BAB IV

PERANCANGAN PERANGKAT LUNAK

Tahap perancangan perangkat lunak dilakukan setelah tahapan analisis kebutuhan selesai dan didefinisikan secara jelas. Dalam tahap ini digambarkan lebih rinci berdasarkan tahap sebelumnya, sehingga diperoleh algoritma dan detail aliran proses dari analisa sistem yang akan dibuat.

Hasil dari tahapan perancangan ini harus dapat diterapkan menjadi procedure dengan alat Bantu bahasa pemrograman Borland Delphi 7. Metode perancangan yang digunakan yaitu memakai metode *flowchart* untuk mengetahui aliran setiap proses yang terjadi pada proses enkripsi/dekripsi menggunakan algoritma RSA.

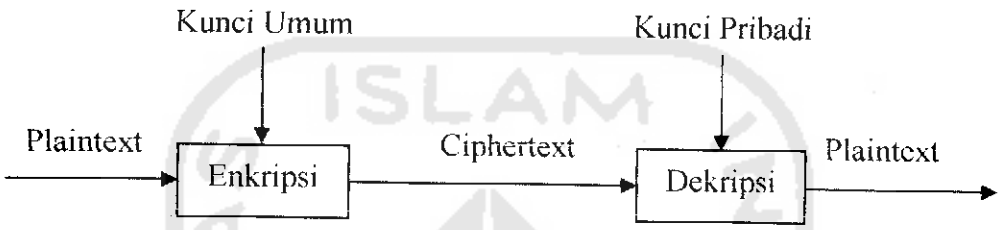
4.1 Metode Perancangan

Metode perancangan yang diterapakan untuk implementasi penyamaran data ini menggunakan metode perancangan diagram alir atau flowchart yang mengacu kepada informasi yang akan ditampilkan. Perancangan tersebut menggunakan model algoritma untuk manggambarkan logika proses

4.2 Rancangan Proses

Program yang dibuat memiliki kemampuan melakukan proses enkripsi dan dekripsi. Setiap proses enkripsi akan menggunakan sebuah kunci yang disebut kunci publik dan setiap proses dekripsi akan menggunakan sebuah kunci yang disebut kunci privat. Pada proses pertama akan diberikan plainteks/data asli yang

akan dienkripsi yang mana menghasilkan cipherteks yang merupakan data yang sudah tidak dapat dibaca. Kemudian darai cipherteks tersebut dengan menggunakan kunci privat didekrip supaya menjadi data sebagaimana asalnya (palinteks).



Gambar 4.1. Proses enkripsi dan dekripsi menggunakan publik key dan privat key

4.3 Perancangan Algoritma RSA

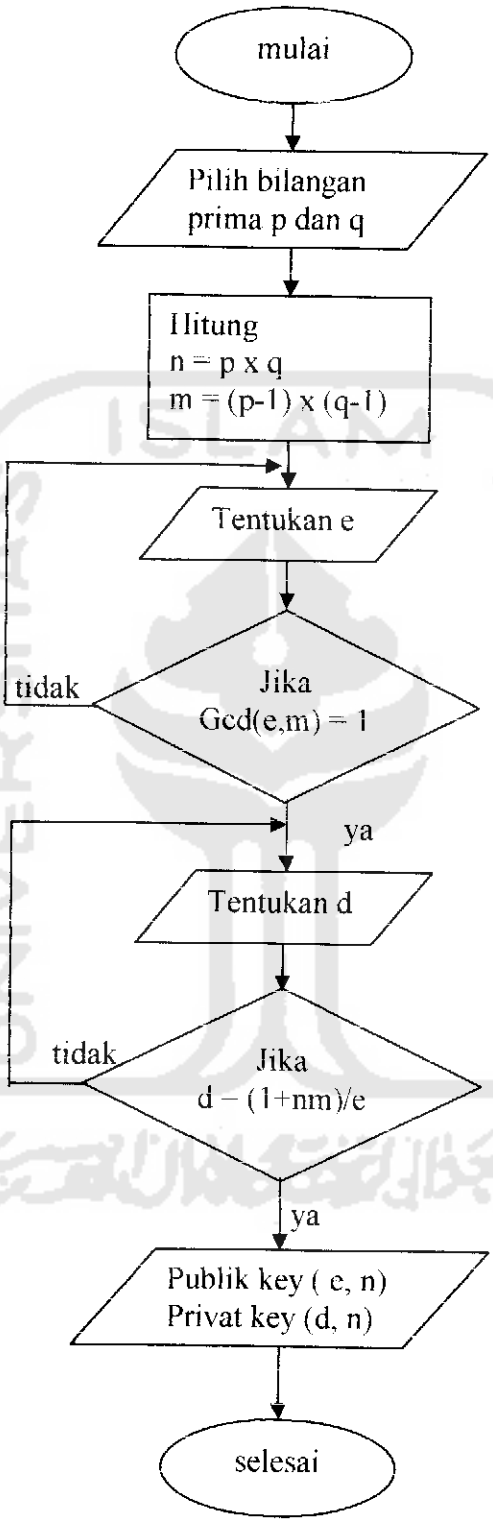
4.3.1 Perancangan Pembangkitan Kunci RSA

Untuk pembangkitan kunci, pertama dicari dua buah bilangan prima secara random (acak), yaitu *p* dan *q* yang digunakan untuk mencari n *dan* m, kemudian digunakan *euclid algorithm*, yaitu mengunakan *Greatest Common Divisor* (GCD). Fungsi ini digunakan untuk memperoleh pasangan kunci privat dan kunci publik, melalui proses pengulangan sampai kondisi yang diinginkan terpenuhi. Pada algoritma ini, kunci privat yang digunakan adalah *d*. Nilai *d* didapatkan dari nilai fungsi random yang lebih besar dari 1 dan lebih kecil dai m , dan *d* adalah bilangan prima yang relatif prima terhadap m dimana m adalah hasil perkalian dua bilangan relatif prima *p-1* dan *q-1*. Perhitungan kunci publik didapat dari fungsi invers d yang di-modulo oleh m

Sebenarnya penggunaan kunci yang panjang akan memiliki tingkat keamanan yang lebih tinggi, namun pada model simulasi ini pasangan bilangan prima *p* dan *q* untuk pembangkit kunci yang digunakan dibatasi hanya sampai angka 100000. Hal ini dikarenakan keterbatasan memori dan space hardisk yang tersedia, karena semakin besar bilangan prima maka akan menghasilkan kunci yang besar juga ini berakibat pada penggunaan memori dalam proses enkripsi/dekripsi serta kebutuhan hardisk yang besar dari akibat pembengkakan ukuran file dari proses enkripsi. Adapun algoritma pembangkitan kunci sebagai berikut:

1. Hasilkan dua buah integer prima, p dan q
2. Hitung m = (p-1)*(q-1)
3. Hitung n = p*q
4. Pilih e yang relatif prima terhadap m. e relatif prima terhadap m artinya faktor pembagi terbesar keduanya adalah 1, secara matematis disebut gcd(e,m) = 1. Untuk mencarinya dapat digunakan algoritma Euclid.
5. Cari d, sehingga e*d = 1 mod (m), atau d = (1+nm)/e Untuk bilangan besar, dapat digunakan algoritma extended Euclid.
6. Kunci publik : e, n
7. Kunci private : d, n

Proses pembangkitan kunci dapat dilihat pada gambar flowchart berikut



Gambar 4.2. Flowchart pembangkitan kunci RSA

Keterangan gambar Flowchart

- p dan q adalah bilangan prima desimal yang dipilih secara acak
- n adalah hasil kali dari bilangan prima p dan q. n akan digunakan sebagai kunci untuk privat dan publik
- m adalah hasil kali (p-1) dengan (q-1). m akan digunakan untuk menentukan e yang merupakan kunci publik
- e adalah sebuah bilangan yang relatif prima terhadap m dipilih secara acak. Untuk mengetahui e relatif prima terhadap m atau tidak maka digunkan teorema euluer (gcd (e,m) = 1)
- gcd (*Greatest Common Divisor)* adalah cara untuk mencari faktor pembagi bersama terbesar.
- d adalah bilangan yang dipilih secara acak yang harus memenuhi syarat (d=(1+nm)/e)
- nm adalah n dikalikan m

4.3.2 Perancangan Proses Enkripsi RSA

Setalah memilih 2 buah bilangan prima dan menghasilkan dua buah kunci. Langkah selanjutya yaitu mengenkripsi data. Proses enkripsi data dengan algorima RSA adalah proses pemangkatan terhadap bilangan, dimana plainteks yang akan dienkripsi diubah kedalam bentuk desimal kemudian dipangkatkan dengan niali d (kunci publik) kemudian di *mod* dengan nilai n dimana n = p x q. sebagai contoh proses perhitungan enkripsi RSA adalah sebagai berikut:

Misalkan diperoleh kunci publik:

e = 3 dan n = 33

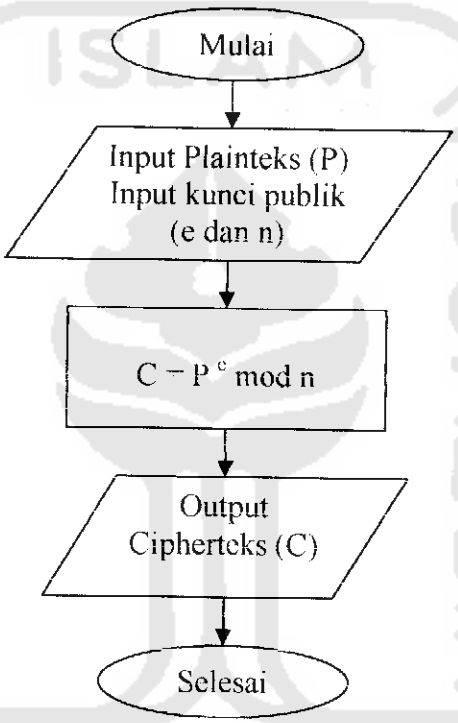
data yang akan dienkripsi (P) adalah 20 (plainteks)

C $= P^{e} \bmod n$

$= 20^{3} \bmod 33$

$= 8$

Proses enkripsi RSA dapat dilihat dalam gambar flowchart berikut



Gambar 4.3. Flowchart proses enkripsi RSA

Keterangan gambar 4.3.

- Plainteks (P) adalah teks yang bisa dibaca. Teks ini akan dienkripsi
- C adalah cipherteks yaitu teks hasil enkripsi. Teks ini sudah tidak dapat dibaca atau dipahami
- Untuk mengenkripsi palainteks (P) maka P dipangkat e kemudian dimod n

- e dan n adalah pasangan kunci publik yang digunakan untuk mengenkrip teks (P)

4.3.3 Perancangan Proses Dekripsi RSA

Pada proses dekripsi RSA sama dengan proses enkripsi RSA, hanya saja berbeda pada bilangan yang digunakan untuk pemangkatan. kalau pada proses enkripsi itu menggunakan kunci publik (e , n) maka dalam proses dekripsi menggunakan kunci privat (d , n). sebagai contoh proses perhitungan dekripsi RSA adalah sebagai berikut: Misalkan diperoleh kunci privat:

d = 7 dan n = 33

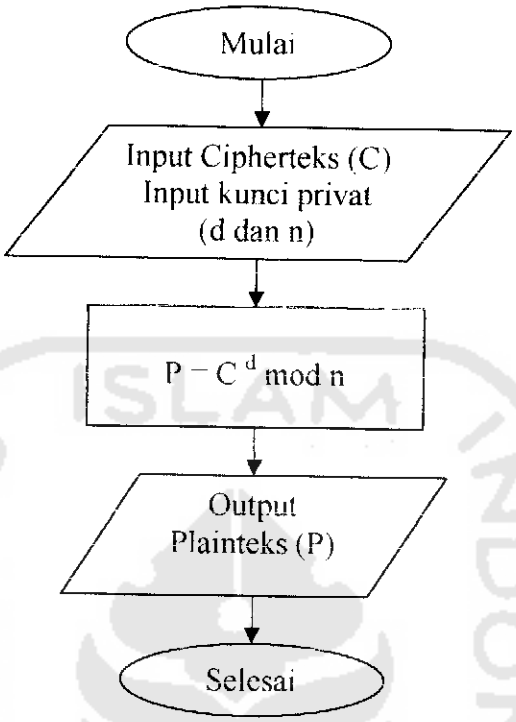
data yang akan didekrip (C) adalah : 8

P $= C^{d} \bmod n$

$= 8^{7} \bmod 33$

$= 20$

Proses dekripsi RSA dapat dilihat dalam gambar flowchart berikut



Gambar 4.4. Flowchart proses enkripsi RSA

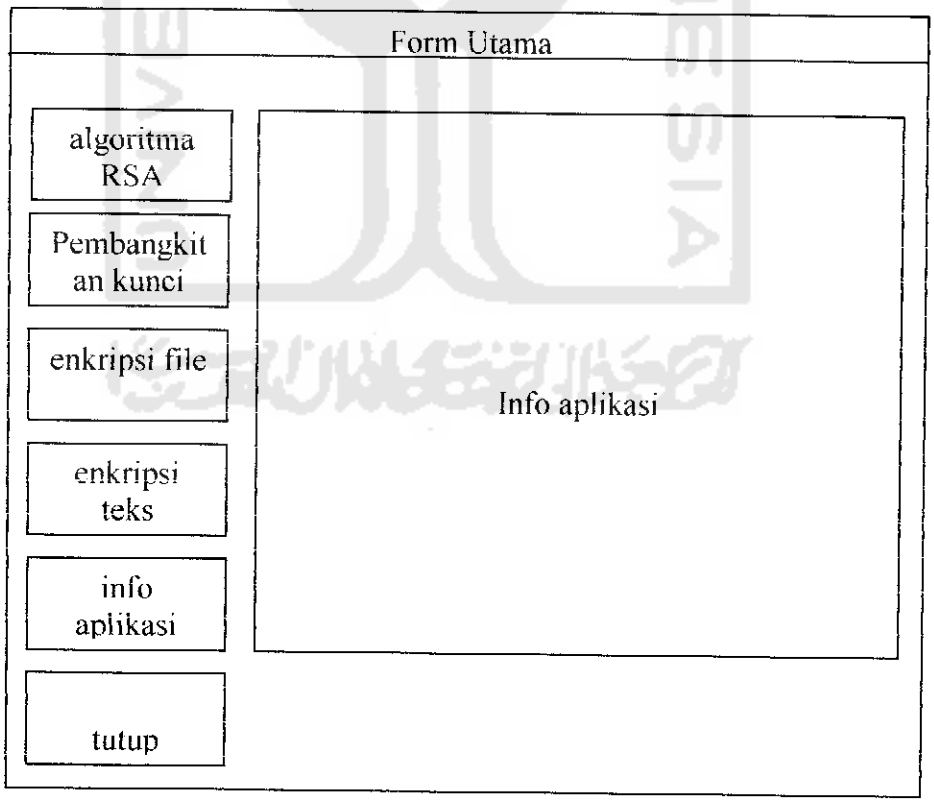
Keterangan gambar 4.4.

- Cipherteks (C) adalah teks yang tidak bisa dibaca. Teks ini akan didekripsi
- P adalah plainteks yaitu teks hasil dekripsi. Teks ini adalah teks yang sudah dapat dibaca kembali.
- Untuk mendekripsi cipherteks maka chiperteks (C) dipangkat d kemudian dimod dengan n
- d dan n adalah pasangan kunci privat yang digunakan untuk mendekrip teks (C)

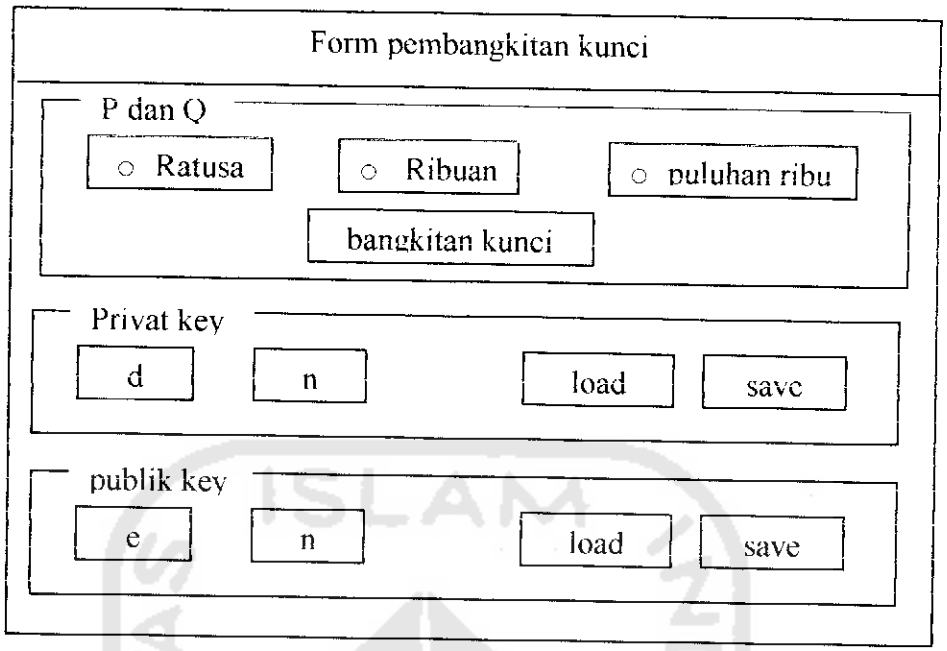
4.4 Perancangan Antarmuka

Antarmuka adalah bagian yang peting untuk sebuah aplikasi, hal ini dikarenakan tidak semua user atau pengguna aplikasi mempunyai kemampuan yang sama dalam penggunaan komputer maka dari itu pernacangan sebuah interface yang *user friendly* adalah hal yang peting. Dengan rancangan yang bagus diharapkan ketika aplikasi sudah jadi semua user dapat mengoperasika aplikasi tersebut walaupun kemampuan pemakaian komputer yang dimiliki user sangat minim.

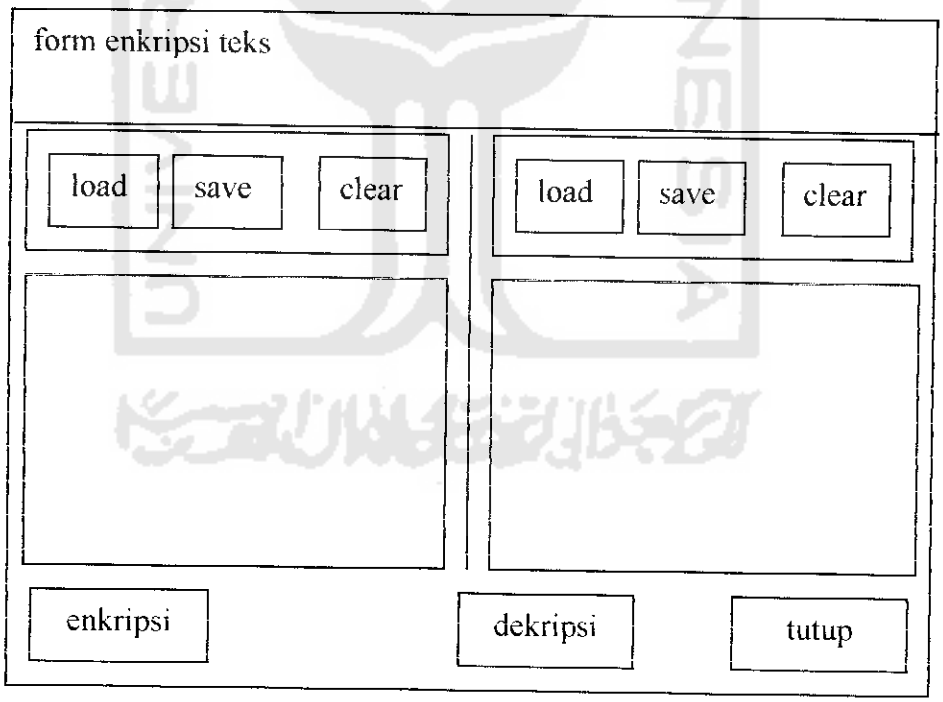
Adapun rancangan interface untuk aplikasi kriptografi RSA yang akan dibuat adalah sebagai berikut:



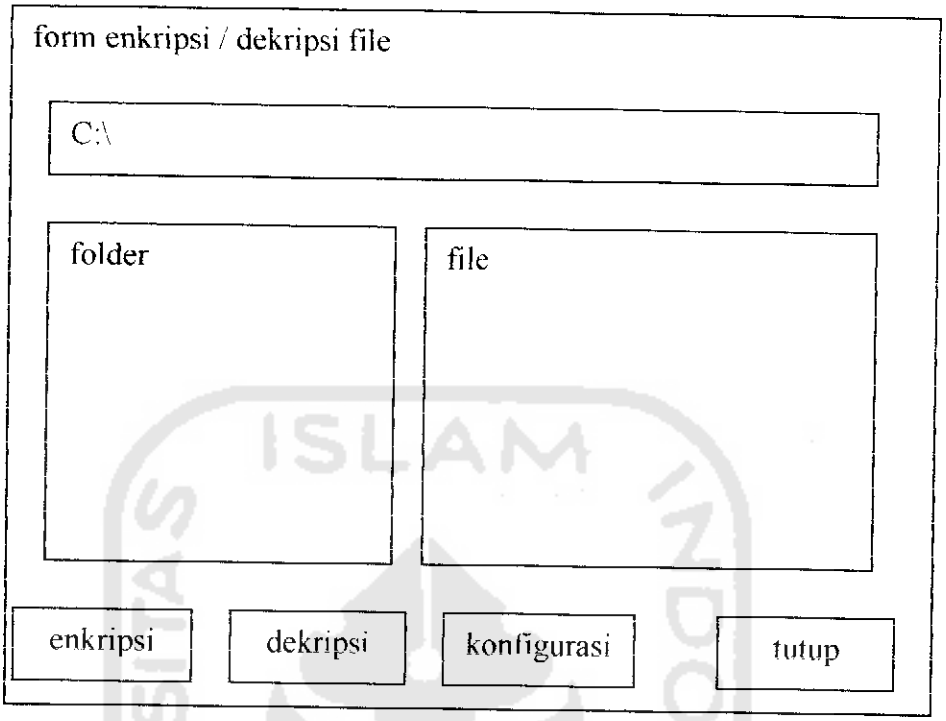
Gambar 4.5. Rancangan form menu utama



Gambar 4.6. Rancangan form pembangkitan kunci



Gambar 4.7. Rancangan form enkripsi/dekripsi teks



Gambar 4.8. Rancangan form enkripsi/dekripsi file