

## **BAB III**

### **ANALISA DAN DESAIN SISTEM**

#### **III.1. Analisa Masalah**

Analisis masalah bertujuan untuk mengidentifikasi permasalahan yang ada pada sistem dimana aplikasi dibangun, meliputi perangkat keras (*hardware*), perangkat lunak (*software*) dan pengguna (*user*). Analisis ini diperlukan sebagai dasar bagi tahapan perancangan sistem. Analisis sistem meliputi identifikasi permasalahan, analisis sistem, Analisis Kriptografi, Analisis proses *Enkripsi*, Analisis proses *Dekripsi*.

Keamanan informasi adalah suatu keharusan yang perlu diperhatikan apalagi jika informasi itu bersifat rahasia. Salah satu media yang paling sering digunakan untuk saling bertukar *file*. Akan tetapi apabila aplikasi ini tidak memiliki sistem keamanan untuk melindungi seluruh *file* yang terjadi di dalamnya, tentu ada kemungkinan *file* tersebut disadap oleh pihak-pihak yang tidak berhak tanpa ada kesulitan sekalipun untuk membaca isi dari *file* atau informasi tersebut.

Untuk mengatasi hal ini, penulis akan mencoba menerapkan Algoritma Kriptografi yaitu Algoritma *Blowfish* kolaborasi dengan Algoritma Rijndael sebagai sistem pengamannya. Hal ini diharapkan mampu untuk melindungi seluruh *file* yang terjadi di dalamnya dengan cara mengenkripsi *file* dikirim ke penerima.

### III.2. Penerapan Metode

Dalam penelitian ini digunakan Metode studi pustaka yang dilakukan dengan cara mengumpulkan literatur dan *Referensi* yang berhubungan dengan Algoritma Kriptografi *Blowfish* kolaborasi dengan Algoritma Rijndael. Selain itu peneliti juga mengumpulkan literatur dan Referensi yang berhubungan dengan pengembangan Aplikasi *dekstop*. Referensi atau data-data yang telah terkumpul disortir sesuai dengan bidang pembahasan mengenai Algoritma Kriptografi *Blowfish* dan Algoritma Rijndael yang akan digunakan untuk mengenkripsi *file*. Dalam proses pengamanan *file* hal pertama yang dilakukan adalah menerapkan Algoritma *Blowfish* dan Algoritma Rijndael dalam me-*enkripsi* data.

#### III.2.1. Algoritma Blowfish

*Blowfish* merupakan Algoritma Kriptografi dengan penggunaan kunci pada blok *cipher* simetris (*symmetric block cipher*) yakni kunci yang digunakan pada proses *Enkripsi* sama dengan kunci yang digunakan pada proses *Dekripsi* dengan data masukan dan keluaran berupa blok-blok data berukuran 64 *bit*. *Blowfish* dirancang oleh *Bruce Schneier* pada tahun 1993 yang ditujukan untuk *mikroprosesor* besar (32 *bit* ke atas dengan *cache* data yang besar). (Dedi Dony, 2016 : 139).

Didalam proses ini terdapat *file* biner hasil konvers *file* asli kedalam *biner* dan kunci yang dibangkitkan terlebih dahulu sebelum dilakukan proses *Enkripsi* atau *Dekripsi* data. Kunci- kunci yang digunakan antara lain terdiri dari, 18 buah yang memiliki 32-*bit subkey* yang tergabung dalam *P-array* (P1, P2, ..., P18).

Selain itu, ada empat 32-bit *S-box* yang masing-masingnya memiliki 256 entri :  
 $S1,0, S1,1, \dots, S1,255. S2,0, S2,1, \dots, S2,255. S3,0, S3,1, \dots, S3,255. S4,0, S4,1, \dots, S4,255$

Konsep dari Algoritma *Blowfish* adalah bahwa setiap *bit file* akan ditambahkan dengan *bit* yang berasal dari kunci – kunci pada *P-array* dan *S-box*. Dalam penambahan *bit* harus mencukupi 64 *bit*, jika melebihi dari 64 *bit* maka akan dilakukan proses perulangan.

### 1. Ekspansi kunci (*Key-expansion*)

Berfungsi merubah kunci (minimum 32-bit, maksimum 448-bit) menjadi beberapa *array subkunci* (subkey) dengan total 4168 *bit* (18x32-bit untuk *Parray* dan 4x256x32-bit untuk *S-box* sehingga totalnya 33344 *bit* atau 4168 *bit*). Kunci disimpan dalam *K-array*:

$K1, K2, \dots, Kj \quad 1 \leq j \leq 14$

Kunci-kunci ini yang dibangkitkan (*generate*) dengan menggunakan *subkunci* yang harus dihitung terlebih dahulu sebelum *enkripsi* atau *dekripsi* data. *Sub-sub* kunci yang digunakan terdiri dari : *P-array* yang terdiri dari 18 buah 32-bit *subkunci*,  $P1, P2, \dots, P18$  *S-box* yang terdiri dari 4 buah 32-bit, masing-masing memiliki 256 entri :  $S1,0, S1,1, \dots, S1,255 S2,0, S2,1, \dots, S2,255 S3,0, S3,1, \dots, S3,255 S4,0, S4,1, \dots, S4,255$  Langkah-langkah perhitungan atau pembangkitan *subkunci* tersebut adalah sebagai berikut:

1. Inisialisasi *P-array* yang pertama dan juga empat *S-box*, berurutan, dengan *string* yang telah pasti.

*String* tersebut terdiri dari digit-digit *heksadesimal* dari *phi*, tidak termasuk

angka tiga di awal.

Contoh :

P1= 0x243f6a88 P2= 0x85a308d3 P3= 0x13198a2e P4= 0x03707344 dan seterusnya sampai *S-box* yang terakhir (daftar *heksadesimal* digit dari *phi* untuk *P-array* dan *Sbox*.

2. *XOR*-kan P1 dengan 32-bit awal kunci, *XOR*-kan P2 dengan 32-bit berikutnya dari kunci, dan seterusnya untuk semua *bit* kunci. Ulangi siklus seluruh *bit* kunci secara berurutan sampai seluruh *P-array* ter-*XOR*-kan dengan *bit-bit* kunci. Atau jika disimbolkan :  $P1 = P1 \_ K1$ ,  $P2 = P2 \_ K2$ ,  $P3 = P3 \_ K3$ , . . .  $P14 = P14 \_ K14$ ,  $P15 = P15 \_ K1$ , . . .  $P18 = P18 \_ K4$ . Keterangan : adalah simbol untuk *XOR*.
3. Enkripsikan *string* yang seluruhnya nol (*all-zero string*) dengan Algoritma *Blowfish*, menggunakan *subkunci* yang telah dideskripsikan pada langkah 1 dan 2.
4. Gantikan P1 dan P2 dengan keluaran dari langkah 3.
5. Enkripsikan keluaran langkah 3 menggunakan Algoritma *Blowfish* dengan *subkunci* yang telah dimodifikasi.
6. Gantikan P3 dan P4 dengan keluaran dari langkah 5.
7. Lanjutkan langkah-langkah di atas, gantikan seluruh elemen *P-array* dan kemudian keempat *S-box* secara berurutan, dengan hasil keluaran Algoritma *Blowfish* yang terus-menerus berubah.

Total keseluruhan, terdapat 521 iterasi untuk menghasilkan *subkunci* dan membutuhkan memori sebesar 4KB Enkripsi Data

## 2. Enkripsi Data

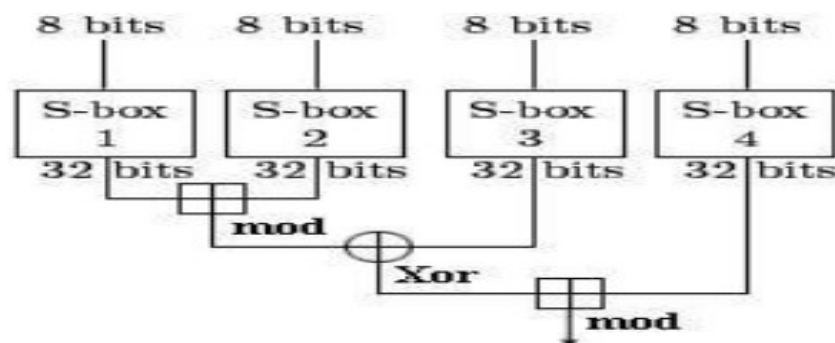
Terdiri dari *iterasi* fungsi sederhana (*Feistel Network*) sebanyak 16 kali putaran (*iterasi*), masukannya adalah 64-bit elemen data  $X$ . Setiap putaran terdiri dari permutasi kunci-*dependent* dan *substitusi* kunci- dan data *dependent*. Semua operasi adalah penambahan (*addition*) dan *XOR* pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel *array* berindeks untuk setiap putaran. Langkahnya adalah seperti berikut.

1. Bagi  $X$  menjadi dua bagian yang masing-masing terdiri dari 32-bit:  $XL$ ,  $XR$ .
2. Lakukan langkah berikut
3. *For*  $i = 1$  to 16:
4.  $XL = XL \_ P_{19}$
5.  $XR = F(XL) \_ XR$
6. Tukar  $XL$  dan  $XR$
7. Setelah iterasi ke-16, tukar  $XL$  dan  $XR$  lagi untuk melakukan membatalkan pertukaran terakhir.
8. Lalu lakukan
9.  $XR = XR \_ P_{17}$
10.  $XL = XL \_ P_{18}$
11. Terakhir, gabungkan kembali  $XL$  dan  $XR$  untuk mendapatkan *cipherteks*.

Untuk lebih jelasnya, gambaran tahapan pada jaringan *feistel* yang digunakan *Blowfish* adalah seperti pada Gambar III.1.

**Gambar III.1. Diagram *Enkripsi* Algoritma *Blowfish***

Pada langkah kedua, telah dituliskan mengenai penggunaan fungsi  $F$ . Fungsi  $F$  adalah: bagi  $XL$  menjadi empat bagian 8-bit:  $a, b, c$  dan  $d$ .  $F(XL) = ((S1, a + S2, b \mod 232) \text{ XOR } S3, c) + S4, d \mod 2$ . Agar dapat lebih memahami fungsi  $F$ , tahapannya dapat dilihat pada Gambar III.2 berikut ini :



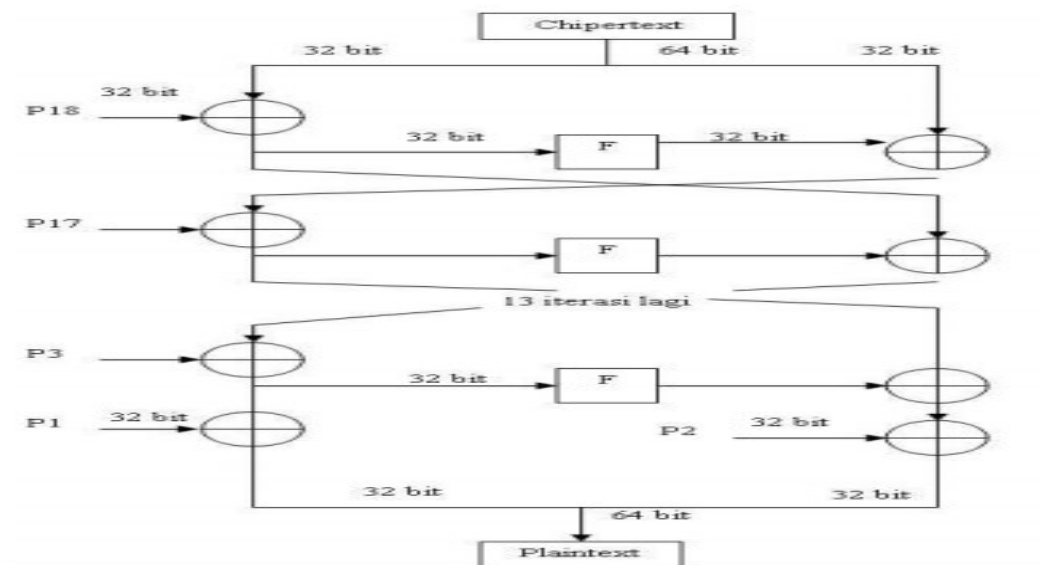
**Gambar III.2. Fungsi  $F$  Dalam *Blowfish***

### 3. Dekripsi Data

*Dekripsi* sama persis dengan *Enkripsi*, kecuali bahwa  $P1, P2, \dots, P18$

digunakan pada urutan yang berbalik (*reverse*). Algoritmanya dapat dinyatakan sebagai berikut :

For i = 1 to 16 do  $XR_i = XL_{i-1} \_ P_{19-i}$ ;  $XL_i = F[XR_i] \_ XR_{i-1}$ ; 21  $XL_{17} = XR_{16} \_ P_1$ ;  $XR_{17} = XL_{16} \_ P_2$ ;



**Gambar III.3. Blok Diagram Dekripsi Blowfish**

### III.2.2. Algoritma Rijndael

Algoritma kriptografi bernama Rijndael yang didesain oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes algoritma kriptografi pengganti DES yang diadakan oleh NIST (National Institutes of Standards and Technology) milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma Rijndael inilah yang kemudian dikenal dengan Advanced Encryption Standard (AES). Setelah mengalami beberapa proses standarisasi oleh NIST, Rijndael kemudian diadopsi menjadi standard algoritma kriptografi secara resmi pada 22 Mei 2002. Pada 2006, AES merupakan salah satu algoritma

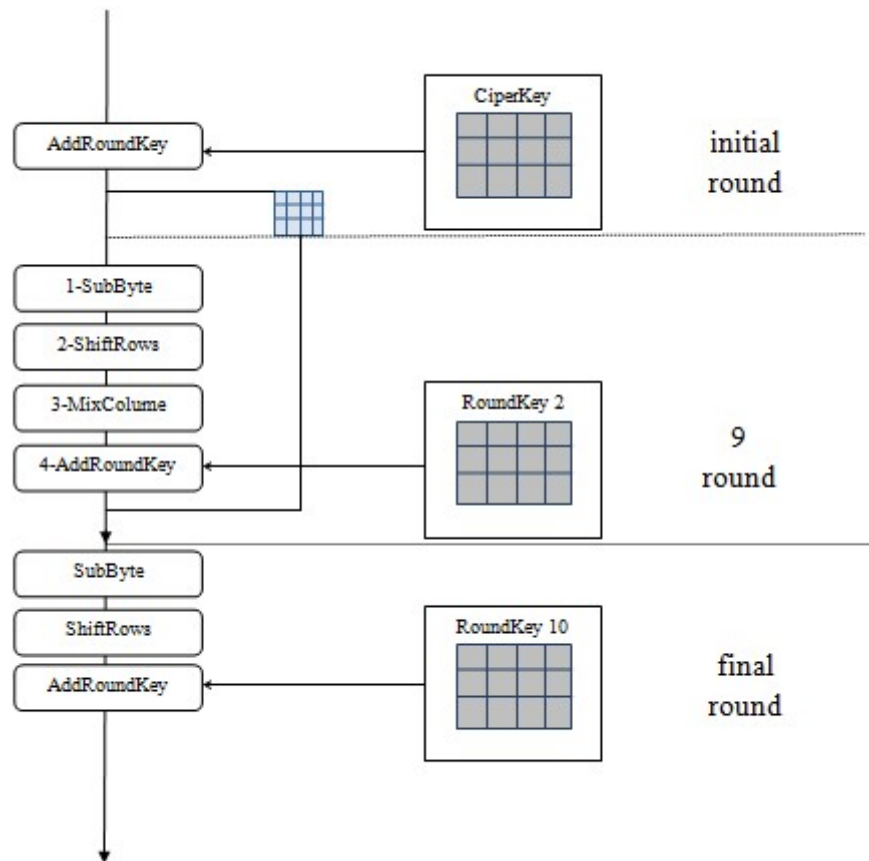
terpopuler yang digunakan dalam kriptografi kunci simetrik. AES ini merupakan algoritma block cipher dengan menggunakan sistem permutasi dan substitusi (P-Box dan S-Box) bukan dengan jaringan Feistel sebagaimana block cipher pada umumnya. Jenis AES terbagi 3, yaitu :

1. AES-128
2. AES-192
3. AES-256

Pengelompokkan jenis AES ini adalah berdasarkan panjang kunci yang digunakan. Angka-angka di belakang kata AES menggambarkan panjang kunci yang digunakan pada tiap-tiap AES. Selain itu, hal yang membedakan dari masing-masing AES ini adalah banyaknya round yang dipakai. AES-128 menggunakan 10 round, AES-192 sebanyak 12 round, dan AES-256 sebanyak 14 round. AES memiliki ukuran block yang tetap sepanjang 128 bit dan ukuran kunci sepanjang 128, 192, atau 256 bit. Tidak seperti Rijndael yang block dan kuncinya dapat berukuran kelipatan 32 bit dengan ukuran minimum 128 bit dan maksimum 256 bit. Berdasarkan ukuran block yang tetap, AES bekerja pada matriks berukuran 4x4 di mana tiap-tiap sel matriks terdiri atas 1 byte (8 bit). Sedangkan Rijndael sendiri dapat mempunyai ukuran matriks yang lebih dari itu dengan menambahkan kolom sebanyak yang diperlukan. Blok cipher tersebut dalam pembahasan ini akan diasumsikan sebagai sebuah kotak. Setiap plainteks akan dikonversikan terlebih dahulu ke dalam blok-blok tersebut dalam bentuk heksadesimal. Barulah kemudian blok itu akan diproses dengan metode yang akan dijelaskan. Secara umum metode yang digunakan dalam pemrosesan enkripsi

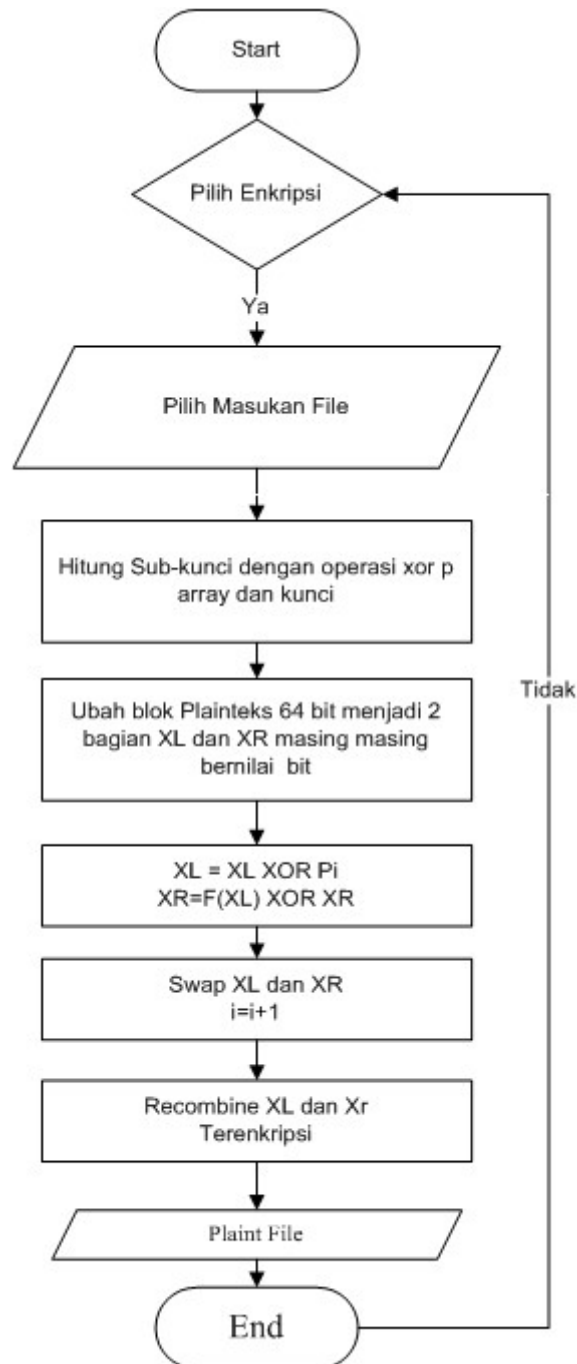


dalam algoritma ini dapat dilihat melalui Gambar berikut :

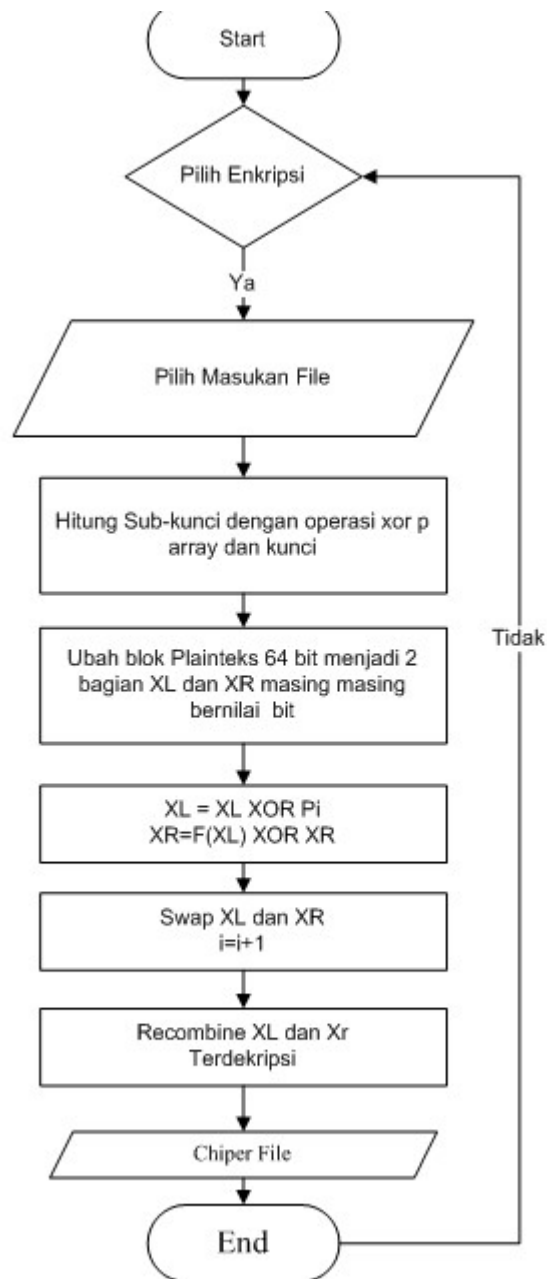


**Gambar III.4. Diagram Rijndael**

### III.2.3 Hybrid Algoritma Blowfish dan Rijndael



Gambar III.5. *Flowchat* Enkripsi Menggunakan *Hybrit* Algoritma Blowfish dan Rijndael



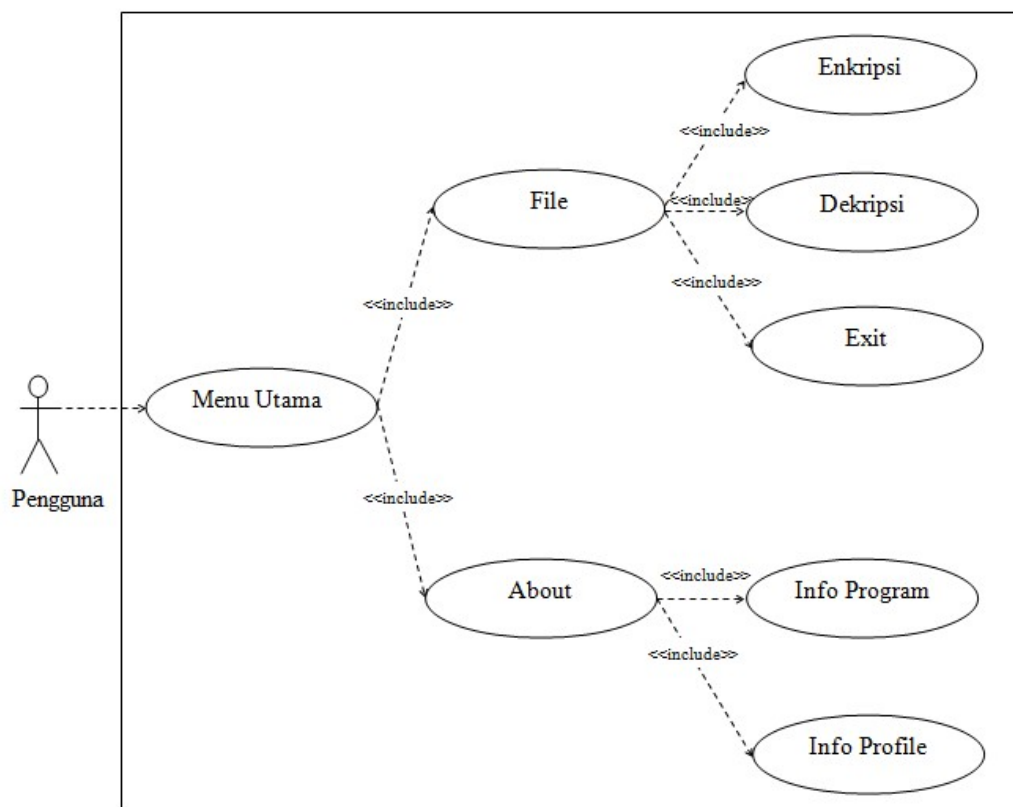
**Gambar III.6. Flowchat Dekripsi Menggunakan Hybrit Algoritma Blowfish dan Rijndael**

### III.3. Desain Sistem

Sebagai solusi dari permasalahan yang telah teridentifikasi dan untuk membuat sebuah sistem yang dapat berjalan dengan baik serta sesuai harapan yang diinginkan maka tentunya terlebih dahulu haruslah membuat tahapan perencanaan sistem berupa *use case diagram*, *class diagram*, *sequence diagram* *activity diagram*.

#### III.3.1 Use Case Diagram

Proses sistem pakar yang akan dirancang digambarkan dengan *use case diagram* yang terdapat pada Gambar III.7 :



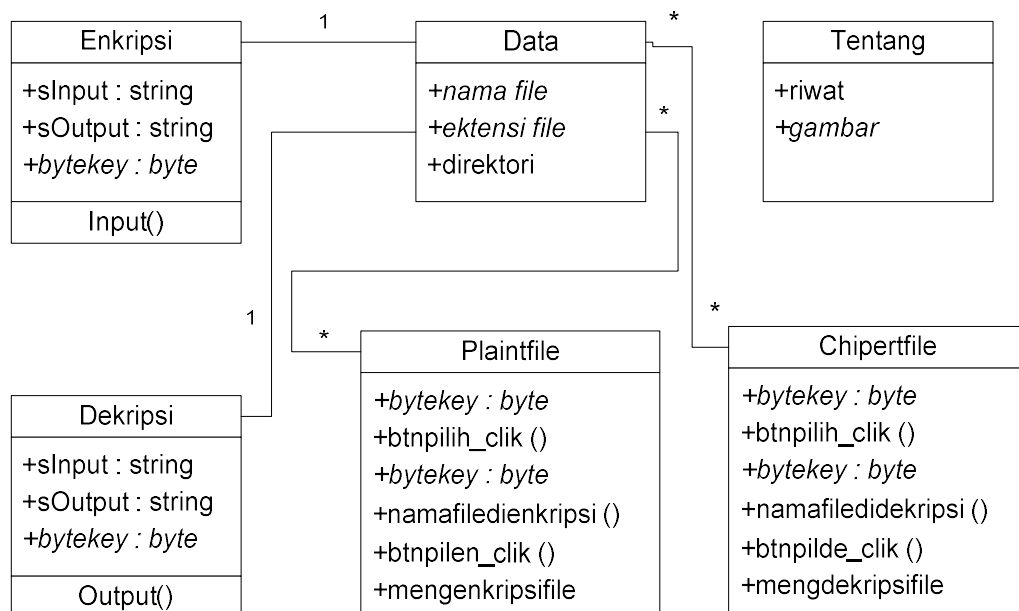
**Gambar III.7. Use Case Pemanfaatan Kriptografi Algoritma Blowfish**

#### ***Kriptografi Algoritma Rijndael dalam Pengamanan Data File***

Dari gambar use case diagram di atas, pengguna memulai/menjalankan

aplikasi, pengguna berfungsi sebagai actor yang harus menjalankan aplikasi proses Enkripsi, Dekripsi, Info Program dan Info Profile. Proses Enkripsi, pengguna memilih menu utama, File dan Enkripsi. Proses Dekripsi, pengguna memilih menu utama, File dan Dekripsi. Pengguna memilih Menu Utama, About, Info Program dan Info Profile.

### III.3.2. Class Diagram



**Gambar III.8. Class Diagram Pengamanan Data File**

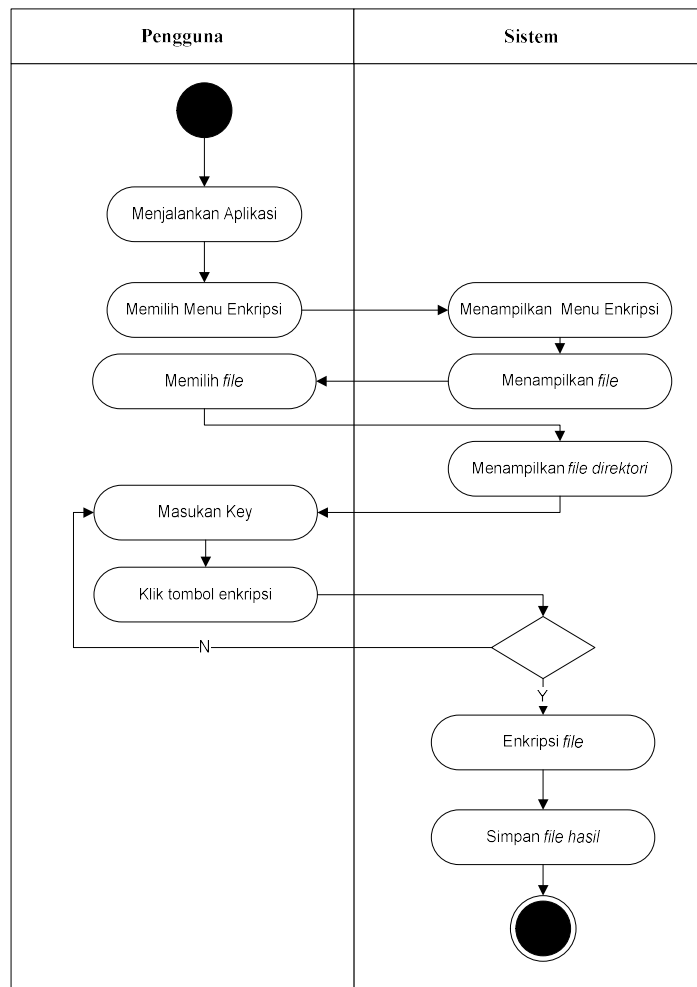
Dari gambar *class diagram* diatas, Atribut dari Enkripsi berisi sInput,sOutput dengan methods Input. Atribut Dekripsi berisi sInput, sOutput dan bytekey dengan Methods Output. Atribut Data berisi nama file ekstensi file dan direktori. Atribut dari Plaintfile, bytekey, btnpilih, namafiledienkripsi,btn, Mengenkrripsi. Atribut data Chiperfile diantaranya bytekey, btnpilih, bytekey, namafiledidekripsi,btn,mengenkrripsi.

### III.3.3. Activity Diagram

Alur proses yang telah digambarkan pada *use case diagram* diatas dijabarkan dengan *activity diagram* :

#### 1. *Activity Diagram Enkripsi File*

Aktiviti *Diagram* yang dilakukan dalam melakukan *Enkripsi File* terhadap sistem yang dapat diterangkan pada gambar III.9.



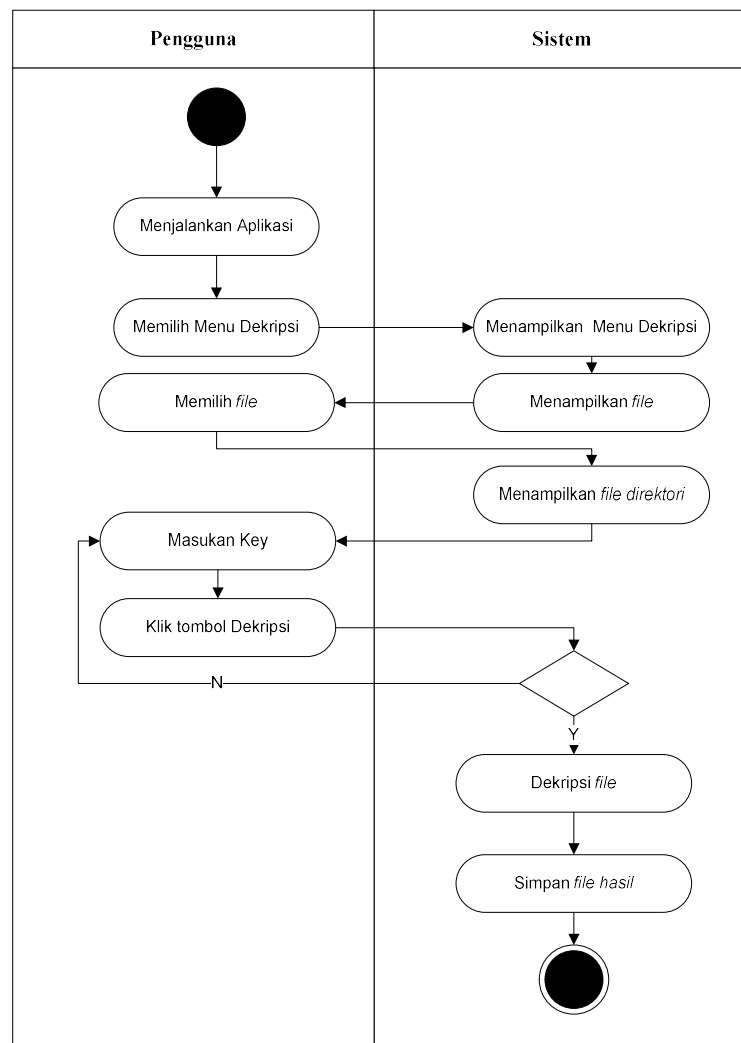
**Gambar III.9. Activity Diagram Enkripsi File**

Dari gambar *Activity Diagram Enkripsi File* diatas, pengguna Menjalankan aplikasi kemudian menu enkripsi sistem menampilkan menu enkripsi dan menampilkan browser file, pengguna memilih file yang akan di enkripsi oleh

sistem, memasukkan key dan menekan proses menjalankan sistem enkripsi jika Ya file akan terenkripsi jika tidak memasukan ulang data yg akan di enkripsi.

## 2. Activity Diagram Dekripsi File

Aktiviti Diagram yang dilakukan dalam melakukan *Dekripsi File* terhadap sistem yang dapat diterangkan pada gambar III.10.



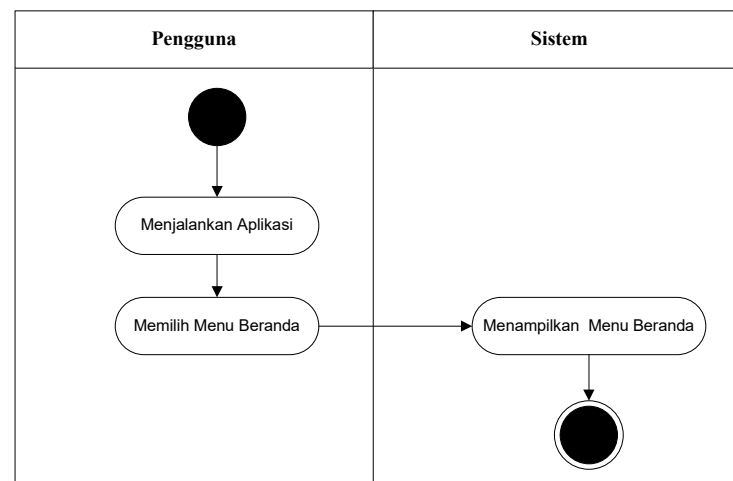
**Gambar III.10. Activity Diagram Dekripsi File**

Dari gambar *Activity Diagram Dekripsi File* diatas, pengguna Menjalankan aplikasi kemudian menu dekripsi sistem menampilkan menu dekripsi dan

menampilkan browser file, pengguna memilih file yang akan di dekripsi oleh sistem, memasukkan key dan menekan proses menjalankan sistem dekripsi jika file akan terenkripsi jika tidak memasukan ulang data yg akan di dekripsi.

### 3. *Activity Diagram* Menu Utama

Aktiviti *Diagram* yang menampilkan halaman beranda terhadap sistem yang dapat diterangkan pada gambar III.11.



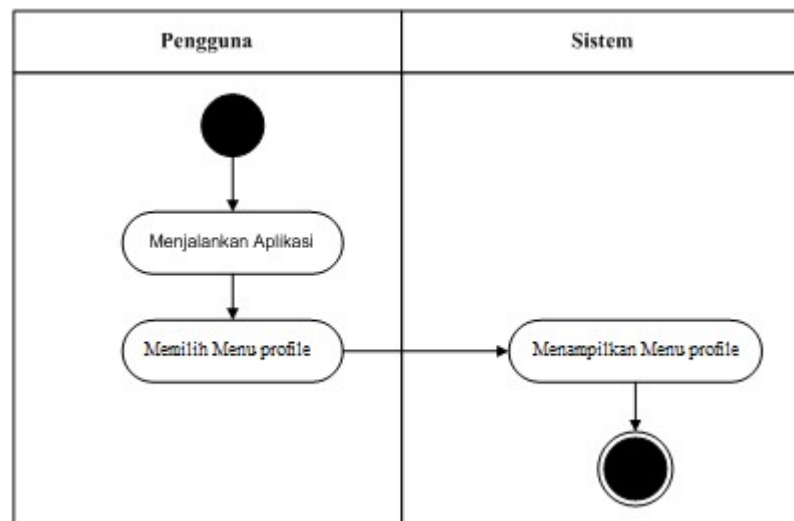
**Gambar III.11. *Activity Diagram* Menu Utama**

Dari gambar *Activity Diagram* menu utama, pengguna menjalankan aplikasi dan memilih menu sistem akan menampilkan menu beranda/menu utama.

### 4. *Activity Diagram* Info Profile

*Activity Diagram* yang menampilkan halaman tentang terhadap sistem yang dapat diterangkan pada gambar III.12.





**Gambar III.12. Activity Diagram Info Profile**

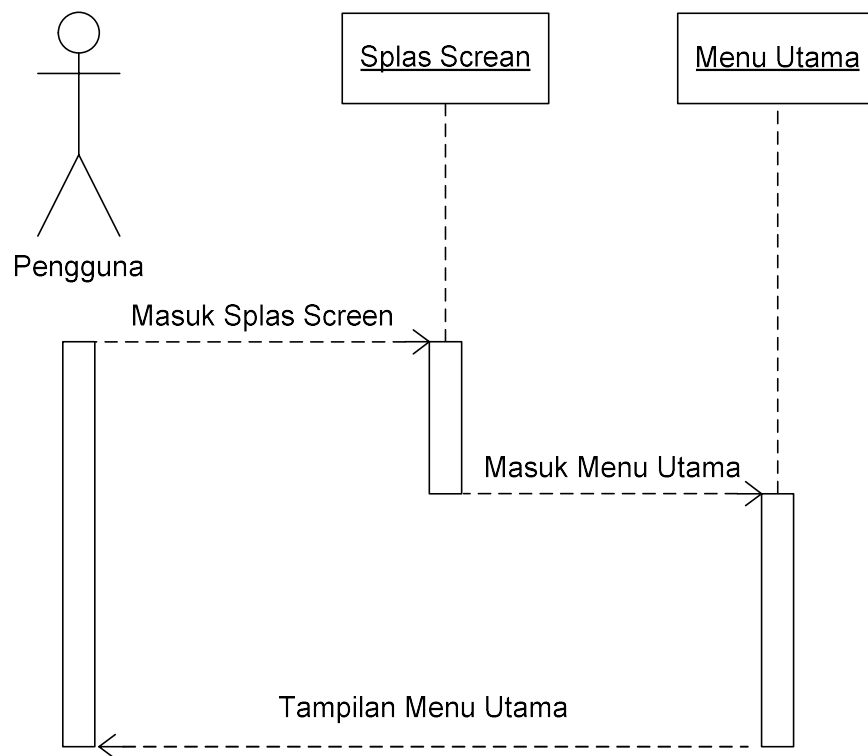
Dari gambar *Activity Diagram* info profile, pengguna menjalankan aplikasi dan memilih menu profile akan menampilkan data profile.

### III.3.4 Sequence Diagram

*Sequence Diagram* menggambarkan tentang skenario atau langkah langkah yang dilakukan sistem sebagai umpan balik (*feedback*) dari interaksi yang dilakukan pengguna. Berikut adalah *sequence diagram* pada perancangan Perangkat lunak pembelajaran kriptografi algoritma *Blofish* dan *Rijndael* :

#### 1. Sequence Diagram Menu Utama

Pada *Sequence Diagram* Menu Utama adalah proses dimana pengguna masuk ke menu utama setelah *splash screen*. Dalam menu utama pengguna dapat melihat menu Beranda, *Enkripsi*, *Dekripsi*, Tentang dann Keluar. Berikut tampilan *Sequence Diagram* Menu Utama :

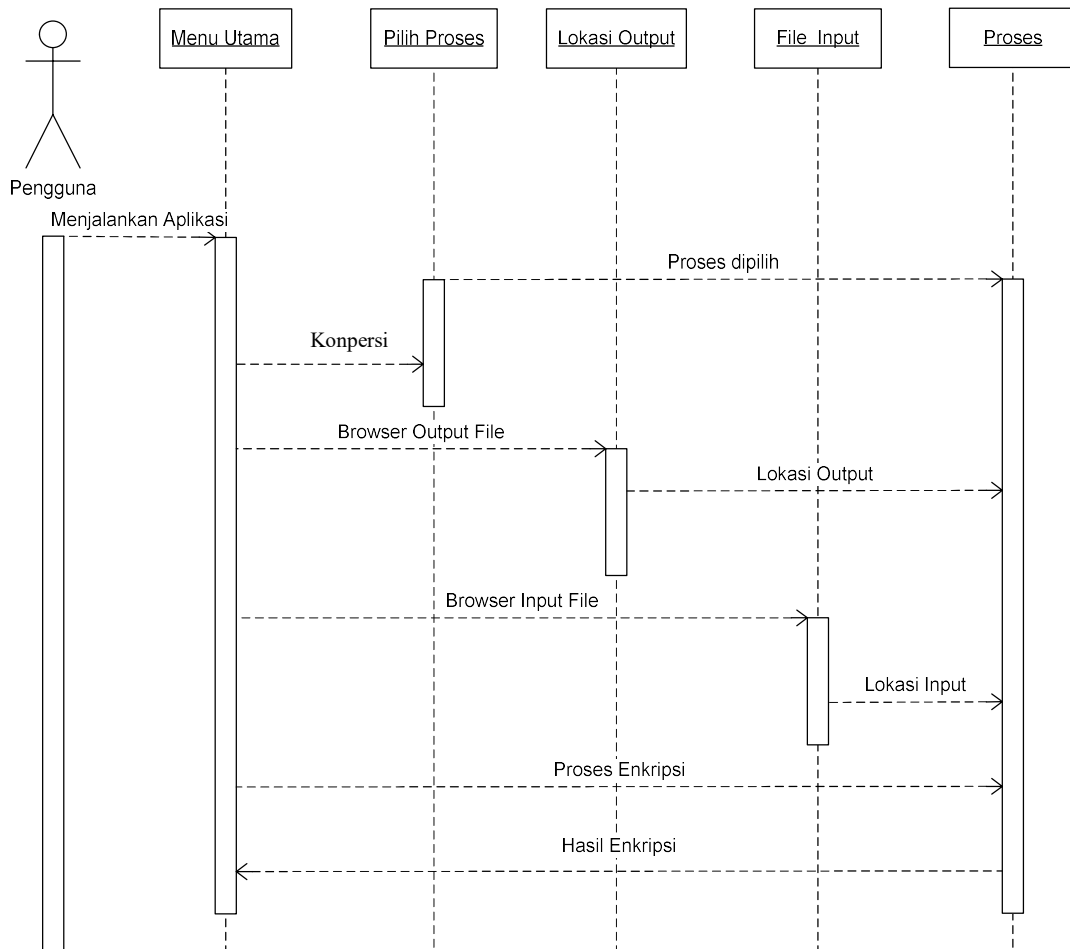


**Gambar III.13. *Sequence Diagram* Menu Utama**

Dari gambar *Sequence Diagram* Menu Utama, pengguna menjalankan aplikasi dan memilih menu utama akan menampilkan masuk ke Menu Utama.

## 2. Sequence Diagram Enkripsi

Pada *sequence diagram* menu *Enkripsi* adalah proses dimana pengguna Memilih *file* yang akan *dienkripsi* dan memasukkan *key* dan menekan tombol *Enripsi*. Berikut tampilan *sequence diagram Enkripsi* :

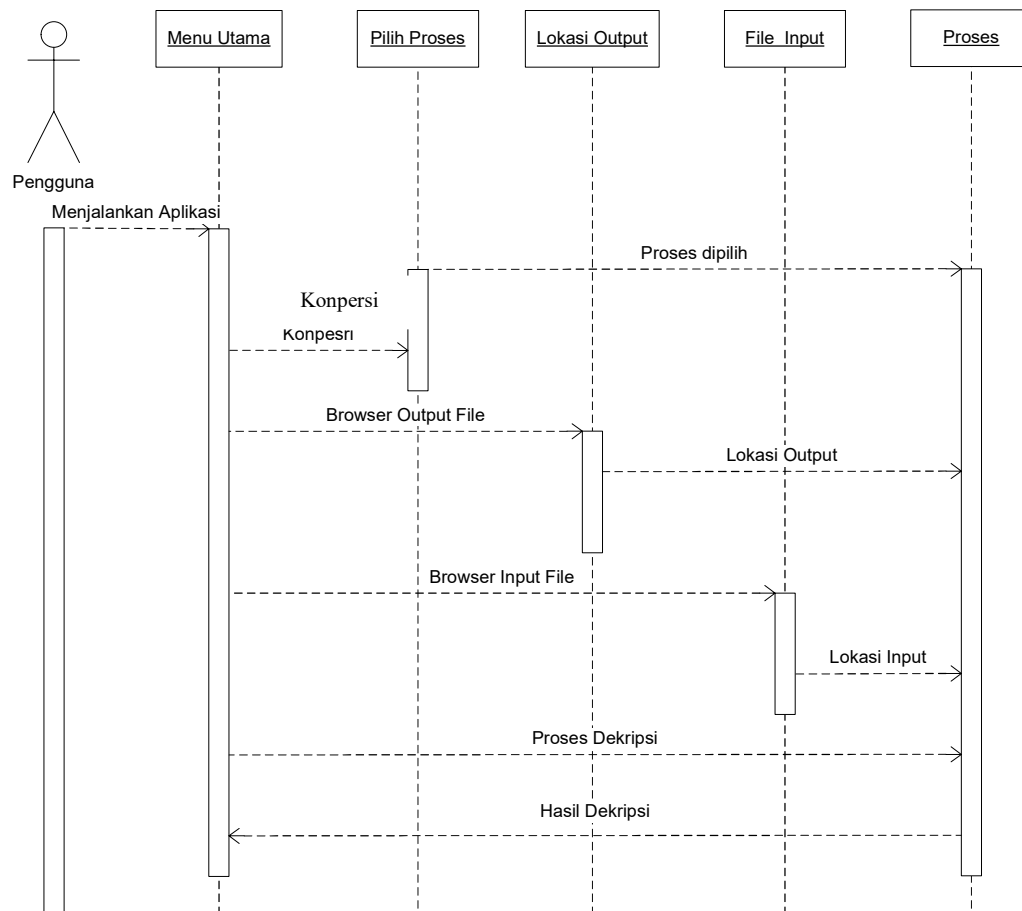


**Gambar III.14. Sequence Diagram Enkripsi**

Dari gambar *Sequence Diagram enkripsi diatas*, dimana pengguna menjalankan aplikasi dan sistem menampilkan menu utama, pengguna memilih browserinput file difile inputan kemudian memilih bworser output file untuk meletakkan hasil enkripsi, key, proses enkripsi, file berhasil di enkripsi.

### 3. Sequence Diagram Dekripsi

Pada *sequence diagram* menu *Enkripsi* adalah proses dimana pengguna Memilih *file* yang akan didekripsi dan memasukkan *key* dan menekan tombol *Enkripsi*. Berikut tampilan *Sequence Diagram Dekripsi* :



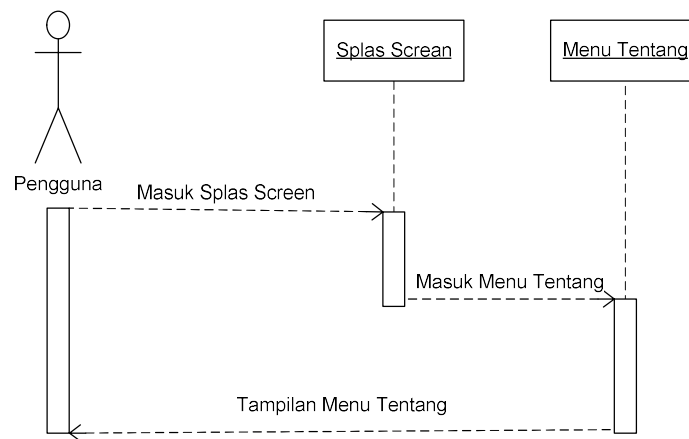
**Gambar III.15. Sequence Diagram Dekripsi**

Dari gambar *Sequence Diagram dekripsi diatas*, dimana pengguna menjalankan aplikasi dan sistem menampilkan menu utama, pengguna memilih browserinput file difile inputan kemudian memilih bworser output file untuk meletakkan hasil dekripsi, key, proses dekripsi, file berhasil di dekripsi.

#### 4. *Sequence Diagram* Tentang

Pada *Sequence Diagram* menu tentang, Berikut tampilan *Sequence Diagram*

Menu Tentang :



**Gambar III.16. *Sequence Diagram* Tentang**

Dari gambar *Sequence Diagram* tentang, pengguna menjalankan aplikasi an memilih tentang akan menampilkan splas Screan dan masuk ke Menu tentang, menampilkan menu tentang.

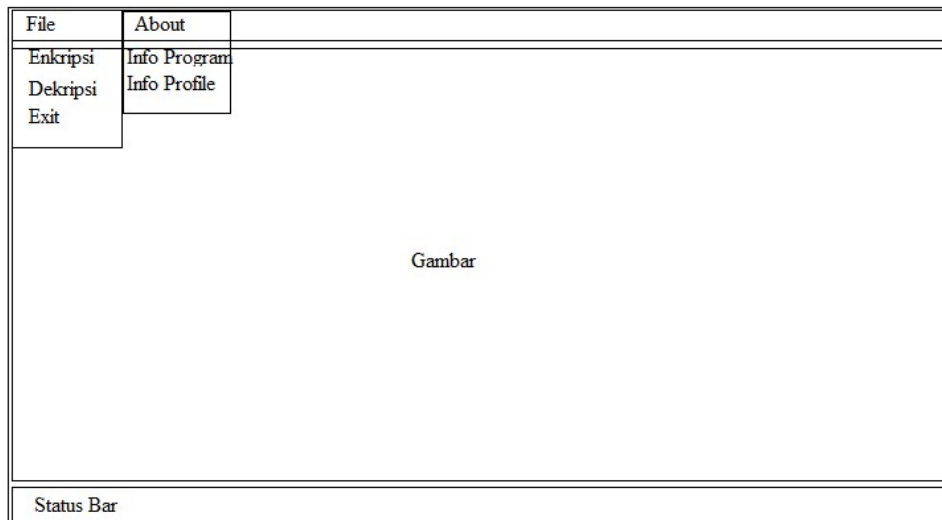
### III.4. **Desain *User Interface***

Desain *user interface* ini berfungsi untuk memberikan gambaran sistem yang akan diusulkan agar dapat dilihat secara lebih detail berdasarkan pada gambaran sistem keseluruhan.

#### 1. **Desain Menu Utama**

*Form* Home / Beranda berisi file dan About, file berisi Menu Enkripsi,

Dekripsi dan exit. About Berisi info Program dan Info profile. dilihat pada Gambar III.17 di bawah ini :



**Gambar III.17. Desain Tampilan Menu Utama**

## 2. Desain Menu Enkripsi

From Menu *Enkripsi* berfungsi mengenkripsi *file*, yang berisi 4 buah textbox (*textbox lokasi file*, *textbox file tujuan*, *textbox key*, *textbox keyconfer*), 3 buah label (*label file*, *labelfile tujuan*, *labelkey*), 2 buah *button*(*buttonbrower* *buttonproses*). dilihat pada Gambar III.18 di bawah ini :

 A screenshot of a graphical user interface window titled 'Enkripsi'. The window has a menu bar with 'File' and 'About'. Below the menu bar, the title 'Enkripsi' is displayed. The main area contains four labels on the left: 'File yang untuk dienkrpsi', 'File Tujuan', 'Key', and 'Konfir Key'. Each label is followed by a text input field. To the right of the first input field is a button labeled 'Browser'. To the right of the 'Key' and 'Konfir Key' input fields is a button labeled 'Proses'. At the bottom of the window is a status bar.

**Gambar III.18. Desain Tampilan Menu *Enkripsi***

### 3. Desain Menu *Dekripsi*

*From* Menu *Dekripsi* berfungsi mengenkripsi *file*, yang berisi 4 buah textbox (*textbox* lokasi *file*, *textbox* *file* tujuan, *textbox* key, *textbox* keyconfir), 3 buah label (*label* *file*, *label* *file* tujuan, *label* key), 2 buah *button* (*button* browser *button* proses). dilihat pada Gambar III.19 di bawah ini :

**Gambar III.19. Desain Tampilan Menu *Dekripsi***

### 4. Desain Menu Tentang

*Form* Menu Info Profile berisikan data penyusun menu tentang berisi 8 label dan 1 picturebox. dilihat pada Gambar III.20 di bawah ini :

File      About	
Info Profile	
Nama	XXXXXXXXXX
Nim	XXXXXXXXXX
Kelas	XXXXXXXXXX
Jurusan	XXXXXXXXXX
<div>Gambar</div>	

**Gambar III.20. Desain Tampilan Menu Info Profile**



