripsi kunci privat d dan n yang merupakan kunci privat digunakan agar pesan yang tidak bisa dibaca (*ciphertext*) dapat dibaca (*plaintext*) untuk si penerima pesan.

### Menghitung Kunci Publik

Pada algoritma RSA menghitung kunci publik adalah hal yang pertama kali dilakukan saat akan mengenkripsi *file* teks. Teks yang sudah ada kemudian di enkripsi dengan menggunakan kunci publik. Alur dari proses perhitungan kunci publik adalah sebagai berikut:

1. Pilih nilai p dan q yang merupakan bilangan prima acak.
2. Hitung n yang merupakan hasil dari p\*q.
3. Hitung Q(n) yang merupakan hasil dari Q(n) = (p-1)\*(q-1).
4. Pilih nilai pada e yang merupakan kunci publik . syarat dari pembangkit kunci e ini adalah 1 < e < Q(n).

Adapun Flowchart menghitung kunci publik sebelum melakukan proses enkripsi dapat dilihat pada Gambar 4.1.



Gambar 4.1 Flowchart Proses Menghitung Kunci Publik RSA

Sebelum melakukan enkripsi, menghitung kunci publik adalah hal yang harus dilakukan seperti pada alur Gambar 4.1, tahap awal dilakukan dengan meng-input pasangan bilangan prima dimana nilai input p dan q merupakan bilangan prima acak dan n merupakan hasil dari input **p\*q**. Setelah melakukan proses tersebut nilai Q(n) di hitung dengan cara **Q(n) = (p-1)\*(q-1)**, serta Q(n) di gunakan juga untuk mencari kunci publik, kunci publik didapat dengan menggunakan cara **1 < e < Q(n)**, e adalah relatif prima dengan Q(n), relatif prima artinya nilai PBB (kunci publik, Q(n)) = 1. Setelah kunci publik didapat tidak perlu lagi meng-input nilai p dan q. Kemudian didapatlah hasil e sebagai proses sebelum melakukan enkripsi seperti pada pers 2.1 dan n digunakan dalam hal merubah (*plaintext*) menjadi (*ciphertext*) seperti pada tabel 2.3.

### Menghitung Kunci Privat

Pada algoritma RSA menghitung kunci privat adalah salah satu proses yang akan dilakukan setelah perhitungan kunci publik pada *file* teks selesai. teks yang sudah di enkripsi kemudian akan di dekripsi kan kembali dengan menggunakan kunci privat. Alur dari proses perhitungan kunci privat adalah sebagai berikut:

1. Pilih p, q, e. p dan q merupakan bilangan prima acak dan e adalah hasil dari proses perhitungan pada kunci publik.
2. Hitung Q(n) dengan cara Q(n) = (p-1)\*(q-1).
3. Input kunci publik e, e didapat dengan cara 1 < e < Q(n), e adalah relatif prima dengan Q(n), relatif prima artinya nilai PBB (kunci publik, Q(n)) = 1.
4. Pilih k, nilai k adalah salah satu bilangan bulat 1, 2, 3, ,.Qn.
5. Kunci privat dihitung dan didapat dengan cara d = 1+kQ(n)/e.

Adapun *Flowchart* menghitung kunci privat sebelum melakukan proses dekripsi dapat dilihat pada Gambar 4.2.



Gambar 4.2 *Flowchart* Proses Menghitung Kunci Privat RSA

Sebelum melakukan dekripsi, menghitung kunci privat pada alur Gambar 4.2 adalah hal yang harus dilakukan setelah melakukan proses perhitungan kunci publik seperti yang ada pada alur Gambar 4.1. Tahap awal dilakukan dengan meng-*input* p dan q merupakan bilangan prima acak dan e adalah kunci publik sehingga e berguna untuk untuk membantu dalam hal proses menghitung kunci privat, dimana e didapat dengan menggunakan cara **1 < e < Q(n)** dan e adalah relatif prima dengan Q(n). Setelah itu input k dimana nilai k merupakan bilangan bulat 1,2,3..,.Qn. Setelah itu kunci privat dihitung dan didapat dengan cara **d = 1+kQ(n)/e**. Kemudian didapat d sebagai kunci privat, dimana d akan digunakan dalam hal proses sebelum melakukan dekripsi seperti pada pers 2.2 dan n digunakan dalam hal merubah (*ciphertext*) menjadi (*plaintext*) seperti pada tabel 2.4.

## Proses Enkripsi Menggunakan Kunci Publik

Proses enkripsi menggunakan kunci pubik adalah suatu proses enkripsi menggunakan hasil dari perhitungan kunci publik yang ada pada alur (Gambar 4.1). Adapun alur dari proses enkripsi yang menggunakan perhitungan kunci publik adalah sebagai berikut :

* + - 1. *Input file* teks yang akan dienkripsi (*plaintext*).
      2. Pilih dua pasangan bilangan prima acak p dan q untuk proses enkripsi.
      3. Hitung n, nilai n digunakan untuk melakukan proses enkripsi.
      4. Hitung Qn, nilai Qn digunakan untuk mendapatkan kunci publik.
      5. Pilih nilai pada e yang merupakan kunci publik dan kunci publik ini didapatkan dengan menggunakan cara 1 < e < Q(n), e adalah relatif prima dengan Q(n).
      6. Proses enkripsi dilakukan berulang-ulang untuk semua huruf sesuai jumlah huruf pada *plaintext*.
      7. *Output*-nya adalah sebuah pesan acak yang tidak bisa di baca (*ciphertext*).

Setelah proses enkripsi maka di dapatkan teks yang tidak bisa di baca tanpa proses dekripsi.

*Flowchart* utama dalam hal enkripsi pesan dengan menggunakan algoritma RSA dapat dilihat pada Gambar 4.3.



Gambar 4.3 *Flowchart* Enkripsi Pesan Dengan Menggunakan Algoritam RSA

Pada saat melakukan enkripsi pesan hal yang pertama kali harus dilakukan adalah meng-*input plaintext* dimana nanti nya setiap karakter pada *plaintext* akan dibaca. Setelah itu input kunci publik dimana kunci publik didapat dengan menggunakan rumus 1 < e < Q(n), e adalah relatif prima dengan Q(n), relatif prima artinya nilai PBB (kunci publik, Q(n)) = 1. Kemudian hitung n dengan rumus n = p\*q. Setelah itu *plaintext* yang sudah menjadi karakter dipangkatkan dengan e, dimana e adalah kunci publik dan diproses dengan n seperti yang ada pada Tabel 2.3. Sehingga didapatlah hasil pesan berupa *ciphertext*.

*Flowchart* proses enkripsi menggunakan kunci publik pada Algoritma RSA dapat dilihat pada Gambar 4.4.



Gambar 4.4 *Flowchart* Proses Enkripsi Menggunakan Kunci Publik

Pada saat melakukan enkripsi adapun langkah-langkah yang harus dilakukan dalam hal proses merubah (*plaintext*) menjadi (*ciphertext*). Proses enkripsi yang ada pada alur Gambar 4.4 menunjukan *plaintext* di-*input*, kemudian nilai n didapat dengan rumus **n = p\*q**. Setelah itu *input* Q(n), nilai Q(n) didapat dengan rumus **Q(n) = (p-1)\*(q-1)**. kemudian *input* e, nilai e didapat dengan menggunakan rumus **1 < e < Q(n)**, e adalah relatif prima dengan Q(n). Nilai e tidak melalui proses perhitungan nilai n tetapi nilai e digunakan pada saat melakukan proses enkripsi. Selanjutnya *text* yang ada pada *file* dibaca kemudian urutan data yang ada pada *text* diurutkan dengan jumlah panjangnya huruf yang digunakan. Kemudian nilai **ab (m[i]\*\*e)** didapat dengan cara huruf-huruf yang sudah ada diubah kedalam huruf *ASCII* dipangkatkan dengan kunci publik. Setelah nilai ab didapat nilai ba di proses dengan mengikuti pers 2.1. Sehingga didapat ba yang merupakan *ciphertext* didapat dari proses enkripsi menggunakan kunci publik.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Plain Text | ASCII(M) | e | n | Me | Me Mod n | cipher text |
| K | 75 | 7 | 221 | 13,348,388,671,875 | 114 | r |
| R | 82 | 7 | 221 | 24,928,547,056,768 | 108 | l |
| I | 73 | 7 | 221 | 11,047,398,519,097 | 44 | , |
| P | 80 | 7 | 221 | 20,971,520,000,000 | 24 | Cancel |
| T | 84 | 7 | 221 | 29,509,034,655,744 | 33 | ! |
| O | 79 | 7 | 221 | 19,203,908,986,159 | 105 | i |
| G | 71 | 7 | 221 | 9,095,120,158,391 | 215 | × |
| R | 82 | 7 | 221 | 24,928,547,056,768 | 108 | L |
| A | 65 | 7 | 221 | 4,902,227,890,625 | 91 | [ |
| F | 70 | 7 | 221 | 8,235,430,000,000 | 60 | < |
| I | 73 | 7 | 221 | 11,047,398,519,097 | 44 | , |

Tabel 4.1 Proses Enkripsi Menggunakan Kunci Publik

*Plaintext* : “KRIPTOGRAFI”  
*ASCII* (M) : “75,82,73,80,84,79,71,82,65,70,73”  
Me : Merupakan proses nilai dari *ASCII* pada *plaintext* di pangkatkan dengan e sebagai kunci publik  
n : Merupakan nilai dari proses hasil sisa pembagian dari suatu bilangan terhadap bilangan lainnya atau bisa disebut juga dengan *mod*/*modulo*

Misal Nilai p=13 dan q=17 kemudian hitung n, yang dimana n merupakan *mod*/*modulo* dengan rumus **n = p\*q**, setelah itu hitung nilai Q(n) yang dimana **Q(n)=(p-1)\*(q-1)**. Setelah menghitung nilai Q(n), kunci publik didapat dengan rumus **1 < e < Q(n)**, e adalah relatif prima dengan Q(n). dimana nilai dari e merupakan kunci publik dan n digunakan dalam hal merubah (*plaintext*) menjadi (*ciphertext*) seperti pada Tabel 2.3.

p = 13 , q = 17  
n = 13 \* 17 = 221   
Q(n) = (13-1)\*(17-1)  
 = (12)\*(16)  
 = 192

Faktor dari 60 : 1,2,3,4,5,6,10,12,15,20,30,60  
Faktor dari 7 : 1,7

60 dan 7 relatif prima karena hanya memiliki faktor yang sama yaitu 1.

Sehingga kunci publik e,n adalah (7,221). Adapun perhitungan proses enkripsi seperti pada Tabel 4.1 adalah sebagai berikut :

*Plaintext* = “KRIPTOGRAFI”

*ASCII* (M) = “75,82,73,80,84,79,71,82,65,70,73”

n = “221”

C = Me Mod n

C = “757 Mod n : 757 Mod 221 = 114 “

“827 Mod n : 827 Mod 221 = 108 “

“737 Mod n : 737 Mod 221 = 44 “

“807 Mod n : 807 Mod 221 = 24 “

“847 Mod n : 847 Mod 221 = 33 “

“797 Mod n : 797 Mod 221 = 105 “

“717 Mod n : 717 Mod 221 = 215 “

“827 Mod n : 827 Mod 221 = 108 “

“657 Mod n : 657 Mod 221 = 91 “

“707 Mod n : 707 Mod 221 = 60 “

“737 Mod n : 737 Mod 221 = 44 “

*CipherText* = “rl,Cancel!i×L[<,”.

## Proses Dekripsi Menggunakan Kunci Privat

Proses dekripsi menggunakan kunci privat adalah suatu proses dekripsi yang mana setelah proses enkripsi dengan menggunakan perhitungan publik selesai. Maka hasil dari proses enkripsinya akan didekripsikan kembali dengan menggunakan proses perhitungan kunci privat yang ada pada RSA. Adapun alur dari proses dekripsi yang menggunakan kunci privat adalah sebagai berikut :

* + - 1. *Input ciphertext*.
      2. Input p,q,k. dimana p,q didapat dengan cara n = p\*q dan k di dapat dengan memilih nilai k, dimana k adalah salah satu bilangan bulat 1, 2, 3, ,.Qn.
      3. Proses dekripsi dilakukan berulang-ulang untuk semua huruf sesuai jumlah huruf pada *ciphertext*.
      4. *Output*-nya adalah pesan yang bisa dibaca (*plaintext*)

Setelah melalui proses enkripsi pesan yang tidak dapat dibaca kemudian bisa dibaca dengan menggunakan proses perhitungan kunci privat.

*Flowchart* utama dalam hal dekripsi pesan dengan menggunakan Algoritma RSA dapat dilihat pada Gambar 4.5.



Gambar 4.5 *Flowchart* Dekripsi Pesan Dengan Menggunakan Algoritma RSA

Setelah melakukan enkripsi pesan seperti pada alur Gambar 4.3, Proses dekripsi dilakukan seperti pada alur Gambar 4.5. Hal yang pertama kali dilakukan pada saat akan mengdekripsi pesan adalah meng-*input ciphertext*, dimana nanti nya setiap karakter pada *ciphertext* akan dibaca. Setelah itu *input* kunci privat dimana kunci privat didapat dengan rumus d = 1+kQ(n)/e, kemudian input n dimana nilai n dihitung dengan rumus n = p\*q. setelah itu *chipertext* yang sudah dibaca menjadi karakter dipangkatkan dengan kunci privat dan di proses dengan n seperti yang ada pada Tabel 2.4. Sehingga didapatlah hasil pesan berupa *plaintext*.

*Flowchart* proses dekripsi menggunakan kunci privat pada Algoritma RSA dapat dilihat pada Gambar 4.6.



Gambar 4.6 *Flowchart* Proses Dekripsi Menggunakan Kunci Privat

Setelah melakukan proses enkripsi menggunakan kunci publik seperti pada alur Gambar 4.4. Kemudian proses dekripsi dilakukan dengan menggunakan kunci privat seperti pada alur Gambar 4.6. Pada saat akan melakukan dekripsi adapun langkah-langkah yang harus dilakukan dalam hal proses merubah (*chipertext*) menjadi (*plaintext*) proses dekripsi yang ada pada alur Gambar 4.6 menunjukan *chipertext* di-*input* kemudian nilai n didapat dan dihitung dengan menggunakan rumus **n = p\*q**, dan k di dapat dengan memilih nilai k, dimana nilai k adalah salah satu bilangan bulat 1, 2, 3, ,.Qn. kemudian hasil dari proses enkripsi yang ada pada alur Gambar 4.4 dipangkatkan dengan kunci privat (**c=i\*\*k/n**) berdasarkan pada pemrograman berbasis python kemudian nilai c di proses dengan mengikuti pers 2.2. Sehingga didapat c yang merupakan *plaintext* dari proses dekripsi menggunakan kunci privat.

Tabel 4.2 Proses Deskripsi Menggunakan Kunci Privat

| Chiper Text | ASCII(C) | d | n | Cd | Cd Mod n | Plain Text |
| --- | --- | --- | --- | --- | --- | --- |
| r | 114 | 55 | 221 | 1,35E+116 | 75 | K |
| l | 108 | 55 | 221 | 6,89E+114 | 82 | R |
| , | 44 | 55 | 221 | 2,45E+94 | 73 | I |
| Cancel | 24 | 55 | 221 | 8,16E+79 | 80 | P |
| ! | 33 | 55 | 221 | 3,30E+87 | 84 | T |
| i | 105 | 55 | 221 | 1,46E+114 | 79 | O |
| × | 215 | 55 | 221 | 1,92E+131 | 71 | G |
| l | 108 | 55 | 221 | 6,89E+114 | 82 | R |
| [ | 91 | 55 | 221 | 5,59E+110 | 65 | A |
| < | 60 | 55 | 221 | 6,29E+101 | 70 | F |
| , | 44 | 55 | 221 | 2,45E+94 | 73 | I |

*ASCII* (C) : “114,108,44,24,33,105,215,108,91,60,44”  
Cd : Merupakan proses nilai dari *ASCII* (C) pada *chipertext* di pangkatkan dengan d sebagai kunci privat  
n : Merupakan nilai dari proses hasil sisa pembagian dari suatu bilangan terhadap bilangan lainnya atau bisa disebut juga dengan *mod*/*modulo*

Jika nilai dari e dan n (7,221) adalah perhitungan proses kunci publik seperti pada Tabel 4.1. Maka lakukanlah proses perhitungan nilai dari d sebagai kunci privat dengan menggunakan rumus **d=1+kQ(n)/e**.

d = 1+(2\*(192))/7

d = 1+(384/7)

d = 1+54

d = 55

Sehingga diperoleh proses perhitungan kunci privat d,n adalah (55,221). Adapun perhitungan proses dekripsi seperti pada Tabel 4.2 adalah sebagai berikut :

*ChiperText* = “rl,Cancel!i×L[<,”

*ASCII* (C) = “114,108,44,24,33,105,215,108,91,60,44”

n = “221”

M = Cd Mod n

M = “11455 Mod n : 11455 Mod 221 = 75 “

“10855 Mod n : 10855 Mod 221 = 82 “

“4455 Mod n : 4455 Mod 221 = 73 “

“2455 Mod n : 2455 Mod 221 = 80 “

“3355 Mod n : 3355 Mod 221 = 84 “

“10555 Mod n : 10555 Mod 221 = 79 “

“21555 Mod n : 21555 Mod 221 = 71 “

“10855 Mod n : 10855 Mod 221 = 82 “

“9155 Mod n : 9155 Mod 221 = 65 “

“6055 Mod n : 6055 Mod 221 = 70 “

“4455 Mod n : 4455 Mod 221 = 73 “

*PlainText* = “KRIPTOGRAFI”

1. **Pengujian**

Pengujian dilakukan untuk mengimplementasikan Algoritma kriptografi RSA untuk mengenkripsi data teks yang ada pada file menggunakan pemrograman Python. Editor pemrograman yang digunakan adalah eclipse. Dengan data teks yang tersimpan di dalam file (.txt). Contoh data yang digunakan untuk proses enkripsi adalah teks “KRIPTOGRAFI”. Selanjutnya data teks yang ada pada noteped diproses lagi untuk mencari *ASCII Number* dengan *Plain Char* “KRIPTOGRAFI” menggunakan pemrograman yang telah dibuat. Sehingga diperoleh *ASCII Number* seperti yang ada pada Tabel 4.3.

Tabel 4.3 *Plain Char* Dan *ASCII Number*

|  |  |
| --- | --- |
| Plain Char | ASCII Number |
| K | 75 |
| R | 82 |
| I | 73 |
| P | 80 |
| T | 84 |
| O | 79 |
| G | 71 |
| R | 82 |
| A | 65 |
| F | 70 |
| I | 73 |

Setelah mendapatkan *ASCII Number* seperti pada Tabel 4.3 untuk melakukan proses enkripsi selanjutnya diperlukan pasangan bilangan prima (p dan q) adalah 13 dan 17. Dari pasangan bilangan prima yang digunakan diperoleh n = 221, qn = 192, dan kunci publik = 7. Kemudian proses enkripsi dilakukan dengan mengikuti alur pada Gambar 4.4. Sehingga diperoleh hasil enkripsi berupa numerik seperti yang ditunjukan pada Tabel 4.4.

Tabel 4.4 Hasil Enkripsi Pada Program

|  |  |  |  |
| --- | --- | --- | --- |
| PlainText | | CipherText | |
| Char | ASCII | ASCII | Char |
| S | 83 | 941 | έ |
| a | 97 | 791 | ̗' |
| m | 109 | 1093 | х |
| a | 97 | 791 | ̗' |
| r | 114 | 1003 | ϫ |
| i | 105 | 117 | u |
| n | 110 | 1035 | Ћ |
| d | 100 | 803 | ̣' |
| a | 97 | 791 | ̗' |
|  | 32 | 1055 | П |
| A | 65 | 761 | ˹ |

Setelah melakukan proses enkripsi seperti pada Tabel 4.4. Dihasilkan teks proses enkripsi adalah “rl,Cancel!i×l[<,”.

Selanjutnya proses dekripsi dilakukan dengan mengikuti alur pada Gambar 4.6. untuk melakukan proses dekripsi selanjutnya diperlukan pasangan bilangan prima (p dan q) adalah 13 dan 17 dengan kunci privat = 55. Sehingga diperoleh data *plaintext* “KRIPTOGRAFI” dalam bentuk karakter seperti yang ada pada Tabel 4.5.

Tabel 4.5 Hasil Dekripsi Pada Program

| CipherText | | PlainText | |
| --- | --- | --- | --- |
| ASCII | Char | Char | ASCII |
| 114 | r | K | 75 |
| 108 | l | R | 82 |
| 44 | , | I | 73 |
| 24 |  | P | 80 |
| 33 | ! | T | 84 |
| 105 | i | O | 79 |
| 215 | × | G | 71 |
| 108 | l | R | 82 |
| 91 | [ | A | 65 |
| 60 | < | F | 70 |
| 44 | , | I | 73 |

Setelah melakukan proses dekripsi seperti pada Tabel 4.5. Dihasilkan teks proses dekripsi adalah “KRIPTOGRAFI”.

Pengujian lain dilakukan dengan membuat data *text* berisi huruf dan angka pada notepad dengan data yang tersimpan kedalam notepad dengan jumlah data yang berbeda dengan data pada Tabel 4.3. Contoh data yang akan digunakan untuk proses enkripsi adalah teks “Samarinda Angkatan 2014”. Selanjutnya data teks yang ada pada noteped diproses lagi untuk mencari ASCII Number dengan *Plain Char* “Samarinda Angkatan 2014” menggunakan pemrograman yang telah dibuat. Sehingga diperoleh *ASCII Number* seperti yang ada pada Tabel 4.6.

Tabel 4.6 *Plain Char* Dan *ASCII Number*

| Plain Char | ASCII Number |
| --- | --- |
| S | 83 |
| a | 97 |
| m | 109 |
| a | 97 |
| r | 114 |
| i | 105 |
| n | 110 |
| d | 100 |
| a | 97 |
|  | 32 |
| A | 65 |
| n | 110 |
| g | 103 |
| k | 107 |
| a | 97 |
| t | 116 |
| a | 97 |
| n | 110 |
|  | 32 |
| 2 | 50 |
| 0 | 48 |
| 1 | 49 |
| 4 | 52 |

Setelah mendapatkan *ASCII Number* seperti pada Tabel 4.6 untuk melakukan proses enkripsi selanjutnya diperlukan pasangan bilangan prima (p dan q) adalah 31 dan 37. Dari pasangan bilangan prima yang digunakan diperoleh n = 1147, qn = 1080, dan kunci publik = 7. Kemudian proses enkripsi dilakukan dengan mengikuti alur pada Gambar 4.4. Sehingga diperoleh hasil enkripsi berupa numerik seperti yang ditunjukan pada Tabel 4.7.

Tabel 4.7 Hasil Enkripsi Pada Program

| PlainText | | CipherText | |
| --- | --- | --- | --- |
| Char | ASCII | ASCII | Char |
| S | 83 | 941 | έ |
| a | 97 | 791 | ̗' |
| m | 109 | 1093 | х |
| a | 97 | 791 | ̗' |
| r | 114 | 1003 | ϫ |
| i | 105 | 117 | u |
| n | 110 | 1035 | Ћ |
| d | 100 | 803 | ̣' |
| a | 97 | 791 | ̗' |
|  | 32 | 1055 | П |
| A | 65 | 761 | ˹ |
| n | 110 | 1035 | Ћ |
| g | 103 | 82 | R |
| k | 107 | 81 | Q |
| a | 97 | 791 | ̗' |
| t | 116 | 277 | ĕ |
| a | 97 | 791 | ̗' |
| n | 110 | 1035 | Ћ |
|  | 32 | 1055 | П |
| 2 | 50 | 69 | E |
| 0 | 48 | 973 | ύ |
| 1 | 49 | 9 | \t |
| 4 | 52 | 1034 | Њ |

Setelah melakukan proses enkripsi seperti pada Tabel 4.7. Dihasilkan teks proses enkripsi adalah “έ̗'х̗'ϫuЋ ̣'̗'П˹ЋRQ̗'ĕ̗'ЋПEύ\tЊ”.

Setelah melakukan proses enkripsi seperti pada Tabel 4.7 Selanjutnya proses dekripsi dilakukan dengan mengikuti alur pada Gambar 4.6. untuk melakukan proses dekripsi selanjutnya diperlukan pasangan bilangan prima (p dan q) adalah 31 dan 37 dengan kunci privat = 463. Sehingga diperoleh data *plaintext* “Samarinda Angkatan 2014” dalam bentuk karakter seperti yang ada pada Tabel 4.8.

Tabel 4.8 Hasil Dekripsi Pada Program

| CipherText | | PlainText | |
| --- | --- | --- | --- |
| ASCII | Char | Char | ASCII |
| 941 | έ | S | 83 |
| 791 | ̗' | a | 97 |
| 1093 | х | m | 109 |
| 791 | ̗' | a | 97 |
| 1003 | ϫ | r | 114 |
| 117 | u | i | 105 |
| 1035 | Ћ | n | 110 |
| 803 | ̣' | d | 100 |
| 791 | ̗' | a | 97 |
| 1055 | П |  | 32 |
| 761 | ˹ | A | 65 |
| 1035 | Ћ | n | 110 |
| 82 | R | g | 103 |
| 81 | Q | k | 107 |
| 791 | ̗' | a | 97 |
| 277 | ĕ | t | 116 |
| 791 | ̗' | a | 97 |
| 1035 | Ћ | n | 110 |
| 1055 | П |  | 32 |
| 69 | E | 2 | 50 |
| 973 | ύ | 0 | 48 |
| 9 | \t | 1 | 49 |
| 1034 | Њ | 4 | 52 |

Setelah melakukan proses dekripsi seperti pada Tabel 4.8. Dihasilkan teks proses dekripsi adalah “Samarinda Angkatan 2014”.

1. **Analisa Hasil**

Berdasarkan hasil pada contoh pengujian Algoritma RSA menggunakan kunci asimteris. Kunci asimetris dibangun menggunakan pasangan bilangan prima. Pada penelitian ini diterapkan algoritma RSA dalam melakukan proses enkripsi pesan teks (*plaintext*) menjadi pesan acak (*ciphertext*) dan proses dekripsi pesan acak menjadi pesan teks aslinya berhasil dilakukan. Adapun kekurangan implementasi algoritma RSA dalam penelitian ini adalah pasangan bilangan prima berurutan terkecil yang dapat digunakan untuk menentukan kunci publik dan kunci privat adalah 11 dan 13. Kemudian untuk pasangan bilangan prima terbesar pada penelitian ini dibatasi sampai maksimal 997.