

BAB IV

PERANCANGAN

Pada bab ini akan dijelaskan secara rinci rancangan sistem aplikasi *file transfer* antar PC menggunakan algoritma RC4 128 bit dan AES 128 bit. Perancangan ini dilakukan untuk mengetahui mekanisme bagaimana proses komunikasi data dapat berlangsung pada program aplikasi ini, mulai dari pembuatan program untuk pengirim dan program untuk penerima, algoritma enkripsi RC4 128 bit dan AES 128 bit, hingga aktivitas yang terjadi pada jaringan tersebut dapat didokumentasikan.

Perancangan aplikasi *transfer file* ini dibagi menjadi 4 tahap, yaitu analisis kebutuhan, topologi perancangan, perancangan perangkat lunak, dan mekanisme *transfer file*. Analisis kebutuhan berisi tentang semua kebutuhan yang diperlukan dari sistem yang akan dibangun. Topologi perancangan berisi tentang mode jaringan yang akan dipakai untuk membangun perancangan program. Perancangan perangkat lunak terdiri dari tampilan program. Sedangkan mekanisme *file transfer* berisi tentang bagaimana program itu dijalankan, sehingga dapat berjalan sebagaimana mestinya.

4.1 Analisis Kebutuhan (*Requirement Analysis*)

Analisis kebutuhan dibagi menjadi 2 yaitu *user requirement* dan *system requirement* [SOM-11:83]. Pada *user requirement* menjelaskan tentang kebutuhan *file transfer* secara global, sedangkan *system requirement* menjelaskan kebutuhan *file transfer* secara lebih spesifik.

4.1.1 *User Requirement*

User requirement dari *file transfer* ini sebagai berikut:

- Penerima dapat menyimpan *file* yang telah dikirimkan oleh PC pengirim dalam bentuk telah terdekripsi.
- Penerima hanya menentukan IP yang dipakai serta *port* yang akan digunakan.

- c. Pengirim akan terhubung dengan menerima IP penerima, *port* dan menentukan *password* serta jenis enkripsi untuk melakukan akses koneksi kepada penerima.
- d. Pengirim dapat mengirimkan *file* hingga 100 MB. *File* yang dikirimkan akan otomatis terenkripsi (*ciphertext*).
- e. *File* yang telah diterima oleh penerima sudah dalam bentuk terdekripsi (*plaintext*).

4.1.2 System Requirement

System requirement berfungsi untuk mendefinisikan kebutuhan-kebutuhan yang harus disediakan oleh program aplikasi. Adapun diantaranya daftar kebutuhan sistem pada penerima dan pengirim, deskripsi *input output*, perancangan sistem.

- a. Kebutuhan Penerima

Tabel 4.1 Daftar Kebutuhan Sistem Penerima

Kebutuhan	Nama Use case
- Penerima memberikan IP yang dipakai, serta <i>port</i> kepada pengirim yang akan melakukan koneksi. - Sistem harus dapat mendekripsikan data sesuai dengan jenis enkripsi yang telah ditentukan oleh pengirim, yaitu algoritma AES 128 bit atau RC4 128 bit. - Sistem memberikan rincian tentang jumlah <i>file</i> yang di-kirimkan, nama perangkat, nama <i>file</i> dan jenis <i>file</i> serta ukuran <i>filenya</i> (KB). - Sistem harus dapat memberikan pemberitahuan ketika terputusnya koneksi	Penerima
- <i>File</i> ketika dikirim sudah otomatis tersimpan dalam bentuk telah terdekripsi sesuai dengan pilihan algoritma.	Output

b. Kebutuhan Pengirim

Table 4.2 Daftar Kebutuhan Sistem Pengirim

Kebutuhan	Nama <i>Use case</i>
- Pengirim menerima IP penerima, dan <i>port</i> yang digunakan oleh penerima. - Sistem harus dapat memberikan pilihan algoritma yang akan dipakai pengirim untuk mentransfer <i>file</i> yaitu algoritma AES 128 bit atau RC4 128 bit dan menentukan <i>password</i>. - Sistem memberikan rincian tentang jumlah <i>file</i> yang dikirimkan, nama perangkat, nama <i>file</i> dan jenis <i>file</i> serta ukuran <i>filenya</i> (KB).	Pengirim
- Mencari lokasi <i>file</i> yang akan dikirimkan. - <i>File</i> yang bisa dikirimkan hingga 100 MB. - <i>File</i> yang dikirimkan akan terenkripsi sesuai dengan pilihan algoritma.	Kirim <i>File (Input)</i>

c. Deskripsi pengirim dan penerima

Tabel 4.3 Definisi Pengirim dan Penerima

Kirim	
<i>Input</i>	<i>Input</i> pada menu kirim <i>file</i> adalah masukan data yang akan dienkripsi oleh PC pengirim sebelum dikirimkan.
<i>Output</i>	<i>Output</i> pada program aplikasi merupakan keluaran atau hasil enkripsi yaitu berupa data <i>ciphertext</i> yang tidak dapat dibaca.
Dekripsi	
<i>Input</i>	<i>Input</i> pada program aplikasi merupakan masukan data berupa <i>ciphertext</i> untuk didekripsi ke data sebenarnya atau <i>plaintext</i> .
<i>Output</i>	<i>Plaintext</i> .

d. Perancangan sistem *transfer file*

Perancangan sistem transfer file terdiri dari proses merancang penerima, koneksi pengirim dan pengiriman *file*.

Table 4.4 Perancangan Sistem Penerima

Tujuan	Penerima
Deskripsi	Penerima ini menentukan IP serta <i>port</i> yang digunakan. Tekan tombol “Jalankan Penerima File”.
Pra-kondisi	PC penerima belum bisa menerima <i>file</i>
Pasca-kondisi	PC telah siap menerima <i>file</i> .
Aliran Utama	
Aksi	Reaksi
Menentukan IP serta <i>port</i> yang digunakan, tekan “Jalankan Penerima File”.	Membuat dan meregister <i>channel</i> TCP ke <i>cannel services</i> . Mengeset nama aplikasi serta meregister nama <i>services</i> PC siap untuk menerima <i>file</i> . Memberikan rincian tentang jumlah <i>file</i> yang dikirimkan , perangkat yang mengirimkan <i>file</i> , nama dan jenis <i>file</i> serta ukuran <i>filenya</i> (KB).
Alternatif 1: Jika IP dan <i>port</i> belum dipilih	
	Sistem menampilkan pemberitahuan
Alternatif 2: Jika <i>port</i> yang digunakan sama	
Memasukkan <i>port</i> yang sudah digunakan.”	Program aplikasi tidak dapat digunakan.
Alternatif 3: Jika koneksi penerima terputus	
Ketika PC akan melakukan kirim <i>file</i> , koneksi pada penerima tiba-tiba terputus.	Program aplikasi akan memberikan pemberitahuan.

Table 4.5 Perancangan Sistem Koneksi Pengirim

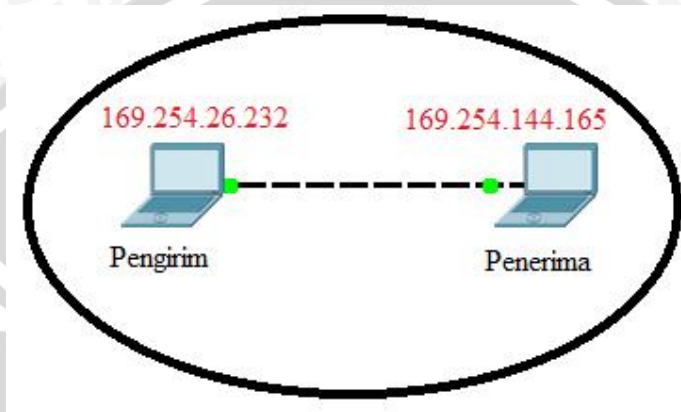
Tujuan	Koneksi Pengirim
Deskripsi	Melakukan koneksi antara pengirim ke penerima
Pra-kondisi	Pengirim belum terkoneksi dengan penerima
Pasca-kondisi	Pengirim telah terkoneksi dengan penerima.
Aliran Utama	
Aksi	Reaksi
Memasukan IP dan <i>port</i> yang digunakan penerima, memilih algoritma dan menentukan <i>password</i> , tekan “Jalankan Pengirim”	Membuat dan meregister <i>channel</i> TCP ke <i>channel services</i> . Mengkoneksikan dengan penerima menggunakan alamat <i>services</i> penerima. Program <i>transfer file</i> dapat digunakan.
Alternatif 1: Jika IP, <i>port</i>, algoritma enkripsi serta <i>password</i> belum dimasukkan	
	Sistem menampilkan pemberitahuan
Alternatif 2: Koneksi pengirim terputus	
	Sistem menampilkan pemberitahuan

Table 4.6 Perancangan Sistem Pengiriman *File*

Tujuan	Menerima data.
Deskripsi	Setelah melakukan koneksi dengan benar dan menekan “Jalankan Penerima <i>File</i> ”, langkah selanjutnya adalah mengirimkan <i>file</i> .
Pra-kondisi	<i>File</i> dalam bentuk <i>chipertext</i>
Pasca-kondisi	<i>File</i> dalam bentuk <i>plaintext</i> .
Aliran Utama	
Aksi	Tanggapan Sistem
Memilih data yang akan dikirim dengan menekan “Kirim <i>File</i> ”.	Program <i>transfer file</i> dapat digunakan. Data yang diterima berupa <i>plaintext</i> (berada pada <i>folder</i> yang di tentukan). Memberikan rincian tentang jumlah <i>file</i> yang dikirimkan , perangkat yang mengirimkan <i>file</i> , nama dan jenis <i>file</i> serta ukuran <i>filenya</i> (KB).
Alternatif 1: Jika menerima data yang sama	
Memilih data yang telah dikirim dengan menekan “Kirim <i>File</i> ”	<i>File</i> akan terduplikasi.

4.2 Topologi Perancangan Jaringan

Topologi perancangan jaringan pada aplikasi ini seperti ditunjukkan pada gambar 4.1. Pada tugas akhir ini penulis mengimplementasikan topologi aplikasi *transfer file* ke dalam jaringan *ad-hoc* karena lebih mudah untuk mengimplementasikannya dan topologi ini cocok untuk aplikasi yang bersifat *real time*.



Gambar 4.1 Topologi Perancangan Jaringan

Keterangan gambar :

a. PC 1 (Pemegang SSID)

Pada langkah pertama *setting* salah satu PC yang ingin melakukan *link*, harus terlebih dahulu memiliki nama SSID. Anda tidak perlu memikirkan apakah ada tanda jaringan sudah bekerja, karena 2 PC dengan koneksi *ad-hoc*, jaringan baru bisa bekerja bila terdapat PC dalam keadaan hidup. Satu PC memegang sebuah nama SSID, dan PC lainnya yang melihat atau mendengar *broadcast* dari PC pertama agar dapat bersatu ke dalam jaringan.

b. PC 2 (Penerima *broadcast*)

Bila *setting* berjalan benar atau dalam *ad-hoc* mode, maka PC 2 akan menampilkan satu *broadcast* atau satu nama SSID dari PC pertama dan anda sudah dapat melakukan *link* ke PC 2. PC sudah saling terkoneksi.

IP PC ini menggunakan IP private yaitu range 169.0.0.0 s/d 169.255.255.255 karena IP ini hanya untuk mengakses jaringan lokal. Untuk

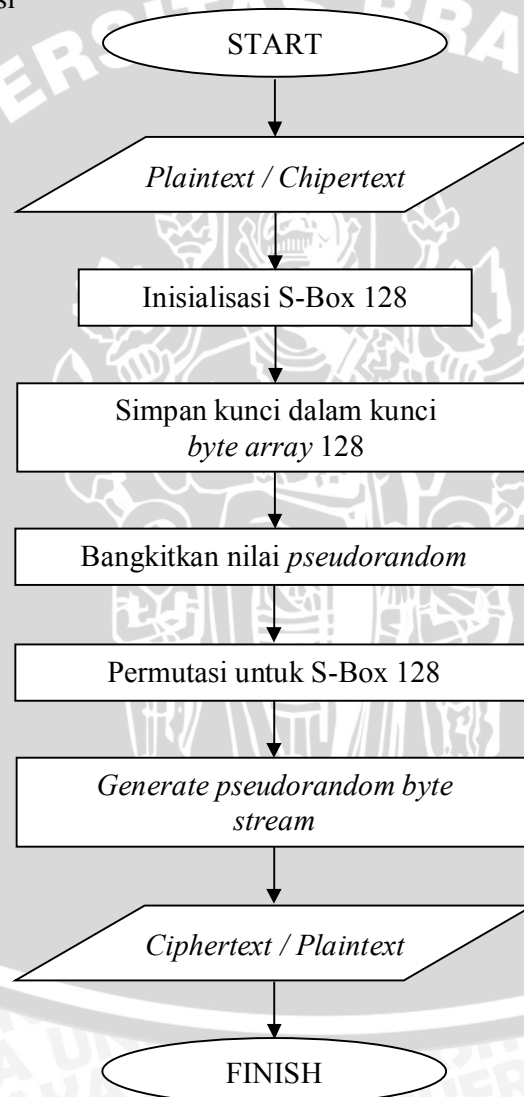
percobaan ini, IP penerima adalah 169.254.26.232, maka pengirim harus terhubung dengan IP penetima agar dapat menggunakan aplikasi ini.

4.3 Mekanisme Proses

Mekanisme proses terdiri dari mekanisme algoritma RC4 128 bit dan AES 128 bit, *file transfer* pada sisi penerima dan pengirim.

4.3.1 Algoritma RC4

- Enkripsi



Gambar 4.2 Flowchart Enkripsi dan Dekripsi Algoritma RC4 128 bit

(Sumber: WIB-10 : 56)

Langkah-langkah yang akan ditempuh oleh program dalam menjalankan proses tersebut meliputi hal-hal berikut ini:

1. User memasukkan *secret key* yang akan digunakan dalam proses enkripsi/dekripsi.
2. Lakukan proses inialisasi awal S-Box berdasarkan indeksinya.
 - a. for $i = 0$ to 127
 - b. $K[i] = \text{Kunci } [i \bmod \text{length}]$: memilih kunci yang akan digunakan
3. Simpan *secret key* yang telah dimasukkan user ke dalam array 128 byte secara berulang sampai array terisi penuh.


```
i = 0 ; j = 0
for I = 0 to 127
{
    J = (j + S[i] + K[i]) mod 128
    Swap S[i] dan S[j]
}

```
4. Bangkitkan nilai *pseudorandom* berdasarkan nilai *key sequence*.


```
i = (i + 1) mod 128
j = (j + S[i]) mod 128
swap S[i] dan S[j]
t = (S[i] + S[j]) mod 128
K = S[t]

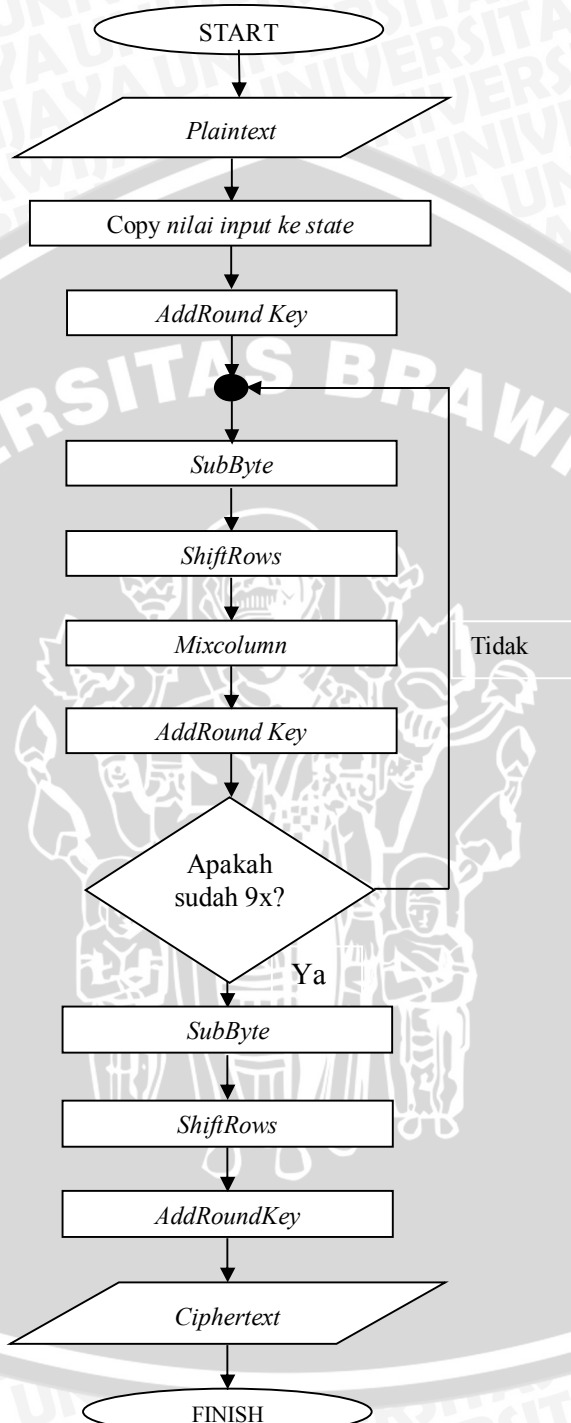
```
5. Lakukan proses permutasi/transposisi nilai dalam S-Box selama 128 kali.
6. Bangkitkan nilai *pseudorandom key byte stream* berdasarkan indeks dan nilai S-Box.
7. Lakukan operasi XOR antara *plaintext/ciphertext* dan *pseudorandom key byte stream* untuk menghasilkan *ciphertext/plaintext*.

- Dekripsi

Pada metode ini, proses dekripsi akan berjalan sama dengan proses enkripsinya sehingga hanya ada satu fungsi yang dijalankan untuk menjalankan kedua proses tersebut.

4.3.2 Algoritma AES

- Enkripsi



Gambar 4.3 Flowchart Enkripsi Algoritma AES 128 bit

(Sumber: WIB-10 : 56)

Langkah-langkah yang akan ditempuh oleh program dalam menjalankan proses enkripsi tersebut meliputi hal-hal berikut ini:

1. Masukkan *plaintext* dan *password*

```
this.state = new byte[4, besar_block];  
for (int i = 0; i < (4 * besar_block); ++i)
```

2. *Copy* nilai dari *input* ke *state*

```
this.state[i % 4, i / 4]=input[i];
```

3. Lakukan putaran awal

```
AddRoundKey(0);
```

4. *Round* = 1

```
for (int round = 1;
```

5. Ulangi selama *round*

```
round <=(jumlah_round - 1 );
```

```
SubBytes();
```

```
ShiftRows();
```

```
MixColumns();
```

```
AddRoundKey(round);
```

6. Akhir ulang

```
round++)
```

7. Lakukan putaran terakhir

```
SubBytes();
```

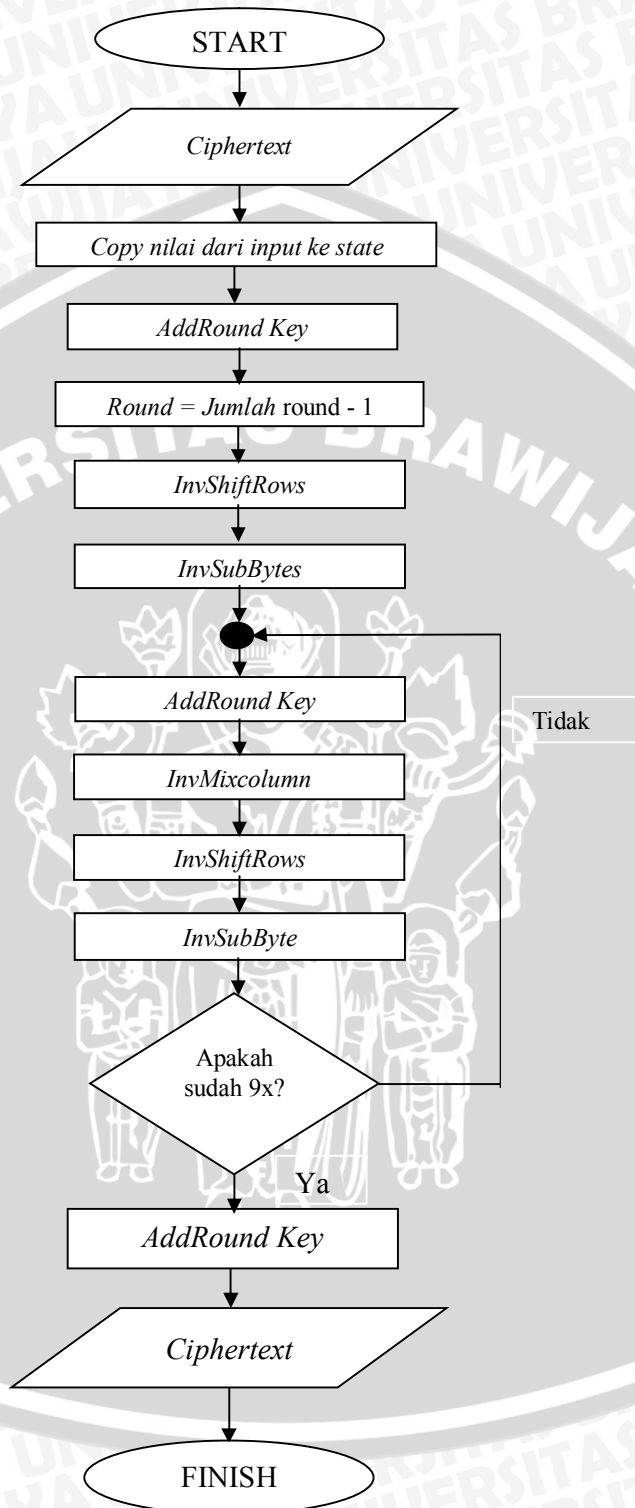
```
ShiftRows();
```

```
AddRoundKey(jumlah_round);
```

8. Masukkan hasil pengkodean input yang tersimpan dalam *state* ke *output* agar mendapatkan *ciphertext*.

```
for (int i = 0; i < (4 * besar_block); ++i)  
{  
    output[i] = this.state[i % 4, i / 4];  
}
```

- Dekripsi



Gambar 4.4 Flowchart Dekripsi Algoritma AES 128 bit

(Sumber: WIB-10 : 56)

Langkah-langkah yang akan ditempuh oleh program dalam menjalankan proses deskripsi antara lain:

1. Masukkan input *chipertext* dan *password* yang sama untuk melakukan enkripsi.

```
this.state = new byte[4, besar_block];  
for (int i = 0; i < (4 * besar_block); ++i)
```

2. *Copy* nilai dari *input* ke *state*

```
this.state[i % 4, i / 4] = input[i];
```

3. Melakukan putaran awal

```
AddRoundKey(jumlah_round);
```

4. *Round* = jumlah_round - 1

```
for (int round = jumlah_round - 1;
```

5. Ulang selama *round*

```
round >= 1;  
InvShiftRows();  
InvSubBytes();
```

6. Akhir ulang

```
AddRoundKey(round);  
InvMixColumns();  
InvShiftRows();  
InvSubBytes();
```

7. Melakukan putaran terakhir

```
AddRoundKey(0);
```

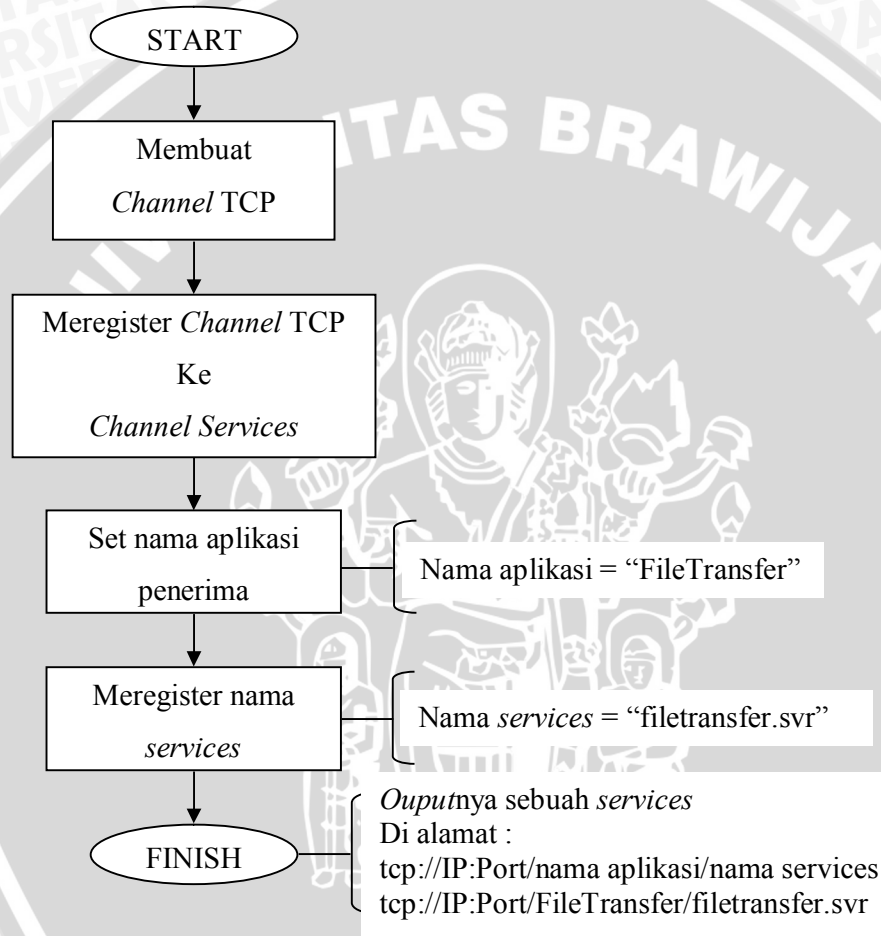
8. Masukkan hasil pengkodean *input* yang tersimpan dalam *state* ke *output* agar menghasilkan *plaintext*.

```
for (int i = 0; i < (4 * besar_block); ++i)  
{  
    output[i] = this.state[i % 4, i / 4];  
}
```


4.3.3 File Transfer

File transfer adalah program yang digunakan pada komputer berbasis windows untuk mentransfer file (*software* atau dokumen). *File transfer* ini terdapat 2 aplikasi yaitu untuk penerima dan pengirim.

4.3.3.1 Penerima

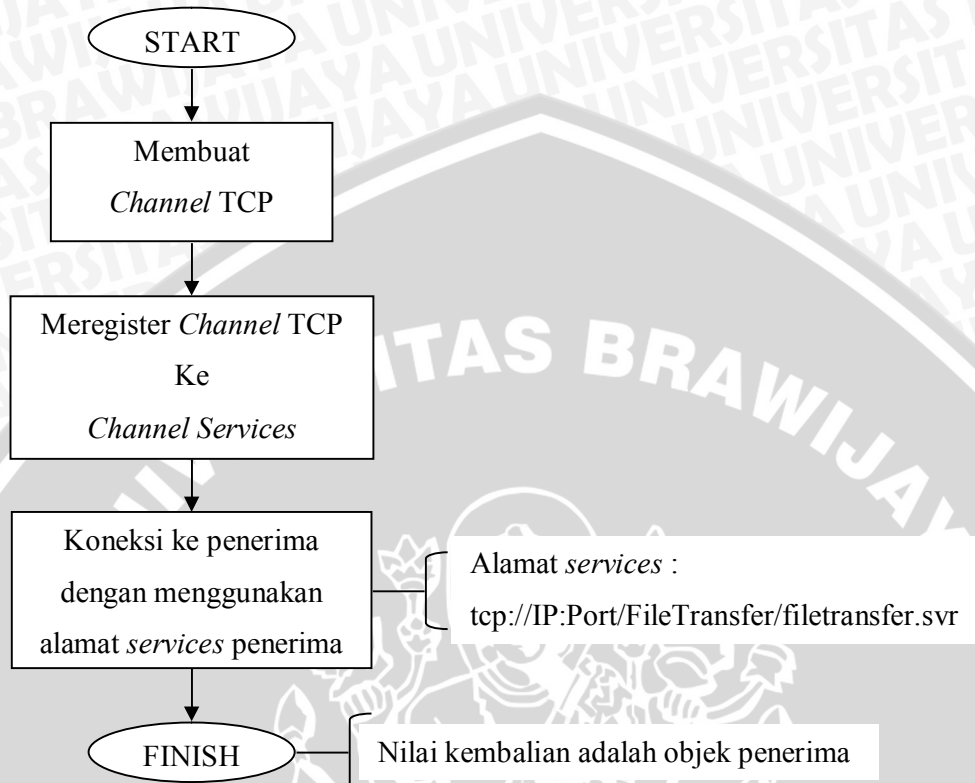


Gambar 4.5 Flowchart Aplikasi Penerima

(Sumber: Perancangan)

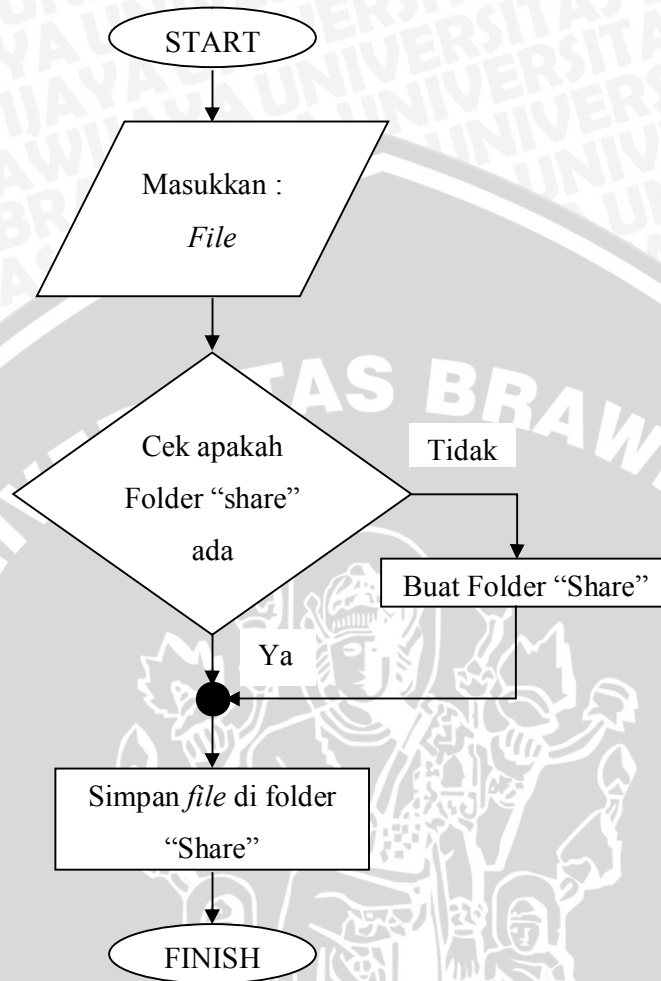
4.3.3.2 Pengirim

a. Koneksi Pengirim ke Penerima



Gambar 4.6 Flowchart Aplikasi Pengirim
(Sumber : Perancangan)

b. Pengirim Mengirim File ke Penerima



Gambar 4.7 Flowchart Aplikasi Mengirim
(Sumber : Perancangan)

4.4 Rancangan Interface

Rancangan *interface* aplikasi dapat dilihat pada gambar 4.8 dan gambar 4.9 berikut ini.

Row	Computer name	Filename	Size (KB)

Gambar 4.8 Rancangan *Interface* Aplikasi Penerima
(Sumber : Perancangan)

Row	Computer name	Filename	Size (KB)

Gambar 4.9 Rancangan *Interface* Aplikasi Pengirim
(Sumber : Perancangan)

Program aplikasi mempunyai 2 buah menu utama yaitu aplikasi untuk penerima dan pengirim. Aplikasi penerima ini digunakan sebagai penghubung agar dapat diakses oleh pengirim. Pengirim adalah pengguna aplikasi *transfer data* untuk mengirimkan data kepada PC lain sebagai penerima.